

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ
по дисциплине «Введение в нереляционные базы данных»
Тема: Приложение-сервис для организации субботников

Студентка гр. 2304

Иванова М.А.

Студент гр. 2304

Климова К.М.

Студентка гр. 2304

Лобанов Е.А.

Студент гр. 2304

Федоров Е.М.

Студент гр. 2303

Шумилов А.В.

Преподаватель

Заславский М.М.

Санкт-Петербург

2025

ЗАДАНИЕ

Студентка Иванова М.А.

Студентка Климова К.М.

Студент Лобанов Е.А.

Студент Федоров Е.М.

Студент Шумилов А.В.

Тема проекта: Приложение-сервис для организации субботников.

Исходные данные:

Необходимо сделать сервис, где можно как самому организовать субботник, так и присоединится к чужому субботнику. В сервисе должны быть помимо стандартных страниц (Профили пользователей, поиск, комментарии, отзывы) также и страницы с персональной и общей статистикой (кто и где участвовал, сколько км убрали).

Содержание пояснительной записи:

«Содержание», «Введение», «Сценарии использования», «Модель данных», «Разработанное приложение», «Выводы», «Приложения», «Список использованных источников».

Предполагаемый объем пояснительной записи:

Не менее 20 страниц.

Дата выдачи задания: 03.02.2025

Дата сдачи реферата: 05.06.2025

Дата защиты реферата: 05.06.2025

Студентка гр. 2304

Иванова М.А.

Студент гр. 2304

Климова К.М.

Студентка гр. 2304

Лобанов Е.А.

Студент гр. 2304

Федоров Е.М.

Студент гр. 2303

Шумилов А.В.

Преподаватель

Заславский М.М.

АННОТАЦИЯ

В рамках ИДЗ было разработано веб-приложение по организации субботников. Приложение включает функционал для просмотра, создания и редактирования субботников, а также для участия пользователей в субботниках. Кроме того, сайт отображает статистические данные о каждом пользователе и субботнике в отдельности и о сервисе в целом. Реализована система фильтрации и поиска для всех существующих в системе сущностей.

При разработке приложения были использованы следующие языки программирования: Python, TypeScript, JavaScript, а также НСУБД ArangoDB и Docker.

Исходный код приложения доступен по ссылке: [nosql1h25-cleanday](#).

SUMMARY

As part of the individual homework assignment, a web application for organizing community clean-up events («subbotniks») was developed. The application includes functionality for viewing, creating, and editing subbotniks, as well as for users to participate in them. In addition, the site displays statistical data for each user and each subbotnik individually, as well as for the service as a whole. A filtering and search system has been implemented for all entities present in the system.

The following programming languages and technologies were used in developing the application: Python, TypeScript, as well as the ArangoDB NoSQL database and Docker.

The source code of the application is available at: [nosql1h25-cleanday](#).

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	11
АКТУАЛЬНОСТЬ ПРОБЛЕМЫ.....	11
ПОСТАНОВКА ЗАДАЧИ.....	11
ПРЕДЛАГАЕМОЕ РЕШЕНИЕ.....	12
КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ.....	12
МАКЕТ UI.....	13
СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ	13
РАБОТА С ПОЛЬЗОВАТЕЛЕМ-ДЕЙСТВУЮЩИМ ЛИЦОМ.....	13
<i>Регистрация нового пользователя:</i>	13
<i>Авторизация пользователя</i>	14
<i>Просмотр информации о своем профиле</i>	15
<i>Изменение информации о своем профиле</i>	16
<i>Выход из профиля</i>	17
РАБОТА СО ВСЕМИ ПОЛЬЗОВАТЕЛЯМИ.....	18
<i>Просмотр всех пользователей</i>	18
<i>Поиск пользователей по совпадению</i>	19
<i>Сложный поиск пользователей по атрибутам</i>	19
<i>Построение графика о пользователях</i>	20
<i>Фильтрация пользователей по атрибуту</i>	21
<i>Просмотр информации о пользователе</i>	22
<i>Просмотр списка субботников, в которых участвовал пользователь</i>	22
<i>Просмотр списка субботников, которые организовал пользователь</i>	23
РАБОТА СО ВСЕМИ СУББОТНИКАМИ.....	23
<i>Просмотр всех субботников</i>	23
<i>Поиск субботников по совпадению</i>	24
<i>Сложный поиск субботников по атрибутам</i>	24
<i>Построение графика о субботниках</i>	26
<i>Фильтрация субботников по атрибуту</i>	26
<i>Фильтрация субботников с помощью чекбокса «Предстоящие субботники»</i>	27
<i>Фильтрация субботников с помощью чекбокса «С моим участием»</i>	27
<i>Просмотр информации о субботнике</i>	28
<i>Просмотр списка всех участников субботника</i>	29
<i>Просмотр списка участников субботника, подходящих под определенное условие</i>	29
<i>Просмотр списка участников субботника с определенным статусом участия</i>	30
<i>Просмотр истории активности участников субботника</i>	30
<i>Участие в субботнике</i>	31
<i>Изменить участие в субботнике.</i>	31

<i>Добавление комментария</i>	32
РАБОТА ПО ОРГАНИЗАЦИИ СУББОТНИКОВ	33
<i>Просмотр всех организованных пользователем субботников</i>	33
<i>Организация нового субботника.....</i>	33
<i>Удаление субботника.....</i>	35
<i>Отмена субботника</i>	36
<i>Изменение субботника</i>	36
<i>Важное замечание!</i>	37
ИНФОРМАЦИЯ О СЕРВИСЕ:	38
<i>Просмотр статистики сервиса.....</i>	38
<i>Экспорт данных</i>	38
<i>Импорт данных.....</i>	39
НЕРЕЛЯЦИОННАЯ МОДЕЛЬ	40
ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ МОДЕЛИ	40
ОПИСАНИЕ НАЗНАЧЕНИЙ КОЛЛЕКЦИЙ, ТИПОВ ДАННЫХ И СУЩНОСТЕЙ.....	40
<i>Типы данных</i>	40
Базовые	40
Пользовательские.....	41
<i>Сущности.....</i>	42
User.....	42
Participation.....	42
Requirement.....	43
CleanDay	43
City	43
Comment.....	43
Location	44
Image	44
Log.....	44
<i>Коллекции.....</i>	44
Коллекции узлов:	44
Коллекции ребер:	45
ОЦЕНКА ОБЪЕМА ИНФОРМАЦИИ, ХРАНИМОЙ В МОДЕЛИ.....	46
<i>Объём памяти для хранения всех видов объектов.....</i>	46
<i>Зависимость общего объёма от количества объектов.....</i>	47
Избыточность данных	49
НПРАВЛЕНИЕ РОСТА МОДЕЛИ	49
ПРИМЕРЫ ДАННЫХ	50
ПРИМЕРЫ ЗАПРОСОВ	50
<i>Регистрация нового пользователя</i>	50
<i>Авторизация пользователя</i>	51
<i>Просмотр информации о своём профиле</i>	51

<i>Обновление данных пользователя</i>	52
<i>Просмотр всех пользователей</i>	53
<i>Поиск пользователей по совпадению</i>	53
<i>Сложный поиск пользователей по атрибутам</i>	53
<i>Фильтрация пользователей по атрибуту</i>	54
<i>Просмотр информации о пользователе</i>	54
<i>Просмотр списка субботников, в которых участвовал пользователь</i>	54
<i>Просмотр списка субботников, которые организовал пользователь</i>	54
<i>Просмотр всех субботников</i>	55
<i>Поиск субботников по совпадению</i>	55
<i>Сложный поиск субботников по атрибутам</i>	56
<i>Фильтрация субботников с помощью чекбокса «Предстоящие субботники»</i>	56
<i>Фильтрация субботников с помощью чекбокса «С моим участием»</i>	56
<i>Просмотр информации о субботнике</i>	56
<i>Просмотр списка всех участников субботника</i>	57
<i>Просмотр списка участников субботника, подходящих под определенное условие</i>	57
<i>Просмотр списка участников субботника с определенным статусом участия</i>	58
<i>Просмотр истории активности участников субботника</i>	58
<i>Участие в субботнике</i>	59
<i>Изменить участие в субботнике</i>	59
<i>Добавление комментария</i>	60
<i>Просмотр всех организованных пользователем субботников</i>	60
<i>Организация нового субботника</i>	60
<i>Отмена субботника</i>	62
<i>Изменение субботника</i>	62
РЕЛЯЦИОННАЯ МОДЕЛЬ	63
ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ МОДЕЛИ	63
ОПИСАНИЕ НАЗНАЧЕНИЙ КОЛЛЕКЦИЙ, ТИПОВ ДАННЫХ И СУЩНОСТЕЙ	64
Типы данных	64
Сущности и их свойства	64
User.....	64
User_Image:.....	65
Participation	65
Participation_Requirement.....	66
Requirement.....	66
CleanDay_Requirement	67
CleanDay	67
Location	68
Image	68
Location_Image	68
CleanDay_Image.....	69

Tag	69
CleanDay_Tag	69
City	69
Comment.....	70
CleanDay_Comment.....	70
Participation_Comment.....	70
Log.....	71
ОЦЕНКА ОБЪЕМА ИНФОРМАЦИИ, ХРАНИМОЙ В МОДЕЛИ.....	71
<i>Объём памяти для сохранения всех видов объектов.....</i>	71
<i>Зависимость общего объёма от количества объектов.....</i>	71
Избыточность данных	73
Чистый объем данных (Vч)	73
Фактический объем данных (V).....	73
Формула избыточности (E)	74
НАПРАВЛЕНИЕ РОСТА МОДЕЛИ	74
ПРИМЕРЫ ДАННЫХ	75
City	75
CleanDay	75
CleanDay_Comment.....	75
CleanDay_Image.....	75
CleanDay_Requirement	75
CleanDay_Tag	76
Comment.....	76
Image	76
Location	76
Location_Image	76
Log.....	76
Participation	77
Participation_Comment.....	77
Participation_Requirement.....	77
Requirement.....	77
Tag	77
User.....	78
User_Image	78
ПРИМЕРЫ ЗАПРОСОВ	78
<i>Регистрация нового пользователя</i>	78
<i>Авторизация пользователя</i>	79
<i>Просмотр информации о своём профиле</i>	79
<i>Обновление данных пользователя</i>	79
<i>Просмотр всех пользователей.....</i>	80
<i>Поиск пользователей по совпадению</i>	80
<i>Сложный поиск пользователей по атрибутам.....</i>	81
<i>Фильтрация пользователей по атрибуту</i>	81

<i>Просмотр информации о пользователе</i>	81
<i>Просмотр списка субботников, в которых участвовал пользователь</i>	81
<i>Просмотр списка субботников, которые организовал пользователь</i>	82
<i>Просмотр всех субботников</i>	82
<i>Поиск субботников по совпадению</i>	82
<i>Сложный поиск субботников по атрибутам</i>	83
<i>Фильтрация субботников с помощью чекбокса «Предстоящие субботники»</i>	83
<i>Фильтрация субботников с помощью чекбокса «С моим участием»</i>	83
<i>Просмотр информации о субботнике</i>	83
<i>Просмотр списка всех участников субботника</i>	84
<i>Просмотр списка участников субботника, подходящих под определенное условие</i>	84
<i>Просмотр списка участников субботника с определенным статусом участия</i>	84
<i>Просмотр истории активности участников субботника</i>	85
<i>Участие в субботнике</i>	85
<i>Изменить участие в субботнике</i>	85
<i>Добавление комментария</i>	86
<i>Просмотр всех организованных пользователем субботников</i>	86
<i>Организация нового субботника</i>	86
<i>Отмена субботника</i>	87
<i>Изменение субботника</i>	87
<i>Получение организаторов с самой плохой явкой</i>	87
СРАВНЕНИЕ МОДЕЛЕЙ	89
УДЕЛЬНЫЙ ОБЪЕМ ИНФОРМАЦИИ	89
ЗАПРОСЫ ПО ОТДЕЛЬНЫМ ЮЗКЕЙСАМ	89
Выход	90
РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ	92
КРАТКОЕ ОПИСАНИЕ.....	92
ИСПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ.....	92
СНИМКИ ЭКРАНА ПРИЛОЖЕНИЯ	93
ВЫВОДЫ	102
ДОСТИГНУТЫЕ РЕЗУЛЬТАТЫ.....	102
Недостатки и пути для улучшения полученного решения.....	103
Будущее развитие решения.....	104
ПРИЛОЖЕНИЯ	106
ДОКУМЕНТАЦИЯ ПО СБОРКЕ И РАЗВЕРТЫВАНИЮ ПРИЛОЖЕНИЯ	106
ИНСТРУКЦИЯ ДЛЯ ПОЛЬЗОВАТЕЛЯ	106
Регистрация и вход	106
Просмотр и участие в субботниках	106

<i>Создание собственного субботника</i>	107
<i>Управление профилем</i>	107
<i>Просмотр статистики</i>	108
<i>Завершение субботника (для организаторов)</i>	108
<i>Выход из системы</i>	108
ЛИТЕРАТУРА	109

ВВЕДЕНИЕ

Актуальность проблемы

В современном мире проблема загрязнения окружающей среды становится все более острой. Мусор на городских территориях, в парках, на берегах водоемов не только портит эстетический вид, но и негативно влияет на экологию и здоровье людей. Субботники - эффективный инструмент решения этой проблемы. Наше приложение позволит объединить людей, заинтересованных в улучшении экологической обстановки, а также сделает организацию субботников проще и эффективнее.

Постановка задачи

Создание веб-сервиса «CleanDay», который будет решать следующие задачи:

1. Организация субботников:

- Предоставление инструментов для создания мероприятий по уборке территории
- Возможность указания места, времени, необходимых требований к участникам
- Определение задач мероприятия с помощью тегов
- Отметка фактического присутствия заявленных участников
- Фотофиксация результатов субботника и некоторых числовых показателей

2. Участие в субботниках:

- Поиск подходящих субботников
- Регистрация на субботник с указанием статуса участия и соответствия требованиям
- Начисление очков опыта за участие в субботнике
- Возможность оставлять комментарии к субботнику

3. Функции для администрирования системы:
 - Массовый экспорт и импорт
 - Отображение логов о всех действиях в системе
 - Многокритериальный поиск и фильтрация по всем существующим сущностям в системе
4. Статистический учет:
 - Персональная статистика участия пользователей
 - Обобщенная статистика работы всего сервиса

Предлагаемое решение

Для реализации backend части веб-приложения используется Python с фреймворком FastAPI, для frontend части – TypeScript с фреймворком React. Для хранения данных применяется НСУБД ArangoDB, а для контейнеризации и развертывания – Docker.

Качественные требования к решению

Решение должно быть удобным, производительным, надежным и легко расширяемым, с высокой степенью взаимодействия между пользователями, удобным механизмом поиска и фильтрации, а также возможностью быстрого развертывания на различных платформах благодаря использованию Docker.

МАКЕТ UI

На рис. 1 представлен разработанный макет UI веб-приложения.

[Ссылка](#) на макет в Figma.



Рис. 1 – Макет UI

СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

Работа с пользователем-действующим лицом

Регистрация нового пользователя:

Действующее лицо: неавторизованный пользователь.

Предусловие:

- Пользователь не авторизован.
- Пользователь находится на главной странице сайта.

Основной сценарий:

1. Пользователь нажимает на ссылку «Зарегистрироваться».
2. Пользователь попадает на страницу с формой регистрации.

3. Пользователь заполняет все необходимые поля:
 - Имя - текстовое поле.
 - Фамилия - текстовое поле.
 - Логин - текстовое поле, значение должно быть уникальным.
 - Город - выбор из списка.
 - Пол - выбор радиокнопки.
 - Пароль - текстовое поле.
 - Повтор пароля - текстовое поле.
 4. Пользователь нажимает на кнопку «Зарегистрироваться».
- Альтернативный сценарий:
- Пользователь вводит уже существующий в системе логин.
 - Пользователь получает уведомление с просьбой придумать другой логин.
 - Пользователь хочет войти в уже существующий аккаунт, а не создать новый.
 - Пользователь нажимает ссылку «войти» на странице с формой регистрации.
 - Переход к п.2 сценария «Авторизация пользователя».
 - Отмена операции
 - Пользователь нажимает ссылку «Назад» на странице с формой регистрации.
 - Пользователь попадает на главную страницу.

Результат:

1. Пользователь входит в систему.
2. Пользователь попадает на главную страницу.

Авторизация пользователя

Действующее лицо: неавторизованный пользователь.

Предусловие:

- Пользователь не авторизован.
- Пользователь находится на главной странице сайта.

Основной сценарий:

1. Пользователь нажимает на ссылку «Войти».
2. Пользователь попадает на страницу с формой входа.
3. Пользователь заполняет все необходимые поля:
 - Логин - текстовое поле, значение должно быть уникальным.
 - Пароль - текстовое поле.
4. Пользователь может поставить галочку в чекбоксе «Запомнить меня».
5. Пользователь нажимает на кнопку «Войти».

Альтернативный сценарий:

- Пользователь вводит неверный логин или пароль.
 - Пользователь получает уведомление с просьбой придумать другой логин.
- Пользователь хочет создать новый аккаунт, а не войти в уже существующий.
 - Пользователь нажимает ссылку «зарегистрироваться» на странице с формой входа.
 - Переход к п.2 сценария «Регистрация нового пользователя»
- Отмена операции
 - Пользователь нажимает ссылку «Назад» на странице с формой входа.
 - Пользователь попадает на главную страницу.

Результат:

1. Пользователь входит в систему.
2. Пользователь попадает на главную страницу.

Просмотр информации о своем профиле

Действующее лицо: авторизованный пользователь.

Предусловие: пользователь авторизован.

Основной сценарий:

1. Пользователь нажимает на круглую иконку пользователя в правом верхнем углу.
2. Пользователь попадает на страницу с просмотром информации о пользователе.

Альтернативный сценарий:

- Пользователь не авторизован.
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Пользователь хочет изменить информацию о своем профиле
 - Переход к п.3 сценария «Изменение информации о своем профиле».
- Отмена операции
 - Пользователь нажимает на нужную ссылку в шапке сайта.
 - Пользователь попадает на нужную страницу.

Результат: пользователь попадает на страницу просмотра информации о своем профиле.

Изменение информации о своем профиле

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован.
2. Пользователь находится на странице просмотра информации о пользователе.
 - См. сценарий «Просмотр информации о своем профиле».

Основной сценарий:

1. Пользователь нажимает кнопку «Редактировать профиль».
2. Пользователь переходит на страницу редактирования профиля.
3. Пользователь редактирует все необходимые поля:

- Аватар - изображение.
 - Имя - текстовое поле.
 - Фамилия - текстовое поле.
 - Логин - текстовое поле, значение должно быть уникальным.
 - Пароль - текстовое поле.
 - Повтор пароля - текстовое поле.
 - Город - выбор из списка.
 - О себе - текстовое поле.
 - Пол - выбор радиокнопки.
4. Пользователь нажимает кнопку «Сохранить».

Альтернативный сценарий:

- Пользователь не авторизован.
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Пользователь вводит уже существующий в системе логин.
 - Пользователь получает уведомление с просьбой придумать другой логин.
- Отмена операции
 - Пользователь нажимает кнопку «Отмена».

Результат:

1. Пользователь изменяет информацию о своем профиле.
2. Пользователь попадает на страницу просмотра информации о своем профиле.

Выход из профиля

Действующее лицо: авторизованный пользователь.

Предусловие: пользователь авторизован.

Основной сценарий:

1. Пользователь нажимает на иконку выхода в правом верхнем углу.

2. Пользователь видит диалоговое окно с просьбой подтвердить операцию.
3. Пользователь нажимает кнопку «Выходи».

Альтернативный сценарий:

- Пользователь не авторизован.
 - Для неавторизованных пользователей кнопки с выходом из профиля не отображается.
- Отмена операции
 - Пользователь видит диалоговое окно с просьбой подтвердить операцию выхода из профиля.
 - Пользователь нажимает кнопку «Отмена».
 - Диалоговое окно исчезает.

Результат:

1. Диалоговое окно исчезает.
2. Пользователь выходит из профиля.
3. Пользователь попадает на главную страницу.

Работа со всеми пользователями

Просмотр всех пользователей

Действующее лицо: любой пользователь.

Предусловие: нет.

Основной сценарий:

1. Пользователь нажимает на ссылку «Пользователи» в шапке сайта или на главном экране.
2. Пользователь попадает на страницу списком всех пользователей сайта.
3. При необходимости, пользователь может настроить отображаемые столбцы с помощью специальной кнопки.

Альтернативный сценарий:

- Отмена операции
 - Пользователь нажимает на нужную ссылку в шапке сайта.

- Пользователь попадает на нужную страницу.

Результат: пользователь попадает на страницу списком всех пользователей сайта.

Поиск пользователей по совпадению

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех пользователей.

- См. сценарий «Просмотр всех пользователей».

Основной сценарий:

1. Пользователь вводит в поле поиска искомый текст.
2. Пользователь нажимает «Enter» или кнопку поиска.
3. Пользователь видит список пользователей, в которых есть совпадение с введенным текстом.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Отмена операции
 - Нажатие на крестик в поле поиска.

Результат: пользователь видит список пользователей, в которых есть совпадение с введенным текстом.

Сложный поиск пользователей по атрибутам

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех пользователей.

- См. сценарий «Просмотр всех пользователей».

Основной сценарий:

1. Пользователь нажимает значок фильтра возле поля поиска.
2. Открывается диалоговое окно со сложной фильтрацией.

3. Пользователь выбирает нужный атрибут:
 - Дата и время создания - дата.
 - Дата и время изменения - дата.
 - ID - числовой формат.
 - Имя - текстовый формат.
 - Фамилия - текстовый формат.
 - Пол - выбор из списка.
 - Логин - текстовый формат.
 - Город - текстовый формат.
 - О себе - текстовый формат.
 - Количество убранных квадратных километров - числовый формат.
 - Количество организованных субботников - числовый формат.
 - Количество субботников, в которых пользователь принял участие - числовый формат.
 4. Пользователь заполняет поле поиска значением атрибута в соответствующем формате.
 5. Пользователь нажимает кнопку «Найти».
 6. Для добавления дополнительных фильтров пункты 3-6 повторяются необходимое число раз.
 7. Пользователь нажимает кнопку «Найти».
 8. Пользователь видит список искомых пользователей.
- Альтернативный сценарий:
- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Результат: пользователь видит список искомых пользователей.
- Построение графика о пользователях**
- Действующее лицо: любой пользователь.
- Предусловие:

1. Пользователь находится на странице просмотра всех пользователей.

- См. сценарий «Просмотр всех пользователей».

Основной сценарий:

1. Пользователь нажимает на кнопку «Построить график».
2. Появляется диалоговое окно для построения графика.
3. Пользователь выбирает атрибуты по оси X и по оси Y.
4. Пользователь нажимает кнопку «Построить».
5. Пользователь видит соответствующий график.

Альтернативный сценарий:

- Невозможно построить график
 - Вывод сообщения об этом.
- Отмена операции
 - Пользователь нажимает на кнопку «Закрыть».
 - Диалоговое окно исчезает.

Результат: пользователь видит статистический график.

Фильтрация пользователей по атрибуту

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех пользователей.

- См. сценарий «Просмотр всех пользователей».

Основной сценарий:

1. Пользователь нажимает значок фильтра возле нужного атрибута.
2. Пользователь выбирает способ фильтрации: по возрастанию, по убыванию.
3. Пользователь видит список найденных пользователей.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.

- Отмена операции

- Нажатие кнопки «Сбросить фильтр».

Результат: пользователь видит список искомых пользователей.

Просмотр информации о пользователе

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех пользователей.

- См. сценарий «Просмотр всех пользователей».

Основной сценарий:

1. Пользователь нажимает на строчку с интересующим его пользователем в списке.

2. Пользователь попадает на страницу с просмотром пользователя.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Назад».

- Пользователь попадает на предыдущую страницу.

Результат: пользователь попадает на страницу просмотра пользователя.

Просмотр списка субботников, в которых участвовал пользователь

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра пользователя.

- См. сценарий «Просмотр информации о пользователе».

Основной сценарий:

1. Пользователь нажимает на кнопку «Участие».

2. Пользователь видит диалоговое окно со списком субботников, в которых участвовал пользователь.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список участников субботника, в которых участвовал пользователь.

Просмотр списка субботников, которые организовал пользователь

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра пользователя.

- См. сценарий «Просмотр информации о пользователе».

Основной сценарий:

1. Пользователь нажимает на кнопку «Организовано».
2. Пользователь видит диалоговое окно со списком субботников, которые организовал пользователь.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список участников субботника, которые организовал пользователь.

Работа со всеми субботниками

Просмотр всех субботников

Действующее лицо: любой пользователь.

Предусловие: нет.

Основной сценарий:

1. Пользователь нажимает на ссылку «Субботники» в шапке сайта или на главном экране.
2. Пользователь попадает на страницу списком всех субботников сайта.
3. При необходимости, пользователь может настроить отображаемые столбцы с помощью специальной кнопки.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на нужную ссылку в шапке сайта.
- Пользователь попадает на нужную страницу.

Результат: пользователь попадает на страницу списком всех субботников сайта.

Поиск субботников по совпадению

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь вводит в поле поиска искомый текст.
2. Пользователь нажимает «Enter» или кнопку поиска.
3. Пользователь видит список субботников, в которых есть совпадение с введенным текстом.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Отмена операции
 - Нажатие на крестик в поле поиска.

Результат: пользователь видит список субботников, в которых есть совпадение с введенным текстом.

Сложный поиск субботников по атрибутам

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь нажимает значок фильтра возле поля поиска.
2. Открывается диалоговое окно со сложной фильтрацией.
3. Пользователь выбирает нужный атрибут:
 - Дата и время создания - дата.
 - Дата и время изменения - дата.
 - ID - числовой формат.
 - Название - текстовый формат.
 - Город - текстовый формат.
 - Адрес - текстовый формат.
 - Дата и время начала - datetime формат.
 - Дата и время конца - datetime формат.
 - Организатор - текстовый формат.
 - Организация - текстовый формат.
 - Площадь территории - числовый формат.
 - Количество участников - числовый формат.
 - Описание - текстовый формат.
 - Статус - выбор из списка.
 - Тэги - выбор из списка.
 - Условия участия - текстовый формат.
4. Пользователь заполняет поле поиска значением атрибута в соответствующем формате.
5. Пользователь нажимает кнопку «Поиск».
6. Для добавления дополнительных фильтров пункты 3-6 повторяются необходимое число раз.
7. Пользователь нажимает кнопку «Найти».
8. Пользователь видит список искомых субботников.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.

Результат: пользователь видит список искомых субботников.

Построение графика о субботниках

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь нажимает на кнопку «Построить график».
2. Появляется диалоговое окно для построения графика.
3. Пользователь выбирает атрибуты по оси X и по оси Y.
4. Пользователь нажимает кнопку «Построить».
5. Пользователь видит соответствующий график.

Альтернативный сценарий:

- Невозможно построить график
 - Вывод сообщения об этом.
- Отмена операции
 - Пользователь нажимает на кнопку «Закрыть».
 - Диалоговое окно исчезает.

Результат: пользователь видит статистический график.

Фильтрация субботников по атрибуту

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь нажимает значок фильтра возле нужного атрибута.
2. Пользователь выбирает способ фильтрации: по возрастанию, по убыванию.

3. Пользователь видит список найденных субботников.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Отмена операции
 - Нажатие кнопки «Сбросить фильтр».

Результат: пользователь видит список искомых субботников.

Фильтрация субботников с помощью чекбокса «Предстоящие субботники»

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь ставит галочку в чекбоксе «Предстоящие субботники».
2. Пользователь видит список субботников, которые еще не состоялись.

Альтернативный сценарий:

- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Отмена операции
 - Нажатие на чекбокс для его очищения.

Результат: пользователь видит список субботников, которые еще не состоялись.

Фильтрация субботников с помощью чекбокса «С моим участием»

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован.
2. Пользователь находится на странице просмотра всех субботников.

- См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь ставит галочку в чекбоксе «С моим участием».
2. Пользователь видит список субботников, к которым он присоединился как участник.

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Поиск не нашел совпадений
 - Вывод сообщения об этом.
- Отмена операции
 - Нажатие на чекбокс для его очищения.

Результат: пользователь видит список субботников, к которым он присоединился как участник.

Просмотр информации о субботнике

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра всех субботников.
 - См. сценарий «Просмотр всех субботников».

Основной сценарий:

1. Пользователь нажимает на строчку с интересующим его субботником в списке.
2. Пользователь попадает на страницу с просмотром субботника.

Альтернативный сценарий:

- Пользователь хочет оставить комментарий
 - Переход к сценарию «Добавление комментария».
- Отмена операции

- Пользователь нажимает на кнопку «Назад».
- Пользователь попадает на предыдущую страницу.

Результат: пользователь попадает на страницу просмотра субботника.

Просмотр списка всех участников субботника

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра субботника.

- См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает на ссылку «Показать участников».
2. Пользователь видит диалоговое окно со списком участников субботника.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список участников субботника.

Просмотр списка участников субботника, подходящих под определенное условие

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра субботника.

- См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает на ссылку «Показать» возле соответствующего условия.

2. Пользователь видит диалоговое окно со списком участников субботника.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список участников субботника, подходящих под определенное условие.

Просмотр списка участников субботника с определенным статусом участия

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает на ссылку «Показать» возле соответствующего статуса.
2. Пользователь видит диалоговое окно со списком участников субботника.

Альтернативный сценарий:

- Отмена операции
 - Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список участников субботника с определенным статусом участия.

Просмотр истории активности участников субботника

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает на кнопку «Просмотр истории активности».
2. Пользователь видит диалоговое окно с хронологическим списком действий участников субботника.

Альтернативный сценарий:

- Отмена операции

- Пользователь нажимает на кнопку «Закрыть».

Результат: пользователь видит список действий участников субботника.

Участие в субботнике

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован
2. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает кнопку «Изменить участие в субботнике».
2. В открывшемся диалоговом окне пользователь выбирает условия организатора субботника, которым он соответствует.
3. Пользователь выбирает статус участия в субботнике: «точно пойду», «опоздаю», «возможно, пойду», «не пойду».

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Повторное нажатие на кнопку «Отменить участие»

Результат: пользователь регистрирует свое участие в субботнике.

Изменить участие в субботнике

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован
2. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает кнопку «Изменить участие в субботнике».
2. В открывшемся диалоговом окне пользователь добавляет или отменяет условия организатора субботника, которым он соответствует.
3. Пользователь выбирает новый статус участия в субботнике (текущий статус отмечен галочкой): «точно пойду», «опоздаю», «возможно, пойду», «не пойду».

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Нажатие на кнопку «Отмена» в диалоговом окне с подтверждением операции.

Результат: пользователь изменяет свое участие в субботнике.

Добавление комментария

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован
2. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь вводит комментарий в текстовом формате в соответствующее поле.
2. Пользователь нажимает иконку «Опубликовать».

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».

Результат: пользователь публикует комментарий.

Работа по организации субботников

Просмотр всех организованных пользователем субботников

Действующее лицо: авторизованный пользователь.

Предусловие: пользователь авторизован.

Основной сценарий:

1. Пользователь нажимает на ссылку «Организация субботников» в шапке сайта.
2. Пользователь попадает на страницу списком всех организованных пользователем субботников.
3. При необходимости, пользователь может настроить отображаемые столбцы с помощью специальной кнопки.

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Пользователь не организовал ни одного субботника
 - В списке всех организованных пользователем субботников не будет ни одной позиции.
- Отмена операции
 - Пользователь нажимает на нужную ссылку в шапке сайта.
 - Пользователь попадает на нужную страницу.

Результат: пользователь попадает на страницу списком всех организованных пользователем субботников.

Организация нового субботника

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован.

2. Пользователь находится на странице просмотра всех организованных пользователем субботников.

- См. сценарий «Просмотр всех организованных пользователем субботников».

Основной сценарий:

1. Пользователь нажимает на кнопку «Новый субботник».
2. Пользователь попадает на страницу создания нового субботника.
3. Пользователь заполняет все необходимые поля:
 - Название - текстовое поле.
 - Город - выбор из списка.
 - Адрес - текстовое поле.
 - Дата и время начала - формат datetime.
 - Дата и время конца - формат datetime.
 - Организация - необязательное текстовое поле.
 - Площадь в квадратных метрах - числовое поле.
 - Описание - текстовое поле.
 - Тэги - выбор из списка.
 - Рекомендуемое число участников - необязательное числовое поле.
 - Условия участия - необязательный список текстовых полей.
4. Пользователь нажимает на кнопку «Создать».

Альтернативный сценарий:

- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Пользователь не организовал ни одного субботника
 - В списке всех организованных пользователем субботников не будет ни одной позиции.
- Отмена операции
 - Пользователь нажимает на кнопку «Назад».

- Пользователь попадает на предыдущую страницу.

Результат:

1. Пользователь организует новый субботник.
2. Пользователь попадает на страницу списком всех организованных пользователем субботников.

Удаление субботника

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь является организатором субботника.
2. Пользователь авторизован.
3. Пользователь находится на странице просмотра субботника или просмотра списка организованных им субботников.
 - См. сценарий «Просмотр информации о субботнике» или сценарий «Просмотр всех организованных пользователем субботников».

Основной сценарий:

1. Пользователь нажимает на кнопку или иконку «Удалить».
2. Пользователь видит диалоговое окно с подтверждением операции.
3. Пользователь нажимает кнопку «Удалить».

Альтернативный сценарий:

- Пользователь не является организатором субботника
 - У пользователя нет кнопки и иконки «Удалить».
- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Пользователь нажимает кнопку «Отмена» в диалоговом окне с подтверждением операции.

Результат:

1. Пользователь удаляет субботник.
2. Пользователь попадает на страницу списком всех организованных пользователем субботников.

Отмена субботника

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь является организатором субботника.
2. Пользователь авторизован.
3. Пользователь находится на странице просмотра субботника.
 - См. сценарий «Просмотр информации о субботнике».

Основной сценарий:

1. Пользователь нажимает на кнопку «Отменить».

Альтернативный сценарий:

- Пользователь не является организатором субботника
 - У пользователя нет кнопки «Отменить».
- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Пользователь нажимает кнопку «Возобновить».

Результат: пользователь отменяет субботник.

Изменение субботника

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь является организатором субботника.
2. Пользователь авторизован.
3. Пользователь находится на странице просмотра субботника или просмотра списка организованных им субботников.

- См. сценарий «Просмотр информации о субботнике» или сценарий «Просмотр всех организованных пользователем субботников».

Основной сценарий:

1. Пользователь нажимает на кнопку или иконку «Изменить».
2. Пользователь попадает на страницу редактирования субботника.
3. Пользователь меняет все необходимые поля (они такие же, как при создании субботника).
4. Пользователь нажимает кнопку «Сохранить».

Альтернативный сценарий:

- Пользователь не является организатором субботника
 - У пользователя нет кнопки и иконки «Изменить».
- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Пользователь нажимает кнопку «Отмена».

Результат: пользователь изменяет информацию о субботнике.

Важное замечание!

Остальные сценарии этого раздела полностью аналогичны сценариям раздела «Работа со всеми субботниками» Различия:

1. Во всех сценариях действующим лицом является авторизованный пользователь.
2. Во все сценарии добавляется предпосылка «пользователь авторизован».
3. Во всех сценариях предпосылка «Пользователь находится на странице просмотра всех субботников» заменяется на:
 - Пользователь находится на странице просмотра всех организованных пользователем субботников.
 - См. сценарий «Просмотр всех организованных пользователем

субботников».

Информация о сервисе:

Просмотр статистики сервиса

Действующее лицо: любой пользователь.

Предусловие: нет.

Основной сценарий:

1. Пользователь нажимает на ссылку «Статистика» в шапке сайта или на главном экране.
2. Пользователь попадает на страницу со статистикой сервиса.

Альтернативный сценарий:

- Отмена операции
 - Пользователь нажимает на нужную ссылку в шапке сайта.
 - Пользователь попадает на нужную страницу.

Результат: пользователь попадает на страницу просмотра статистики сервиса.

Экспорт данных

Действующее лицо: любой пользователь.

Предусловие:

1. Пользователь находится на странице просмотра статистики сервиса.
 - См. сценарий «Просмотр статистики сервиса».

Основной сценарий:

1. Пользователь нажимает на кнопку «Экспорт».
2. Пользователь выбирает формат файла в диалоговом окне (CSV, JSON).
3. Пользователь нажимает на кнопку «Экспорт» для осуществления операции.

Альтернативный сценарий:

- Отмена операции
 - Пользователь нажимает кнопку «Отмена» в диалоговом окне.

Результат: пользователь загружает файл с данными в выбранном формате.

Импорт данных

Действующее лицо: авторизованный пользователь.

Предусловие:

1. Пользователь авторизован.
2. Пользователь находится на странице просмотра статистики сервиса.
 - См. сценарий «Просмотр статистики сервиса».

Основной сценарий:

1. Пользователь нажимает на кнопку «Импорт».
2. Пользователь выбирает файла для импорта в диалоговом окне.
3. Пользователь нажимает на кнопку «Импорт» для осуществления операции.

Альтернативный сценарий:

- Неподходящий формат файла
 - Пользователь не может загрузить файл формата, кроме CSV и JSON.
- Превышен максимально допустимый размер файла
 - Пользователь получает сообщение об этом
- Пользователь не авторизован
 - Пользователь переходит на страницу с формой входа.
 - Переход к п.2 сценария «Авторизация пользователя».
- Отмена операции
 - Пользователь нажимает кнопку «Отмена» в диалоговом окне.

Результат: пользователь загружает данные на сайт.

Вывод: операция чтения преобладают над операциями записи в системе.

НЕРЕЛЯЦИОННАЯ МОДЕЛЬ

Графическое представление модели

Сущность Log связана со всеми другими сущностями. Подробнее в коллекциях по связям.

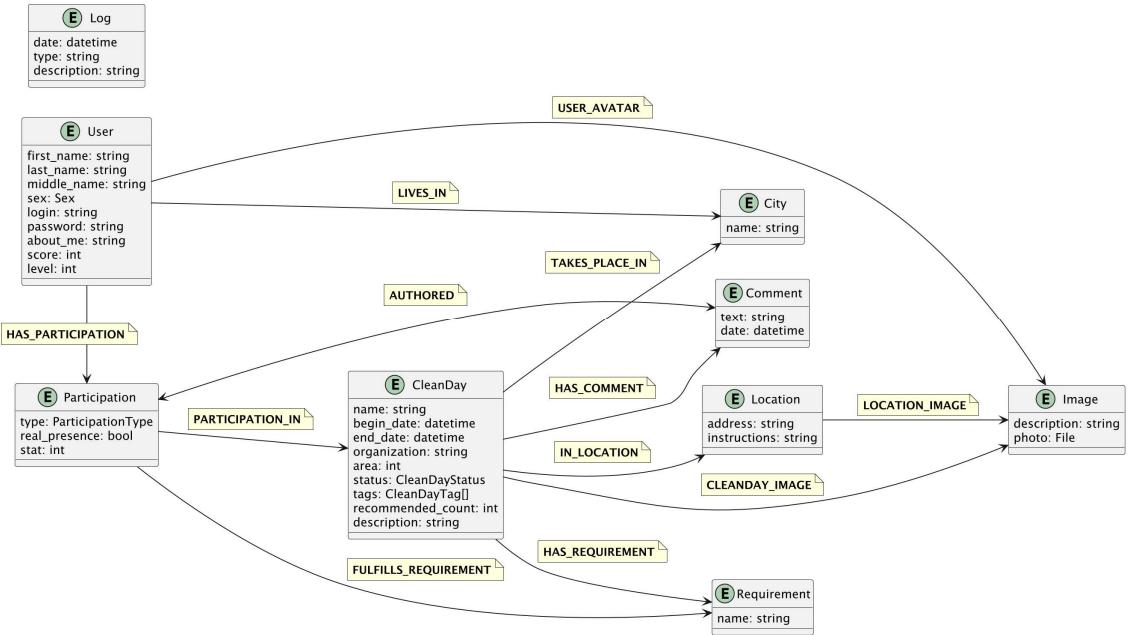


Рис. 2 – Графическое представление нереляционной модели

Описание назначений коллекций, типов данных и сущностей

Типы данных

Сущности используют следующие типы данных.

Базовые

- **string:** текст.
- **int:** целое число.
- **datetime:** дата и время.

Пользовательские

- **Sex** (6 байт) - пол:
 - «Male»
 - «Female»
 - «Other»
- **ParticipationType** (15 байт) - тип участия:
 - «Точно пойду»
 - «Опоздаю»
 - «Возможно, пойду»
 - «Не пойду»
 - «Организатор»
- **CleanDayStatus** (12 байт) - статус мероприятия:
 - «Запланирован»
 - «Проходит»
 - «Завершен»
 - «Отменен»
 - «Перенесен»
- **CleanDayTag** (18 байт) - статус мероприятия:
 - «Сбор мусора»
 - «Сортировка мусора»
 - «Посадка растений»
 - «Разбитие клумб»
 - «Разбитие газонов»
 - «Очистка водоемов»
 - «Уборка снега»
 - «Уборка листвьев»
 - «Уход за растениями»
 - «Ремонт»
 - «Покраска»

- «Установка кормушек»
- «Мастер-классы»
- «Игры и конкурсы»
- «Пикник»
- **File** (~1 MB) - закодированная строка, содержащая данные файла.

Сущности

User

Данные о пользователе, зарегистрированном в системе.

Поля (446 байт):

- **first_name**: string (~50 байт) - имя
- **last_name**: string (~50 байт) - фамилия
- **middle_name**: string (~50 байт) - отчество
- **sex**: Sex (6 байт) - пол
- **login**: string (~50 байт) - логин
- **password**: string (~32 байт) - пароль
- **score**: int (4 байта) - счет
- **level**: int (4 байта) - уровень
- **about_me**: string (~200 байт) - описание «о себе»

Participation

Тип участия в субботнике.

Поля (20 байт):

- **declared_type**: ParticipationType (15 байт) - тип участия, заявленный пользователем
- **real_presence**: bool (1 байт) - фактическое присутствие (1) или отсутствие (0) пользователя на субботнике
- **stat**: int (4 байта) - количество очков опыта за участие

Requirement

Требование для участия в субботнике.

Поля (100 байт):

- **name:** string (~100 байт) - описание требования

CleanDay

Данные о субботнике.

Поля (590 байт):

- **name:** string (~100 байт) - название субботника
- **begin_date:** datetime (8 байт) - дата и время начала субботника
- **end_date:** datetime (8 байт) - дата и время конца субботника
- **organization:** string (~100 байт) - название организации, проводящей субботник
- **area:** int (4 байт) - количество квадратных километров территории, на которой проводится субботник
- **status:** CleanDayStatus (12 байт) - текущий статус субботника
- **tags:** CleanDayTag[] (~3 * 18 байт) - тэги субботника
- **recommended_count:** int (4 байта) - рекомендованное количество участников в субботнике
- **description:** string (~300 байт) - описание субботника

City

Данные о городе, в котором проживает пользователь или проводится субботник.

Поля (100 байт):

- **name:** string (~100 байт) - название города

Comment

Данные о комментарии.

Поля (208 байт):

- **text:** string (~200 байт) - текст комментария
- **date:** datetime (8 байт) - дата и время отправки комментария

Location

Данные о локации субботника.

Поля (350 байт):

- **address:** string (~150 байт) - адрес локации
- **instructions:** string (~200 байт) - инструкции о том, как добраться до локации

Image

Данные изображения.

Поля (1048876 байт = 1.3 MB):

- **name:** string (~100 байт) - название изображения
- **description:** string (~200 байт) - описание изображения
- **photo:** file (~1 MB) - изображение

Log

Лог об изменениях в системе.

Поля (228 байт):

- **date:** datetime (8 байт) - дата и время изменения
- **type:** string (~20 байт) - тип изменения
- **description:** string (~200 байт) - описание изменения

Коллекции

Коллекции узлов:

- **User** - список данных о пользователях, зарегистрированных в системе
- **CleanDay** - список данных о субботниках

- **Comment** - список данных о комментариях
- **Log** - список изменений в системе
- **Requirement** - список требований для участия в субботнике
- **City** - список данных о городах
- **Participation** - список данных о типе участия в субботниках
- **Location** - список данных о локации субботников
- **Image** - список данных об изображениях

Коллекции ребер:

- **relates_to_user** - относится к пользователю ($\text{Log} \rightarrow \text{User}$)
- **relates_to_cleanday** - относится к субботнику ($\text{Log} \rightarrow \text{CleanDay}$)
- **relates_to_comment** - относится к комментарию ($\text{Log} \rightarrow \text{Comment}$)
- **relates_to_requirement** - относится к требованию ($\text{Log} \rightarrow \text{Requirement}$)
- **relates_to_city** - относится к городу ($\text{Log} \rightarrow \text{City}$)
- **relates_to_participation** - относится к участию ($\text{Log} \rightarrow \text{Participation}$)
- **relates_to_location** - относится к локации ($\text{Log} \rightarrow \text{Location}$)
- **relates_to_image** - относится к изображению ($\text{Log} \rightarrow \text{Image}$)
- **has_comment** - имеет комментарий ($\text{CleanDay} \rightarrow \text{Comment}$)
- **authored** - авторство ($\text{Comment} \leftrightarrow \text{Participation}$)
- **has_requirement** - имеет требование ($\text{CleanDay} \rightarrow \text{Requirement}$)
- **takes_place_in** - проходит ($\text{CleanDay} \rightarrow \text{City}$)
- **lives_in** - живет ($\text{User} \rightarrow \text{City}$)
- **has_participation** - принимает участие ($\text{User} \rightarrow \text{Participation}$)
- **participation_in** - участие ($\text{User} \rightarrow \text{CleanDay}$)
- **in_location** - в локации ($\text{CleanDay} \rightarrow \text{Location}$)
- **fulfills_requirement** - удовлетворяет требованию ($\text{Participation} \rightarrow \text{Requirement}$)
- **cleanday_image** - изображение субботника ($\text{CleanDay} \rightarrow \text{Image}$)

- **user_image** - изображение пользователя (User → Image)
- **location_image** - изображение локации (Location → Image)

Оценка объема информации, хранимой в модели

Описанные выше сущности хранятся как узлы в ArangoDB. К каждому узлу и ребру добавляются следующие служебные поля:

- key (~15 байт) - строка-уникальный идентификатор документа в пределах коллекции,
 - id (~25 байт) - строка-уникальный глобальный идентификатор документа,
 - rev (~15 байт) - строка-ревизия документа (для контроля версий)
- Кроме того, к ребрам добавляются еще два служебных поля:
- from (~25 байт) - идентификатор документа, из которого исходит ребро
 - to (~25 байт) - идентификатор документа, куда входит ребро

Получается, что дополнительно к каждому узлу-сущности добавляется около 55 байт служебной информации, а к каждому ребру-связи - примерно 105 байт служебной информации.

Объём памяти для хранения всех видов объектов

В табл. 1 перечислены объемы памяти, необходимые для хранения каждой сущности в ArangoDB и для хранения этой же сущности «в чистом виде».

Таблица 1 – Объемы данных каждого вида сущностей

Сущность	«Чистый» объем данных	Реальный объем данных
User	446 байт	$446 + 55 = 501$ байт
CleanDay	590 байт	$590 + 55 = 645$ байт
Location	350 байт	$350 + 55 = 405$ байт
Requirement	100 байт	$100 + 55 = 155$ байт
City	100 байт	$100 + 55 = 155$ байт
Participation	20 байт	$20 + 55 = 75$ байт
Comment	208 байт	$208 + 55 = 263$ байт
Image	1048876 байт	$1048876 + 55 = 1048931$ байт

Сущность	«Чистый» объем данных	Реальный объем данных
Log	228 байт	$28 + 55 = 283$ байт

Каждая связь несет в себе только служебную информацию, поэтому «чистый» объем данных любого вида связи = 0 байт, реальный объем данных любого вида связи = 105 байт.

Зависимость общего объема от количества объектов

В табл. 2 представлено примерное количество сущностей и связей в модели в зависимости от количества пользователей. Обозначим количество зарегистрированных пользователей сервиса за U , а количество субботников за C , причем $C = 1/5 U$.

Таблица 2 – Суммарный объем памяти, занимаемый всеми сущностями

Сущность	Количество	«Чистый» объем данных	Реальный объем данных
User	U	446 U байт	501 U байт
CleanDay	$C = 1/5 U$	118 U байт	129 U байт
Location	$C = 1/5 U$	70 U байт	81 U байт
Requirement	$3C = 3/5 U$	60 U байт	93 U байт
City	$U + C = 6/5 U$	120 U байт	186 U байт
Participation	$3U$	60 Uбайт	225 U байт
Comment	U	208 U байт	263 U байт
Image	$U + 4C = 9/5 U$	1887976,8 U байт	1888075,8 U байт
Log	$9U$ <i>(сумма всех количеств выше)</i>	2052 U байт	2547 U байт
Итого без изображений	16,2 U	3134 U байт	4025 U байт
Итого	18 U	1891110,8 U байт	1892100,8 U байт

Теперь проанализируем количество связей в зависимости от количества пользователей U:

- **relates_to_user** - U
- **relates_to_cleanday** - C = 1/5 U
- **relates_to_comment** - U
- **relates_to_requirement** - 3C = 3/5 U
- **relates_to_city** - U + C = 6/5 U
- **relates_to_participation** - 3U
- **relates_to_location** - C = 1/5 U
- **relates_to_image** - U + 4*C = 9/5 U
- **has_comment** - U
- **authored** - U
- **has_requirement** - 3C = 3/5 U
- **takes_place_in** - C = 1/5 U
- **lives_in** - U
- **has_participation** - 3U
- **participation_in** - 3U
- **in_location** - C = 1/5 U
- **fulfills_requirement** - 3U * 2/3 = 2U
- **cleanday_image** - 2C = 2/5 U
- **user_image** - U
- **location_image** - 2C = 2/5 U

Количество всех связей без учета изображений: 21 U
Количество всех связей: 22,8 U

Суммарный объем всех связей без учета изображений: 21 U * 105 байт = 2205 U байт

Суммарный объем всех связей: 22,8 U * 105 байт = 2394 U байт

Объем всех «чистых» данных без учета изображений = 3134 U байт

Суммарный объем всех «чистых» данных = 1891110,8 U байт
Фактический суммарный объем всех данных без учета изображений = 4025 U +
2205 U = 6230 U байт

Фактический суммарный объем всех данных, хранимых в ArangoDB = 1892100,8 U + 2394 U = 1894494,8 U байт

Чистый объем данных $V_{\text{ч}}(U)$ = 1891110,8 U байт
Фактический объем данных $V(U)$ = 1894494,8 U байт,
где U - количество зарегистрированных в системе пользователей

Избыточность данных

$$\text{Вычислим избыточность данных: } E(U) = \frac{V(U)}{V_{\text{ч}}(U)} = \frac{1894494,8*U}{1891110,8*U} = 1,0018$$

Тот же расчет, но без учета изображений, которые занимают большую часть памяти: $E(U) = \frac{V(U)}{V_{\text{ч}}(U)} = \frac{6230*U}{3134*U} = 1,988$

Второй расчет более информативен, поскольку объем памяти, занимаемых изображениями, в разы больше объема памяти, занимаемого остальными данными. Кроме того, объем памяти, занимаемых изображениями, одинаков и в «чистых» данных, и в фактических.

Таким образом, избыточность модели без учета изображений существенна и составляет 98,8%.

Наличие изображений сводит избыточность модели к минимуму - 0,18%.

Направление роста модели

Нетрудно догадаться, что наибольшее влияние на рост модели оказывают изображения: их и много (4U), и объем памяти для одного изображения многократно превышает объем памяти, необходимый для любой другой сущности.

На втором месте по влиянию на рост модели находятся логи по тем же причинам: большое количество логов (9 U) + значительный «вес» одного лога.

Наименьшее влияние на модель оказывают требования для участия в субботнике - их немного, и они состоят всего из одного поля, занимающего

незначительный объем памяти.

Примеры данных

На рис. 3 представлены примеры данных.

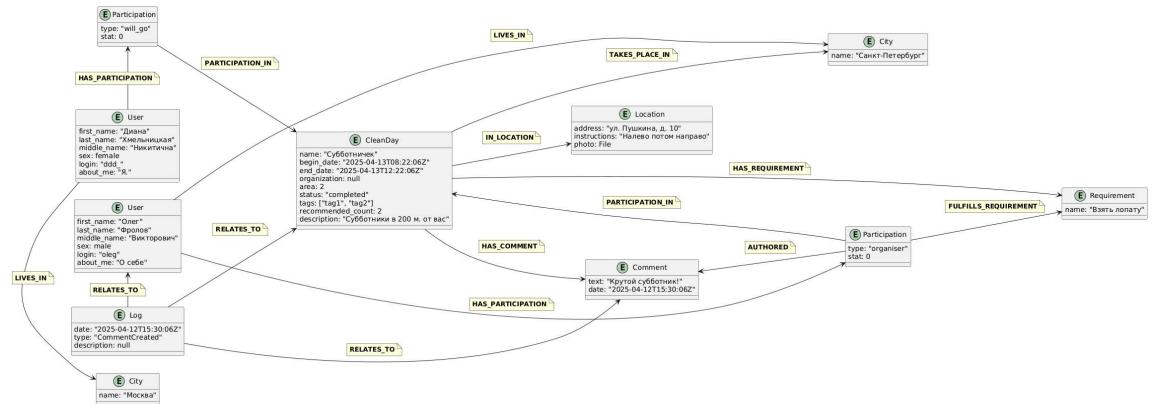


Рис. 3 – Примеры данных

Примеры запросов

Регистрация нового пользователя

Получение пользователя по логину

```
FOR u IN User
  FILTER u.login == @login
  RETURN u
```

Создание пользователя

```
INSERT {
  first_name: «Иван»,
  last_name: «Иванов»,
  email: «jane@example.com»,
  sex: «male»,
  login: «ivan»,
  password: «hash»,
  about_me: «Обо мне»
} INTO User
RETURN NEW
```

Получение городов по имени

```
FOR c IN City
  FILTER LIKE(c.name, CONCAT(«%», @name, «%»), true)
  RETURN c
{
  «name»: «Город»
}
```

Добавление города к пользователю

```
INSERT {
  _from: CONCAT(«User/», @user_id),
  _to: CONCAT(«City/», @city_id)
} INTO lives_in
```

```
{  
  «user_id»: 42,  
  «city_id»: 43  
}
```

Добавление аватара пользователя

```
LET img = (  
  INSERT {  
    description: «avatar»,  
    photo: @file  
  }  
  RETURN NEW  
)  
  
INSERT {  
  _from: CONCAT(«User/», @user_id),  
  _to: img._id  
} INTO user_avatar  
{  
  «user_id»: 42,  
  «file»: «file_bytes»  
}
```

- Запросов: **5** (Больше если потребуется больше запросов по городам)
- Задействованные коллекции: User, City, Photo

Авторизация пользователя

См выше: - Получение пользователя по логину

- Запросов: **1**
- Задействованные коллекции: User

Просмотр информации о своём профиле

Получение пользователя по id

```
FOR u IN User  
  FILTER u.key == @id  
  RETURN u  
{  
  «id»: 42  
}
```

Получение города пользователя по его id

```
LET userId = CONCAT(«User/», @id)  
  
FOR city IN OUTBOUND userId lives_in  
  LIMIT 1  
  RETURN city  
{  
  «id»: 42  
}
```

- Запросов: 2
- Задействованные коллекции: User, City

Обновление данных пользователя

См выше: - Получение городов по имени - Добавление города к пользователю

Изменение данных пользователя по его id

```
UPDATE @id WITH @changes IN User
{
  «id»: 42,
  «changes»: {
    «first_name»: «Иван»,
    «last_name»: «Иванов»,
    «email»: «jane@example.com»,
    «sex»: «male»,
    «login»: «ivan»,
    «password»: «hash»,
    «about_me»: «Обо мне»
  }
}
```

Удаление связи города с пользователем по id

```
LET userId = CONCAT(«User/», @user_id)

FOR edge IN lives_in
  FILTER edge._from == userId
  REMOVE edge IN lives_in
{
  «user_id»: 42
}
```

Обновление аватара пользователя

```
LET userId = CONCAT(«User/», @user_id)

FOR file IN 1..1 OUTBOUND userId user_avatar
  UPDATE file WITH {
    photo: @file
  } IN File
  RETURN NEW
{
  «user_id»: 42,
  «file»: «file_bytes»
}
```

- Запросов: 1-3 (1 для стандартных атрибутов, 2 для города, 1 для аватара)
- Задействованные коллекции: User, City

Просмотр всех пользователей

См. выше: - Получение города пользователя по его id

Получение пользователей с пагинацией

```
FOR u IN User
    LIMIT @offset, @count
    RETURN u
{
    «offset»: 0,
    «count»: 10
}
```

Прим. Здесь и далее запрос на получение города объединяется с запросом на получение пользователей.

- Запросов: 1
- Задействованные коллекции: User, City

Поиск пользователей по совпадению

Получение пользователей с фильтрацией по атрибутам

```
LET filterObject = @filter_params

FOR u IN User
    FILTER
        (LENGTH(FOR key IN ATTRIBUTES(filterObject) FILTER LIKE(u[key], CONCAT(«%», filterObject[key], «%»)), true) == LENGTH(ATTRIBUTES(filterObject)))
    LIMIT @offset, @limit
    RETURN u
{
    «filter_params»: {
        «key»: «value»
    },
    «offset»: 0,
    «limit»: 10
}
```

- Запросов: 1
- Задействованные коллекции: User, City

Сложный поиск пользователей по атрибутам

См. выше: - Получение пользователей с фильтрацией по атрибутам

- Запросов: 1
- Задействованные коллекции: User, City

Фильтрация пользователей по атрибуту

Получение пользователей с сортировкой

```
FOR u IN User
    SORT u.first_name ASC
    LIMIT @offset, @count
    RETURN u
{
    «offset»: 0,
    «count»: 10
}
```

Прим. Другие условия сортировки строятся похожим образом.

- Запросов: **1**
- Задействованные коллекции: User, City

Просмотр информации о пользователе

См выше: - Получение пользователя по id - Получение города пользователя по его id

- Запросов: **2**
- Задействованные коллекции: User, City

Просмотр списка субботников, в которых участвовал пользователь

Получение субботников пользователя

```
LET userId = CONCAT(«User/», @id)
```

```
FOR p IN OUTBOUND userId has_participation
    FOR cl_day IN OUTBOUND p participation_in
        RETURN cl_day
```

- Запросов: **1**
- Задействованные коллекции: User, Participation, CleanDay

Просмотр списка субботников, которые организовал пользователь

Получение организованных пользователем субботников

```
LET userId = CONCAT(«User/», @id)
```

```
FOR p IN OUTBOUND userId has_participation
    FILTER p.type == «organiser»
    FOR cl_day IN OUTBOUND p participation_in
        RETURN cl_day
```

- Запросов: **1**

- Задействованные коллекции: User, Participation, CleanDay

Просмотр всех субботников

Получение субботников с пагинацией

```
FOR cd IN CleanDay
    LIMIT @offset, @count
    RETURN cd
{
    «offset»: 0,
    «count»: 10
}
```

Получение города, в котором проходит субботник, по его id

```
LET cdId = CONCAT(«CleanDay», @id)

FOR city IN OUTBOUND cdId takes_place_in
    LIMIT 1
    RETURN city
{
    «id»: 42
}
```

Прим. Здесь и далее запрос на получение города объединяется с запросом на получение субботника.

- Запросов: 1
- Задействованные коллекции: CleanDay, City

Поиск субботников по совпадению

Получение субботников с фильтрацией по атрибутам

```
LET filterObject = @filter_params
```

```
FOR cd IN CleanDay
    FILTER
        (LENGTH(FOR key IN ATTRIBUTES(filterObject) FILTER LIKE(cd[key], CONCAT(«%», filterObject[key], «%»), true) RETURN 1) == LENGTH(ATTRIBUTES(filterObject)))
    LIMIT @offset, @limit
    RETURN u
{
    «filter_params»: {
        «key»: «value»
    },
    «offset»: 0,
    «limit»: 10
}
```

- Запросов: 1
- Задействованные коллекции: CleanDay, City

Сложный поиск субботников по атрибутам

См. выше: - Получение субботников с фильтрацией по атрибутам

Прим. В данном случае условия фильтрации будут динамичными.

- Запросов: **1**
- Задействованные коллекции: CleanDay, City

Фильтрация субботников с помощью чекбокса «Предстоящие субботники»

Аналогично предыдущему, отличие в условии фильтрации.

- Запросов: **1**
- Задействованные коллекции: CleanDay, City

Фильтрация субботников с помощью чекбокса «С моим участием»

См. выше: - Получение субботников пользователя

- Запросов: **1**
- Задействованные коллекции: CleanDay, User

Просмотр информации о субботнике

См выше: - Получение города, в котором проходит субботник по его id

Получение субботника по id

```
FOR cd IN CleanDay
  FILTER cd.key == @id
  RETURN cd
{
  «id»: 42
}
```

Получение локации субботника по его id

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```
FOR loc IN OUTBOUND cdId at_location
  LIMIT 1
  RETURN loc
{
  «id»: 42
}
```

Получение фото субботника по его id

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```

FOR file IN 1..1 OUTBOUND cdId cleanday_image
    RETURN file
{
    «id»: 42
}

```

- Запросов: 3
- Задействованные коллекции: CleanDay, City, Location

Просмотр списка всех участников субботника

Получение всех участников субботника

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```

FOR par IN INBOUND cdId participation_in
    FOR user IN INBOUND par has_participation
        LIMIT 1
        RETURN user
{
    «id»: 42
}

```

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User

Просмотр списка участников субботника, подходящих под определенное условие

Получение всех участников субботника с условием

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```

FOR par IN INBOUND cdId participation_in
    FOR user IN INBOUND par has_participation
        LIMIT 1
        FILTER user.first_name == «Олег»
        RETURN user
{
    «id»: 42
}

```

Получение участников, не пришедших на субботник

```

FOR u IN User
    FOR p IN OUTBOUND u has_participation
        FILTER p.type == «Точно пойду»
            AND p.real_presence == false
        RETURN DISTINCT u

```

Прим. Условие фильтрации в реальных запросах может варьироваться

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User

Просмотр списка участников субботника с определенным статусом участия

Получение всех участников субботника с определённым статусом участия

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```
FOR par IN INBOUND cdId participation_in
  FILTER par.type == @state
  FOR user IN INBOUND par has_participation
    LIMIT 1
    RETURN user
{
  «id»: 42,
  «state»: «will_go»
}
```

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User

Просмотр истории активности участников субботника

Получение логов активности субботника

```
LET cdId = CONCAT(«CleanDay/», @id)
```

```
FOR logEntry IN INBOUND cdId relates_to_cleanday

  LET user = FIRST(FOR user IN OUTBOUND logEntry relates_to_user
    LIMIT 1
    RETURN user)
  LET comment = FIRST(FOR comment IN OUTBOUND logEntry relates_to_comment
    LIMIT 1
    RETURN comment)

  RETURN {
    entry: logEntry,
    user: user,
    comment: comment
  }
{
  «id»: 42
}
```

- Запросов: 1
- Задействованные коллекции: CleanDay, Log, User, Comment

Участие в субботнике

Добавление участия пользователя с типом

```
LET userId = CONCAT(«User/», @user_id)
LET cleanDayId = CONCAT(«CleanDay/», @cleanday_id)

LET par = FIRST(
    INSERT {
        type: @participation_type,
        stat: 0
    } INTO Participation
    RETURN NEW
)

INSERT {
    _from: userId,
    _to: par._id
} INTO has_participation

INSERT {
    _from: par._id,
    _to: cleanDayId
} INTO participation_in
{
    «user_id»: 42,
    «cleanday_id»: 43,
    «participation_type»: «will_go»
}
```

- Запросов: 3
- Задействованные коллекции: CleanDay, Participation, User

Изменить участие в субботнике

Обновление участия пользователя

```
LET userId = CONCAT(«User/», @user_id)
LET cleanDayId = CONCAT(«CleanDay/», @cleanday_id)

FOR par IN OUTBOUND userId has_participation
    FOR cd IN OUTBOUND par participation_in
        FILTER cd._id == cleanDayId

        UPDATE par WITH {
            type: @participation_type
        } IN Participation
    {
        «user_id»: 42,
        «cleanday_id»: 43,
        «participation_type»: «will_go»
    }
```

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User

Добавление комментария

Добавления комментария пользователя в субботнике

```
LET userId = CONCAT(«User/», @user_id)
LET cleanDayId = CONCAT(«CleanDay/», @cleanday_id)

FOR par IN OUTBOUND userId has_participation
    FOR cd IN OUTBOUND par participation_in
        FILTER cd._id == cleanDayId

        LET comm = FIRST(
            INSERT {
                text: @text,
                time: @time
            } INTO Comment
            RETURN NEW
        )

        INSERT {
            _from: par._id,
            _to: comm._id
        } INTO authored

        INSERT {
            _from: cd._id,
            _to: comm._id
        } INTO has_comment
```

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User, Comment

Просмотр всех организованных пользователем субботников

См. выше: - [Получение организованных пользователем субботников](#)

- Запросов: 1
- Задействованные коллекции: CleanDay, Participation, User

Организация нового субботника

См. выше: - Получение городов по имени ##### Создание нового субботника

```
LET locationId = CONCAT(«Location/», @location_id)
LET cityId = CONCAT(«City/», @city_id)
LET userId = CONCAT(«User/», @user_id)

LET cleanDay = FIRST(
    INSERT {
        name: @name,
        begin_date: @begin_date,
        end_date: @end_date,
```

```

organization: @organization,
area: @area,
status: @status,
tags: @tags,
recommended_count: @recommended_count,
description: @description
} INTO CleanDay
RETURN NEW
)

INSERT {
    _from: cleanDay._id,
    _to: locationId
} INTO at_location

INSERT {
    _from: cleanDay._id,
    _to: cityId
} INTO takes_place_in

LET participation = FIRST(
    INSERT {
        type: «organiser»,
        stat: 0
    } INTO Participation
    RETURN NEW
)

INSERT {
    _from: userId,
    _to: participation._id
} INTO has_participation

INSERT {
    _from: participation._id,
    _to: cleanDay._id
} INTO participation_in

RETURN {
    cleanDay,
    participation
}
{
    «location_id»: 42,
    «city_id»: 43,
    «user_id»: 44,
    «name»: «name»,
    «begin_date»: «timestamp»,
    «end_date»: «timestamp»,
    «organization»: null,
    «area»: 200,
    «status»: «planned»,
    «tags»: [
        «tag1»
    ],
    «recommended_count»: 20,
}

```

```

    «description»: «super duper»
}

Добавление фото субботника

LET cdId = CONCAT(«CleanDay/», @id)

LET img = FIRST(
  INSERT {
    description: @description,
    photo: @file
  } INTO File
  RETURN NEW
)

INSERT {
  _from: cdId,
  _to: img._id
} INTO cleanday_image

RETURN img
{
  «id»: 42,
  «file»: «file_buffer»,
  «description»: «file_buffer»
}

```

- Запросов: 7
- Задействованные коллекции: CleanDay, Participation, User, City, Location, Photo

Отмена субботника

Изменение параметров субботника

```

UPDATE @id WITH @changes IN CleanDay
{
  «id»: 42,
  «changes»: {
    «status»: «canceled»
  }
}

```

- Запросов: 1
- Задействованные коллекции: CleanDay

Изменение субботника

См. выше: - Изменение параметров субботника - Добавление фото субботника

- Запросов: 2

- Задействованные коллекции: CleanDay

Получение организаторов с самой плохой явкой

```

FOR u IN User
    LET organizedCDs = (
        FOR p IN OUTBOUND u has_participation
            FILTER p.type == «Организатор»
            FOR cd IN OUTBOUND p participation_in
                RETURN DISTINCT cd
    )
    FILTER LENGTH(organizedCDs) > 0

    LET stats = (
        FOR cd IN organizedCDs
            LET totalWillGo = LENGTH(
                FOR pr IN INBOUND cd._id participation_in
                    FILTER pr.type == «Точно пойду»
                    RETURN 1
            )
            LET missed = LENGTH(
                FOR pr IN INBOUND cd._id participation_in
                    FILTER pr.type == «Точно пойду»
                    AND pr.real_presence == false
                    RETURN 1
            )
            FILTER totalWillGo > 0
            RETURN {
                cleandayId: cd._id,
                totalWillGo,
                missed,
                ratio: missed / totalWillGo
            }
    )

    LET avgNoShowRatio = AVERAGE(stats[*].ratio)

    SORT avgNoShowRatio DESC

    RETURN {
        organizer: u,
        organizedCount: LENGTH(organizedCDs),
        stats,
        avgNoShowRatio
    }

```

- Запросов: **1**
- Задействованные коллекции: User, Participation, CleanDay

РЕЛЯЦИОННАЯ МОДЕЛЬ

Графическое представление модели

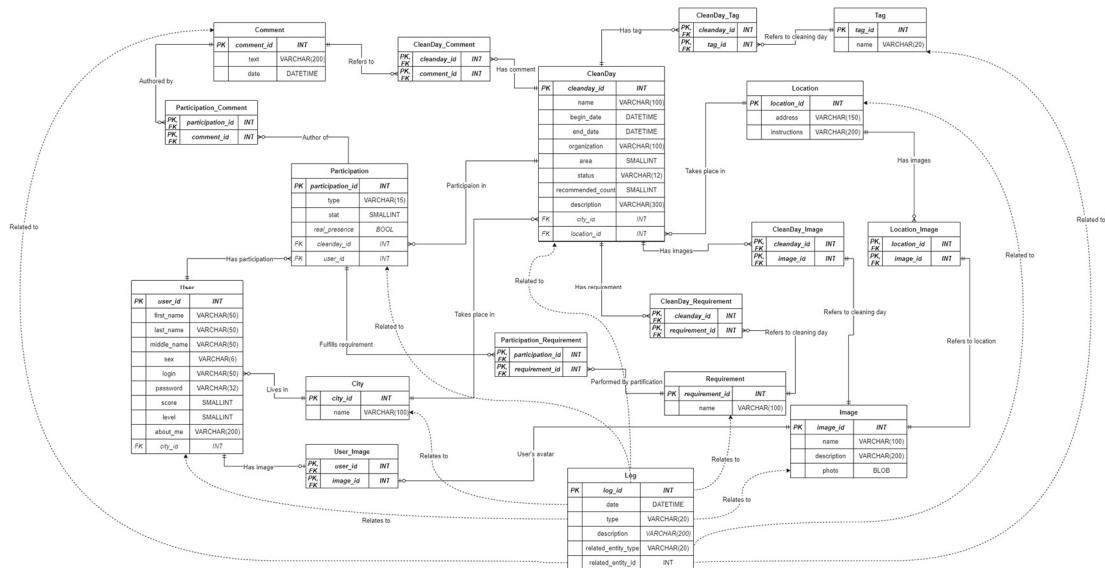


Рис. 5 – Графическое представление реляционной модели данных
Описание назначений коллекций, типов данных и сущностей

Типы данных

- INT: целое число
- SMALLINT: целое небольшое число
- VARCHAR(N): строка символов длины N
- DATETIME: дата и время в формате YYYY-MM-DD HH:MI:SS
- BLOB: двоичная строка для хранения файлов

Сущности и их свойства

User

Сущность пользователя, зарегистрированного в системе

Поля (*в среднем 464 байт*):

- **user_id**: Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **first_name**: VARCHAR(50) (*~50 + 2 байт*) - имя
- **last_name**: VARCHAR(50) (*~50 + 2 байт*) - фамилия
- **middle_name**: VARCHAR(50) (*~50 + 2 байт*) - отчество
- **sex**: VARCHAR(6) (*~6 + 2 байт*) - пол. Может принимать значения:

- «Male»
 - «Female»
 - «Other»
- **login:** VARCHAR(50), Unique ($\sim 50 + 2$ байт) - логин
- **password:** VARCHAR(64) ($\sim 64 + 2$ байт) - захэшированный пароль
- **about_me:** VARCHAR(200) ($\sim 200 + 2$ байт) - описание «о себе»
- **score:** SMALLINT (2 байта) - счёт
- **level:** SMALLINT (2 байта) - уровень
- **city_id:** Foreign Key referencing CITY(city_id) (4 байта) - идентификатор города, в котором живёт пользователь

User_Image:

Промежуточная сущность, обозначающая принадлежность определённого изображения определённому пользователю

Поля (8 байт):

- **user_id:** Foreign Key referencing USER(user_id), Part of Primary Key (4 байт) - идентификатор пользователя, имеющего фото
- **image_id:** Foreign Key referencing IMAGE(image_id), Part of Primary Key (4 байт) - идентификатор фото, которое принадлежит пользователю

Participation

Сущность, характеризующая участие пользователя в субботнике

Поля (*в среднем 31 байт*):

- **participation_id:** Primary Key, INT, Auto-increment (4 байт) - уникальный идентификатор
- **declared_type:** VARCHAR(15) ($\sim 15 + 2$ байт) - тип/степень участия в субботнике. Может принимать значения:
 - «Точно пойду»
 - «Опоздаю»

- «Возможно, пойду»
 - «Не пойду»
 - «Организатор»
- **real_presence:** BOOLEAN (*1 байт*) - фактическое присутствие (1) или отсутствие (0) пользователя на субботнике
- **stat:** SMALLINT (*2 байта*) - количество начисляемых пользователю за участие очков
- **user_id:** Foreign Key referencing USER(user_id) (*4 байта*) - идентификатор пользователя, который участвует в субботнике
- **cleanday_id:** Foreign Key referencing CLEANDAY(cleanday_id) (*4 байта*) - идентификатор субботника, в котором участвует пользователь

Participation_Requirement

Промежуточная сущность, связывающая участие пользователя в субботнике и требование, которое выполняется участием

Поля (*8 байт*):

- **participation_id:** Foreign Key referencing PARTICIPATION(participation_id), Part of Primary Key (*4 байта*) - идентификатор участия
- **requirement_id:** Foreign Key referencing REQUIREMENT(requirement_id), Part of Primary Key (*4 байта*) - идентификатор выполняемого требования

Requirement

Сущность требования для участия в субботнике

Поля (*в среднем 106 байт*):

- **requirement_id:** Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **name:** VARCHAR(100) (*~100 + 2 байт*) - описание требования

CleanDay_Requirement

Промежуточная сущность, связывающая субботник и необходимые для участия требования

Поля (*8 байт*):

- **cleanday_id**: Foreign Key referencing CLEANDAY(cleanday_id), Part of Primary Key (*4 байта*) - идентификатор субботника
- **requirement_id**: Foreign Key referencing REQUIREMENT(requirement_id), Part of Primary Key (*4 байта*) - идентификатор требования для субботника

CleanDay

Сущность организованного субботника

Поля (*в среднем 552 байта*):

- **cleanday_id**: Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **name**: VARCHAR(100) (~100 + 2 байта) - название
- **begin_date**: DATETIME (*8 байт*) - дата и время начала
- **end_date**: DATETIME (*8 байт*) - дата и время окончания
- **organization**: VARCHAR(100) (~100 + 2 байта) - название проводящей организации
- **area**: SMALLINT (*2 байта*) - площадь убираемой территории в квадратных метрах
- **status**: VARCHAR(12) (~12 + 2 байт) - текущий статус субботника. Может принимать значения:
 - «Запланирован»
 - «Проходит»
 - «Завершен»
 - «Отменен»
 - «Перенесен»
- **recommended_count**: SMALLINT (*2 байта*) - рекомендованное число

участников

- **description:** VARCHAR(300) ($\sim 300 + 2$ байта) - описание
- **city_id:** Foreign Key referencing CITY(city_id) (4 байта) - идентификатор города, в котором проводится субботник
- **location_id:** Foreign Key referencing LOCATION(location_id) (4 байта) - идентификатор локации, где проводится субботник

Location

Сущность локации субботника

Поля (в среднем 358 байт):

- **location_id:** Primary Key, INT, Auto-increment () - уникальный идентификатор
- **address:** VARCHAR(150) ($\sim 150 + 2$ байт) - адрес
- **instructions:** VARCHAR(200) ($\sim 200 + 2$ байт) - описание, как добраться

Image

Сущность изображения, загруженного пользователем

Поля (в среднем 1048882 байт $\sim 1024,3$ КБ ~ 1 МБ):

- **image_id:** Primary Key, INT, Auto-increment (4 байта) - уникальный идентификатор
- **name:** VARCHAR(100) ($\sim 100 + 2$ байта) - название
- **description:** VARCHAR(200) ($\sim 200 + 2$ байта) - описание
- **photo:** BLOB (~ 1 МБ) - само изображение в формате двоичной строки

Location_Image

Промежуточная сущность для связи локации и изображений

Поля (8 байт):

- **location_id:** Foreign Key referencing LOCATION(location_id), Part of Primary Key (4 байта) - идентификатор локации, которой принадлежит фото
- **image_id:** Foreign Key referencing IMAGE(image_id), Part of Primary Key (4 байта) - идентификатор изображения, которое принадлежит локации

байта) - идентификатор изображения, которое принадлежит локации

CleanDay_Image

Промежуточная сущность для связи субботника и изображений

Поля (*8 байт*):

- **cleanday_id:** Foreign Key referencing CLEANDAY(cleanday_id), Part of Primary Key (*4 байта*) - идентификатор субботника, которому принадлежит фото
- **image_id:** Foreign Key referencing IMAGE(image_id), Part of Primary Key (*4 байта*) - идентификатор изображения, которое принадлежит субботнику

Tag

Сущность тега на субботнике

Поля (*в среднем 26 байт*):

- **tag_id:** Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **name:** VARCHAR(20), Unique (*~20 + 2 байта*) - содержимая подпись тега

CleanDay_Tag

Промежуточная сущность для связи субботника и тегов

Поля (*8 байт*):

- **cleanday_id:** Foreign Key referencing CLEANDAY(cleanday_id), Part of Primary Key (*4 байта*) - идентификатор субботника, которому принадлежит тег
- **tag_id:** Foreign Key referencing TAG(tag_id), Part of Primary Key (*4 байта*) - идентификатор тега, который принадлежит субботнику

City

Сущность города, в котором проходит город или проводится субботник

Поля (*в среднем 106 байт*):

- **city_id:** Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **name:** VARCHAR(100), Unique (*~100 + 2 байта*) - название

Comment

Сущность комментария под субботником

Поля (*в среднем 214 байт*):

- **comment_id:** Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **text:** VARCHAR(200) (*~200 + 2 байта*) - содержимый текст
- **date:** DATETIME (*8 байт*) - дата и время создания

CleanDay_Comment

Промежуточная сущность для связи субботника и комментария

Поля (*8 байт*):

- **cleanday_id:** Foreign Key referencing CLEAN_DAY(clean_day_id), Part of Primary Key (*4 байта*) - идентификатор комментируемого субботника
- **tag_id:** Foreign Key referencing TAG(tag_id), Part of Primary Key (*4 байта*) - идентификатор комментария под субботником

Participation_Comment

Промежуточная сущность для связи участия и комментария

Поля (*8 байт*):

- **participation_id:** Foreign Key referencing PARTICIPATION(participation_id), Part of Primary Key (*4 байта*) - идентификатор участия, которое является автором комментария
- **comment_id:** Foreign Key referencing COMMENT(comment_id), Part of Primary Key (*4 байта*) - идентификатор созданного комментария

Log

Сущность лога изменений в системе

Поля (*в среднем 262 байт*):

- **log_id:** Primary Key, INT, Auto-increment (*4 байта*) - уникальный идентификатор
- **date:** DATETIME (*8 байт*) - дата и время изменения в системе
- **type:** VARCHAR(20) (*~20 + 2 байт*) - тип изменения
- **description:** VARCHAR(200) (*~200 + 2 байт*) - описание
- **related_entity_type:** VARCHAR(20) (*~20 + 2 байт*) - тип полиморфно связанной сущности
- **related_entity_id:** INT (*4 байта*) - идентификатор полиморфно связанной сущности

Оценка объема информации, хранимой в модели

Объем памяти для сохранения всех видов объектов

Vч(User) $\sim 2 * 2 + (50 + 2) * 4 + 6 + 2 + 200 + 2 + 32 + 2 = 456$ байт

Vч(Participation) $\sim 15 + 2 + 2 = 19$ байт

Vч(Requirement) $\sim 100 + 2 = 102$ байт

Vч(CleanDay) $\sim (100 + 2) * 2 + 8 * 2 + 2 * 2 + 12 + 2 + 300 + 2 = 540$ байт

Vч(City) $\sim 100 + 2 = 102$ байт

Vч(Comment) $\sim 200 + 2 + 8 = 210$ байт

Vч(Location) $\sim 150 + 2 + 200 + 2 = 354$ байт

Vч(Image) ~ 1048878 байт $\sim 1024,3$ КБ ~ 1 МБ

Vч(Log) $\sim 8 + (20 + 2) * 2 + 200 + 2 = 254$ байт

Vч(Tag) $\sim 20 + 2 = 22$ байта

Зависимость общего объема от количества объектов

Принятая переменная: U - количество зарегистрированных пользователей сервиса.

Таблица 3 – Объем данных каждого вида сущности

Сущность	Количество	“Чистый” объем данных	Реальный объем данных
User	U	456 U байт	464 U байт
CleanDay	C = 1/5 U	108 U байт	110.4 U байт
Location	C = 1/5 U	70.8 U байт	71.6 U байт
Requirement	3C = 3/5 U	61.2 U байт	63.6 U байт
Tag	3C = 3/5 U	13.2 U байт	15.6 U байт
City	U + C = 6/5 U	122.4 U байт	127.2 U байт
Participation	3U	57 U байт	93 U байт
Comment	U	210 U байт	214 U байт
Image	U + 4*C = 9/5 U	1887980.4 U байт	1887987.6 U байт
Log	9U (сумма всех количеств выше)	2438.4 U байт	2515.2 U байт
Итого без изображений	16,2 U	3537 U байт	3674.6 U байт
Итого	18 U	1891517.4 U байт	1891662.2 U байт

Теперь проанализируем количество связей в зависимости от количества пользователей U:

- **Participation_Requirement** = 2 U
- **CleanDay_Requirement** = 3/5 U
- **CleanDay_Tag** = 3/5 U
- **Participation_Comment** = U
- **CleanDay_Comment** = U
- **User_Image** = U
- **CleanDay_Image** = 2/5 U
- **Location_Image** = 2/5 U

Количество всех связей без учета изображений: 6.2 U

Количество всех связей: 7 U

Суммарный объем всех связей без учета изображений: 6.2 U * 8 байт = 49.6 U байт

Суммарный объем всех связей: 7 U * 105 байт = 56 U байт

Объем всех «чистых» данных без учета изображений = 3537 U байт

Суммарный объем всех «чистых» данных = 1891517.4 U байт

Фактический суммарный объем всех данных без учета изображений = 3674.6 U + 49.6 U = 3724.2 U байт

Фактический суммарный объем всех данных, хранимых в базе = 1891517.4 U + 56 U = 1891573.4 U байт

Конечная формула:

Чистый объем данных $V_{\text{ч}}(U)$ = 1891517.4 U байт

Фактический объем данных $V(U)$ = 1891573.4 U байт,

где U - количество зарегистрированных в системе пользователей

Избыточность данных

Формула (отношение между фактическим объемом модели и «чистым» объемом данных) - зависимость избыточности от одной переменной (с числовыми коэффициентами).

Чистый объем данных ($V_{\text{ч}}$)

Чистый объем данных — это объем без учета структуры хранения, связей и метаинформации. Мы учитываем только содержимое полей, которые несут полезную информацию.

Чистый объем данных $V_{\text{ч}}(U)$ = 3674.6 U байт - Без учёта изображения

Чистый объем данных $V_{\text{ч}}(U)$ = 1891517.4 U байт

Фактический объем данных (V)

Фактический объем данных (V) с учетом всех метаданных и структуры хранения, как рассчитано выше, выражается следующей формулой:

Фактический объем данных $V(U)$ = 3724.2 U байт - Без учёта изображения

Фактический объем данных $V(U)$ = 1891573.4 U байт

Формула избыточности (E)

$$\text{Вычислим избыточность модели: } E(U) = \frac{V(U)}{V_{\text{Ч}}(U)} = \frac{1891573.4U}{1891517.4U} = 1.00001$$

Тот же расчет, но без учета изображений, которые занимают большую часть памяти: $E(U) = \frac{V(U)}{V_{\text{Ч}}(U)} = \frac{3724.2U}{3674.6U} = 1.013$

Второй расчет более информативен, поскольку объем памяти, занимаемых изображениями, в разы больше объема памяти, занимаемого остальными данными. Кроме того, объем памяти, занимаемых изображениями, одинаков и в «чистых» данных, и в фактических.

Таким образом, избыточность модели без учета изображений существенна и составляет 1.3%.

Наличие изображений сводит избыточность модели к минимуму - 0,001%.

Направление роста модели

Нетрудно догадаться, что наибольшее влияние на рост модели оказывают изображения: их и много (4U), и объем памяти для одного изображения многократно превышает объем памяти, необходимый для любой другой сущности.

На втором месте по влиянию на рост модели находятся логи по тем же причинам: большое количество логов (9 U) + значительный «вес» одного лога.

Наименьшее влияние на модель оказывают требования для участия в субботнике - их немного, и они состоят всего из одного поля, занимающего незначительный объем памяти.

Примеры данных

City

Табл. 4 – Пример сущности City

city_id	name
1	Санкт-Петербург
2	Москва

CleanDay

Табл. 5 – Пример сущности CleanDay

cleanday_id	name	begin_date	end_date	organization	area	status	recommended_count	description	city_id	location_id
1	Субботничеク	2025-04-13 09:00:00	2025-04-13 12:00:00	ЛЭТИ	50	completed	20	Участникам можно не идти на пары	1	1

CleanDay_Comment

Табл. 6 – Пример сущности CleanDay

cleanday_id	comment_id
1	1
2	1

CleanDay_Image

Табл. 7 – Пример сущности CleanDay

cleanday_id	image_id
1	1

CleanDay_Requirement

Табл. 8 – Пример сущности CleanDay_Requirement

cleanday_id	requirement_id
1	1
1	2

CleanDay_Tag

Табл. 9 – Пример сущности CleanDay_Tag

cleanday_id	tag_id
1	1
1	2

Comment

Табл. 10 – Пример сущности Comment

comment_id	text	date
1	Люблю субботники, ёмаё	2025-04-01 12:00:00

Image

Табл. 11 – Пример сущности Image

image_id	name	description	photo
1	cleanday1.jpg	Изображение с субботника 1	BLOB
2	location1.jpg	Изображение локации 1	BLOB
3	avatar1.jpg	Фото профиля пользователя 1	BLOB

Location

Табл. 12 – Пример сущности Location

location_id	address	instructions
1	Улица Профессора Попова, дом 5	Придётся лезть через забор

Location_Image

Табл. 13 – Пример сущности Location_Image

location_id	image_id
1	2

Log

Табл. 14 – Пример сущности Log

log_id	date	type	description		related_entity_type	related_entity_id
1	2025-04-01 12:00:00	create	Пользователь принял участие в субботнике		Participation	1

Participation

Табл. 15 – Пример сущности Participation

participation_id	type	stat	user_id	cleanday_id
1	Точно пойду	1	1	1

Participation_Comment

Табл. 16 – Пример сущности Participation_Comment

participation_id	comment_id
1	1

Participation_Requirement

Табл. 17 – Пример сущности Participation_Requirement

participation_id	requirement_id
1	1
1	2

Requirement

Табл. 18 – Пример сущности Requirement

requirement_id	name
1	Взять лопату
2	Выглядеть эффектно

Tag

Табл. 19 – Пример сущности Tag

tag_id	name
1	tag1
2	tag2

User

Табл. 20 – Пример сущности User

use_r_id	first_name	last_name	middle_name	s_ex	lo_gi_n	password	score	level	about_me	city_id
1	Юрий	Борисов	Александрович	m_a_l_e	y_u_r_a_b	482C811DA5D5B 4BC6D497FFA98 491E38	0	0	Я люблю субботники	1

User_Image

Табл. 21 – Пример сущности User_Image

user_id	image_id
1	3

Примеры запросов

Регистрация нового пользователя

Получение пользователя по логину

```
SELECT *
FROM User
WHERE login = @login;
```

Создание пользователя

```
INSERT INTO User (first_name, last_name, email, sex, login, password, about_me, city_id)
VALUES ('Иван', 'Иванов', 'jane@example.com', 'male', 'ivan', 'hash', 'Обо мне', @city_id);
```

Получение городов по имени

```
SELECT *
FROM City
WHERE LOWER(name) LIKE LOWER(CONCAT('%', @name, '%'));
```

Добавление города к пользователю

```
UPDATE User
SET city_id = @city_id
WHERE user_id = @user_id;
```

Добавление аватара пользователя

```
-- Шаг 1: Вставка изображения
INSERT INTO Image (description, photo)
VALUES ('avatar', @file);
```

```
-- Шаг 2: Связь пользователя с изображением
INSERT INTO User_Image (user_id, image_id)
VALUES (@user_id, @new_image_id);
```

- Запросов: **6**
- Задействованные коллекции: User, City, Image, User_Image
- Примечание: между запросами необходимо сохранение новых id или использование вложенных запросов

Авторизация пользователя

См выше:

- Получение пользователя по логину
- Запросов: **1**
- Задействованные коллекции: User

Просмотр информации о своём профиле

Получение пользователя по id

```
SELECT *
FROM User
WHERE user_id = @id;
```

Получение города пользователя по его id

```
SELECT c.*
FROM City c
JOIN User u ON c.city_id = u.city_id
WHERE u.user_id = @id;
```

- Запросов: **2 + 1 вложенный**
- Задействованные коллекции: User, City

Обновление данных пользователя

См выше:

- Получение городов по имени
- Добавление города к пользователю

Изменение данных пользователя по его id

```
UPDATE User
SET first_name = @first_name
last_name = @last_name
middle_name = @middle_name
sex = @sex
password = @password
about_me = @about_me
score = @score
level = @level
```

```
    city_id = @city_id
WHERE user_id = @id;
-- Примечание: поля для обновления зависят от передаваемых параметров
```

Удаление связи города с пользователем по id

```
UPDATE User
SET city_id = NULL
WHERE user_id = @user_id;
```

Обновление аватара пользователя

```
UPDATE Image
SET photo = @file
WHERE image_id = (SELECT image_id FROM User_Image WHERE user_id = @user_id LIMIT 1);
```

- Запросов: **3 + 1 вложенный**
- Задействованные коллекции: User, City, Image, User_Image

Просмотр всех пользователей

См. выше:

- [Получение города пользователя по его id](#)

Получение пользователей с пагинацией

```
SELECT *
FROM User
ORDER BY user_id
LIMIT @count OFFSET @offset;
```

- Запросов: **2**
- Задействованные коллекции: User, City
- Замечание: важно отсортировать результат запроса для корректной пагинации

Поиск пользователей по совпадению

Получение пользователей с фильтрацией по атрибутам

```
SELECT *
FROM User
WHERE LOWER(first_name) LIKE LOWER(CONCAT('%', @first_name_filter, '%'))
AND LOWER(email) LIKE LOWER(CONCAT('%', @email_filter, '%')) -- Условия добавляются через AND для
других полей из @filter_params
ORDER BY user_id
LIMIT @limit OFFSET @offset;
```

- Запросов: **2**
- Задействованные коллекции: User, City
- Замечания:

- важно отсортировать результат запроса для корректной пагинации
- запрос должен строиться динамически в зависимости от искомых параметров

Сложный поиск пользователей по атрибутам

См. выше:

- Получение пользователей с фильтрацией по атрибутам
- Запросов: **2**
- Задействованные коллекции: User, City

Фильтрация пользователей по атрибуту

Получение пользователей с сортировкой

```
SELECT *
FROM User
ORDER BY first_name ASC
LIMIT @count OFFSET @offset;
```

Прим. Другие условия сортировки строятся похожим образом.

- Запросов: **2**
- Задействованные коллекции: User, City

Просмотр информации о пользователе

См выше:

- Получение пользователя по id
- Получение города пользователя по его id
- Запросов: **2**
- Задействованные коллекции: User, City

Просмотр списка субботников, в которых участвовал пользователь

Получение субботников пользователя

```
SELECT cd.*
FROM CleanDay cd
JOIN Participation p ON cd.cleanday_id = p.cleanday_id
WHERE p.user_id = @id;
```

- Запросов: **1**
- Задействованные коллекции: User, Participation, CleanDay

Просмотр списка субботников, которые организовал пользователь

Получение организованных пользователем субботников

```
SELECT cd.*  
FROM CleanDay cd  
JOIN Participation p ON cd.cleanday_id = p.cleanday_id  
WHERE p.user_id = @id AND p.type = 'organiser';
```

- Запросов: 1
- Задействованные коллекции: User, Participation, CleanDay

Просмотр всех субботников

Получение субботников с пагинацией

```
SELECT *  
FROM CleanDay  
ORDER BY cleanday_id  
LIMIT @count OFFSET @offset;
```

Получение города, в котором проходит субботник, по его id

```
SELECT c.*  
FROM City c  
JOIN CleanDay cd ON c.city_id = cd.city_id  
WHERE cd.cleanday_id = @id;
```

- Запросов: 1
- Задействованные коллекции: CleanDay, City
- Замечание: важно отсортировать результат запроса для корректной пагинации

Поиск субботников по совпадению

Получение субботников с фильтрацией по атрибутам

```
SELECT *  
FROM CleanDay  
WHERE LOWER(name) LIKE LOWER(CONCAT('%', @name_filter, '%'))  
AND status = @status_filter -- Условия добавляются через AND для других полей из @filter_params  
ORDER BY cleanday_id  
LIMIT @limit OFFSET @offset;
```

- Запросов: 1
- Задействованные коллекции: CleanDay, City
- Замечания:
 - важно отсортировать результат запроса для корректной пагинации
 - запрос должен строиться динамически в зависимости от искомых

параметров

Сложный поиск субботников по атрибутам

См. выше:

- Получение субботников с фильтрацией по атрибутам
- Запросов: 1
- Задействованные коллекции: CleanDay, City

Фильтрация субботников с помощью чекбокса «Предстоящие субботники»

Аналогично предыдущему, отличие в условии фильтрации.

- Запросов: 1
- Задействованные коллекции: CleanDay, City

Фильтрация субботников с помощью чекбокса «С моим участием»

См. выше:

- Получение субботников пользователя
- Запросов: 1
- Задействованные коллекции: CleanDay, User

Просмотр информации о субботнике

См выше:

- Получение города, в котором проходит субботник по его id

Получение субботника по id

```
SELECT *
FROM CleanDay
WHERE cleanday_id = @id;
```

Получение локации субботника по его id

```
SELECT l.*
FROM Location l
JOIN CleanDay cd ON l.location_id = cd.location_id
WHERE cd.cleanday_id = @id;
```

Получение фото субботника по его id

```
SELECT i.*
FROM Image i
```

```
JOIN CleanDay_Image cdi ON i.image_id = cdi.image_id  
WHERE cdi.cleanday_id = @id;
```

- Запросов: 3
- Задействованные коллекции: CleanDay, City, Location, CleanDay_Image

Просмотр списка всех участников субботника

Получение всех участников субботника

```
SELECT u.*  
FROM User u  
JOIN Participation p ON u.user_id = p.user_id  
WHERE p.cleanday_id = @id;
```

- Запросов: 1
- Задействованные коллекции: Participation, User

Просмотр списка участников субботника, подходящих под определенное условие

Получение всех участников субботника с условием

```
SELECT u.*  
FROM User u  
JOIN Participation p ON u.user_id = p.user_id  
WHERE p.cleanday_id = @id AND u.first_name = 'Олег';
```

Прим. Условие фильтрации в реальных запросах может варьироваться

- Запросов: 1
- Задействованные коллекции: Participation, User

Просмотр списка участников субботника с определенным статусом участия

Получение всех участников субботника с определённым статусом участия

```
SELECT u.*  
FROM User u  
JOIN Participation p ON u.user_id = p.user_id  
WHERE p.cleanday_id = @id AND p.type = @type;
```

- Запросов: 1
- Задействованные коллекции: Participation, User

Получение участников, не пришедших на субботник

```
SELECT DISTINCT u.*  
FROM User u
```

```
JOIN Participation p ON u.user_id = p.user_id  
WHERE p.type = 'Точно пойду'  
    AND p.real_presence = false;
```

- Запросов: 1
- Задействованные коллекции: User, Participation

Просмотр истории активности участников субботника

Получение логов активности субботника

```
SELECT  
    l.* , -- Все поля из Log  
    u.user_id, u.first_name, u.last_name, -- Выборочные поля пользователя  
    c.comment_id, c.text, c.time -- Выборочные поля комментария  
FROM  
    Log l  
LEFT JOIN  
    User u ON l.user_id = u.user_id -- LEFT JOIN если пользователь может быть не указан  
LEFT JOIN  
    Comment c ON l.comment_id = c.comment_id -- LEFT JOIN если комментарий может быть не указан  
WHERE  
    l.cleanday_id = @id  
ORDER BY l.timestamp DESC;
```

- Запросов: 1
- Задействованные коллекции: Log, User, Comment

Участие в субботнике

Добавление участия пользователя с типом

```
INSERT INTO Participation (user_id, cleanday_id, type, stat)  
VALUES (@user_id, @cleanday_id, @participation_type, 0);
```

- Запросов: 3
- Задействованные коллекции: Participation

Изменить участие в субботнике

Обновление участия пользователя

```
UPDATE Participation  
SET type = @participation_type  
WHERE user_id = @user_id AND cleanday_id = @cleanday_id;
```

- Запросов: 1
- Задействованные коллекции: Participation

Добавление комментария

Добавления комментария пользователя в субботнике

-- Шаг 1: Нахождение ID участия (Participation)

```
SELECT participation_id  
FROM Participation  
WHERE user_id = @user_id AND cleanday_id = @cleanday_id;
```

-- Шаг 2: Вставка сам комментарий

```
INSERT INTO Comment (text, time)  
VALUES (@text, @time);
```

-- Шаг 3: Связывание участия и комментария

```
INSERT INTO Participation_Comment (participation_id, comment_id)  
VALUES (@p_id, @new_comment_id);
```

- Запросов: 3
- Задействованные коллекции: Participation, Comment, Participation_Comment

Просмотр всех организованных пользователем субботников

См. выше:

- Получение организованных пользователем субботников

Организация нового субботника

См. выше:

- Получение городов по имени

Создание нового субботника

-- Шаг 1: Вставка записи о субботнике

```
INSERT INTO CleanDay (name, begin_date, end_date, organization, area, status, tags, recommended_count,  
description, location_id, city_id)  
VALUES (@name, @begin_date, @end_date, @organization, @area, @status, @tags, @recommended_coun  
t, @description, @location_id, @city_id);
```

-- Шаг 2: Регистрация организатора как участника

```
INSERT INTO Participation (user_id, cleanday_id, type, stat)  
VALUES (@user_id, @new_cleanay_id, 'organiser', 0); -- @new_cleanay_id получен на шаге 1
```

Добавление фото субботника

-- Шаг 1: Вставка изображения

```
INSERT INTO Image (description, photo)  
VALUES (@description, @file);
```

-- Шаг 2: Связывание субботника с изображением

```
INSERT INTO CleanDay_Image (cleanday_id, image_id)  
VALUES (@id, @new_image_id);
```

- Запросов: **5**
- Задействованные коллекции: CleanDay, Participation, Image, CleanDay_Image

Отмена субботника

Изменение параметров субботника

```
UPDATE CleanDay
SET name = @name,
begin_date = @begin_date,
end_date = @end_date,
organization = @organization,
area = @area,
status = @status,
tags = @tags,
recommended_count = @recommended_count,
description = @description,
location_id = @location_id,
city_id = @city_id
WHERE cleanday_id = @id;
```

- Запросов: **1**
- Задействованные коллекции: CleanDay

Изменение субботника

См. выше:

- Изменение параметров субботника
- Добавление фото субботника
- Запросов: **2**
- Задействованные коллекции: CleanDay

Получение организаторов с самой плохой явкой

```
WITH OrganizedCleanDays AS (
    -- Находим все пары пользователь-субботник, где пользователь является организатором
    SELECT DISTINCT
        p.user_id,
        p.cleanday_id
    FROM Participation p
    WHERE p.type = 'Организатор'
),
CleanDayAttendanceStats AS (
    -- Для каждого субботника считаем тех, кто «Точно пойду» и тех из них, кто не пришел
    SELECT
        p.cleanday_id,
    -- Считаем общее количество «Точно пойду»
```

```

COUNT(CASE WHEN p.type = 'Точно пойду' THEN 1 END) AS totalWillGo,
-- Считаем количество «Точно пойду», но не пришедших (real_presence = false)
COUNT(CASE WHEN p.type = 'Точно пойду' AND p.real_presence = false THEN 1 END) AS missed
FROM Participation p
WHERE EXISTS (SELECT 1 FROM OrganizedCleanDays ocd WHERE ocd.cleanday_id = p.cleanday_id)
AND p.type = 'Точно пойду'
GROUP BY p.cleanday_id
),
OrganizerEventRatio AS (
-- Рассчитываем коэффициент неявки для каждого субботника, организованного пользователем
SELECT
ocd.user_id,
ocd.cleanday_id,
cas.totalWillGo,
cas.missed,
-- Рассчитываем коэффициент, обрабатывая деление на ноль
CASE
WHEN cas.totalWillGo > 0 THEN CAST(cas.missed AS REAL) / cas.totalWillGo
ELSE NULL
END AS no_show_ratio
FROM OrganizedCleanDays ocd
JOIN CleanDayAttendanceStats cas ON ocd.cleanday_id = cas.cleanday_id
WHERE cas.totalWillGo > 0
),
OrganizerSummary AS (
-- 4. Считаем средний коэффициент неявки и общее число организованных субботников для каждого организатора
SELECT
oer.user_id,
-- Средний коэффициент неявки по всем субботникам пользователя
AVG(oer.no_show_ratio) AS avgNoShowRatio,
-- Общее количество организованных субботников
(SELECT COUNT(*) FROM OrganizedCleanDays ocd WHERE ocd.user_id = oer.user_id) AS organizedCount
FROM OrganizerEventRatio oer
GROUP BY oer.user_id
)
-- Соединяем данные пользователя с вычисленной статистикой
SELECT
u.*,
os.organizedCount,
os.avgNoShowRatio
FROM User u
JOIN OrganizerSummary os ON u.user_id = os.user_id
ORDER BY
os.avgNoShowRatio DESC; -- Сортируем по убыванию среднего коэффициента неявки

```

- Запросов: **3 на создание промежуточных коллекций и 1 для вывода результата**
- Задействованные коллекции: User, Participation

СРАВНЕНИЕ МОДЕЛЕЙ

Удельный объем информации

Таблица 22 – Удельный объем информации

Параметр	NoSQL	SQL
Формула объема(байт)	$V_1(U) = 1894491,8 \times U$	$V_2(U) = 1891573,4 \times U$
Зависимость от U	линейная	линейная
Избыточность без учёта изображений	1,989 (98.9% метаданных)	1,013 (1.3% метаданных)
Избыточность с учётом изображений	1.0018 (0.18% метаданных)	1.00001 (0,001% метаданных)

SQL экономичнее по объему при любом U, в то время как NoSQL имеет более высокую избыточность и больший объем, из-за явного хранения связей. Но так как в базах данных хранятся изображения, эта разница и избыточность становятся незначительны.

Запросы по отдельным юзкейсам

Таблица 23 – Запросы по отдельным юзкейсам

Юзкейс	NoSQL	SQL
Регистрация нового пользователя	6 запросов	5 запросов
Авторизация пользователя	1 запрос	1 запрос
Просмотр информации о своём профиле	2 запроса	2 запроса + 1 вложенный
Обновление данных пользователя	от 1 до 3 запросов в зависимости от обновляемых полей	3 запроса + 1 вложенный
Просмотр всех пользователей	1 запрос	2 запроса
Поиск пользователей по совпадению	1 запрос	2 запроса
Сложный поиск пользователей по атрибутам	1 запрос	2 запроса
Фильтрация пользователей по атрибуту	1 запрос	2 запроса
Просмотр информации о пользователе	2 запрос	2 запроса
Просмотр списка субботников, в которых участвовал пользователь	1 запрос	1 запрос
Просмотр списка субботников, которые организовал пользователь	1 запрос	1 запрос
Просмотр всех субботников	1 запрос	1 запрос

Юзкейс	NoSQL	SQL
Поиск субботников по совпадению	1 запрос	1 запрос
Сложный поиск субботников по атрибутам	1 запрос	1 запрос
Фильтрация субботников с помощью чекбокса «Предстоящие субботники»	1 запрос	1 запрос
Фильтрация субботников с помощью чекбокса «С моим участием»	1 запрос	1 запрос
Просмотр информации о субботнике	3 запроса	3 запроса
Просмотр списка всех участников субботника	1 запрос	1 запрос
Просмотр списка участников субботника, подходящих под определенное условие	1 запрос	1 запрос
Просмотр списка участников субботника с определенным статусом участия	1 запрос	1 запрос
Просмотр истории активности участников субботника	1 запрос	1 запрос
Участие в субботнике	3 запроса	3 запроса
Изменить участие в субботнике	1 запрос	1 запрос
Добавление комментария	1 запрос	3 запроса
Просмотр всех организованных пользователем субботников	1 запрос	1 запрос
Организация нового субботника	7 запросов	5 запросов
Отмена субботника	1 запрос	1 запрос
Изменение субботника	2 запроса	2 запроса

Как видно из данного сравнения, NoSQL в целом экономичнее SQL в плане количества запросов, и проигрывает только в двух юзкейсах, связанных с добавлением нового пользователя или создания субботника, в остальных же требует либо столько же запросов, либо меньше.

Вывод

В ходе сравнения нереляционной и реляционной моделей были сделаны следующие выводы:

1. Реляционная модель компактнее нереляционной для разработанных юзкейсов, но с учётом изображений, которые будут использоваться в базе данных, эта разница незначительна
2. Нереляционная модель упрощает взаимодействие с базой данных для юзкейсов, связанных с просмотром, поиском, фильтрацией и изменением

пользователей, и усложняет только для юзкейсов создания новых пользователей или субботников.

Стоит отметить, что запросы показаны для использования в коде, потому что были пропущены вложенные запросы на получение новодобавленных id.

РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

Краткое описание

Данное веб-приложение предназначено для организации и координации субботников. Платформа позволяет пользователям как создавать собственные субботники, так и присоединяться к уже запланированным мероприятиям. Система включает инструменты для планирования, организации и отчетности по экологическим инициативам с акцентом на поддержание чистоты в общественных пространствах.

Сервис включает следующие ключевые возможности:

- Создание и управление субботниками с подробным описанием
- Участие в субботниках с разными статусами (пойду, опоздаю, возможно пойду)
- Учет личной статистики пользователей (убранная территория, участие)
- Система уровней и опыта для геймификации процесса
- Отчеты и фотографии по проведенным мероприятиям
- Фильтрация и поиск пользователей и субботников
- Экспорт/импорт статистических данных

Использованные технологии

Для реализации приложения были использованы следующие технологии:

- Frontend:
 - *React c TypeScript* - основной фреймворк для разработки интерфейса;
 - *Material-UI (MUI)* - библиотека компонентов для создания современного UI;
 - *React Router* - управление маршрутизацией;
 - *TanStack Query* - управление состоянием и кэширование запросов;
 - *Day.js* - работа с датами и временем с поддержкой русской локализации;

- *Vite* - сборщик модулей и инструмент разработки.
- Backend:
 - *Python* - основной язык программирования;
 - *FastAPI* - высокопроизводительный веб-фреймворк для создания API;
 - *ArangoDB* - NoSQL база данных;
 - *Docker* и *docker-compose* - контейнеризация приложения.

Снимки экрана приложения

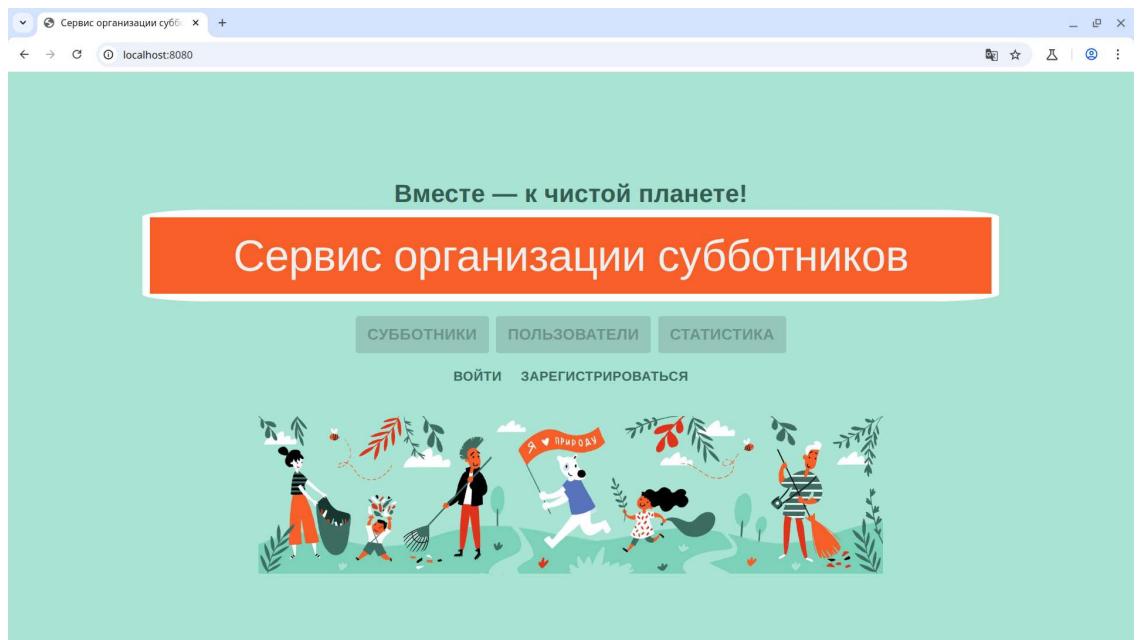


Рис. 6 – Неавторизованный пользователь зашёл на сайт

Сервис организации субботников

Имя *

Фамилия *

Логин *

Город *

Пол
 Женский Мужской Другое

Пароль *

Пароль должен содержать не менее 8 символов, среди которых не менее 1 цифры, не менее 1 строчной буквы, не менее 1 прописной буквы, не менее 1 специального символа.

Повторите пароль *

Рис. 7 – Экран регистрации

Пол
 Женский Мужской Другое

Пароль *

Пароль должен содержать не менее 8 символов, среди которых не менее 1 цифры, не менее 1 строчной буквы, не менее 1 прописной буквы, не менее 1 специального символа.

Повторите пароль *

ЗАРЕГИСТРИРОВАТЬСЯ

НА ГЛАВНУЮ ВХОД

Рис. 8 – Экран регистрации, продолжение

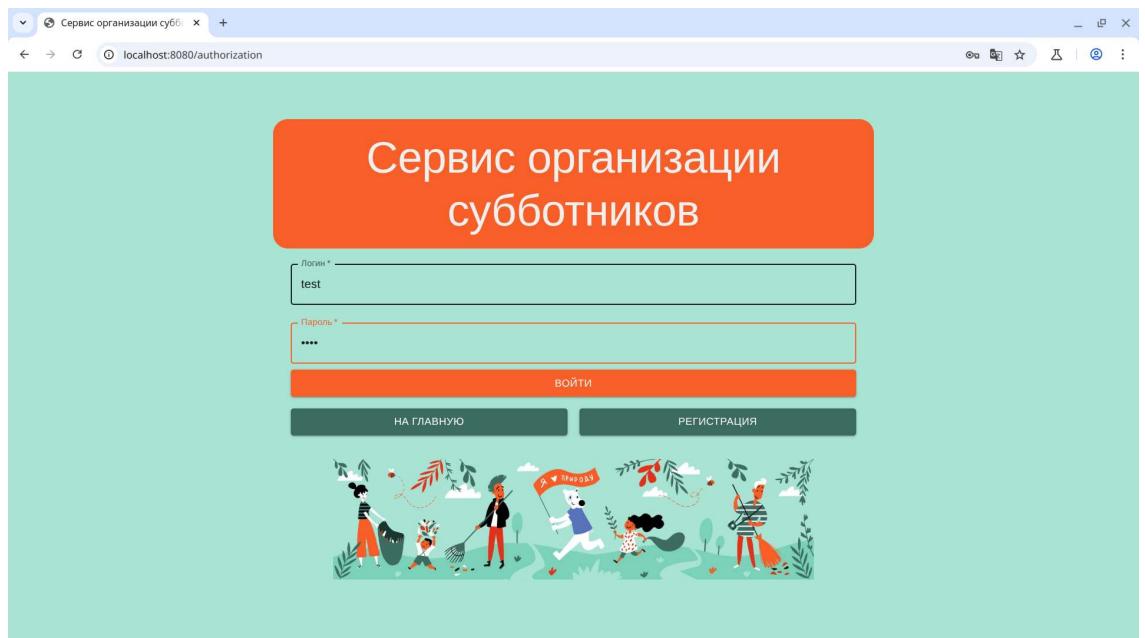


Рис. 9 – Экран входа

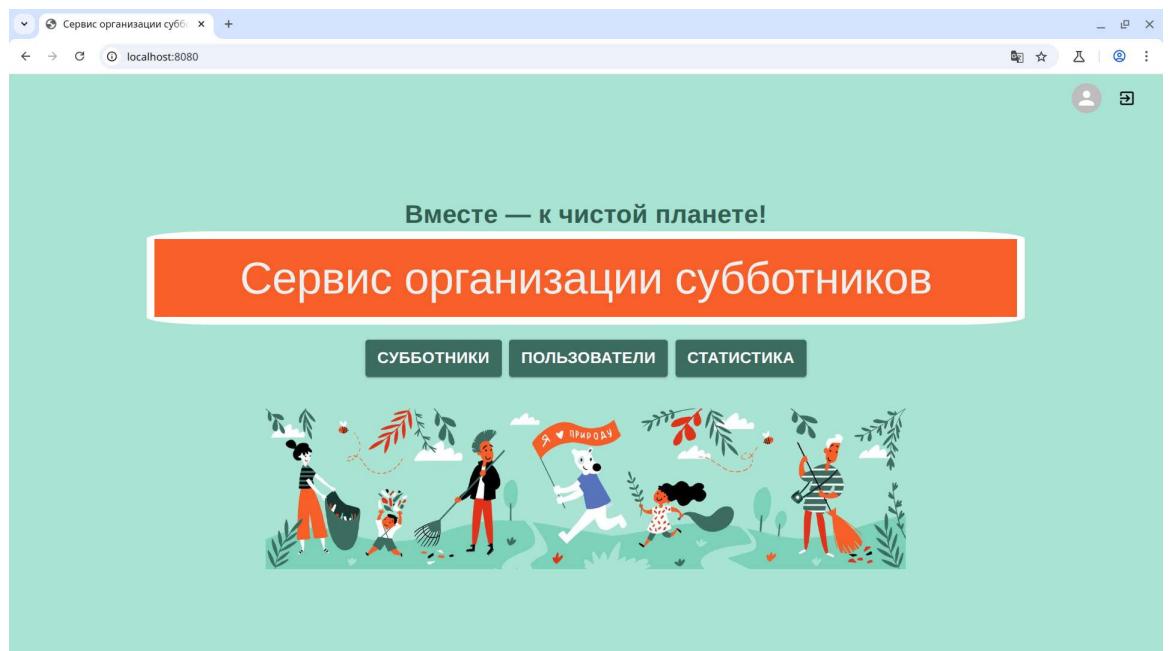


Рис. 10 – Главный экран приложения

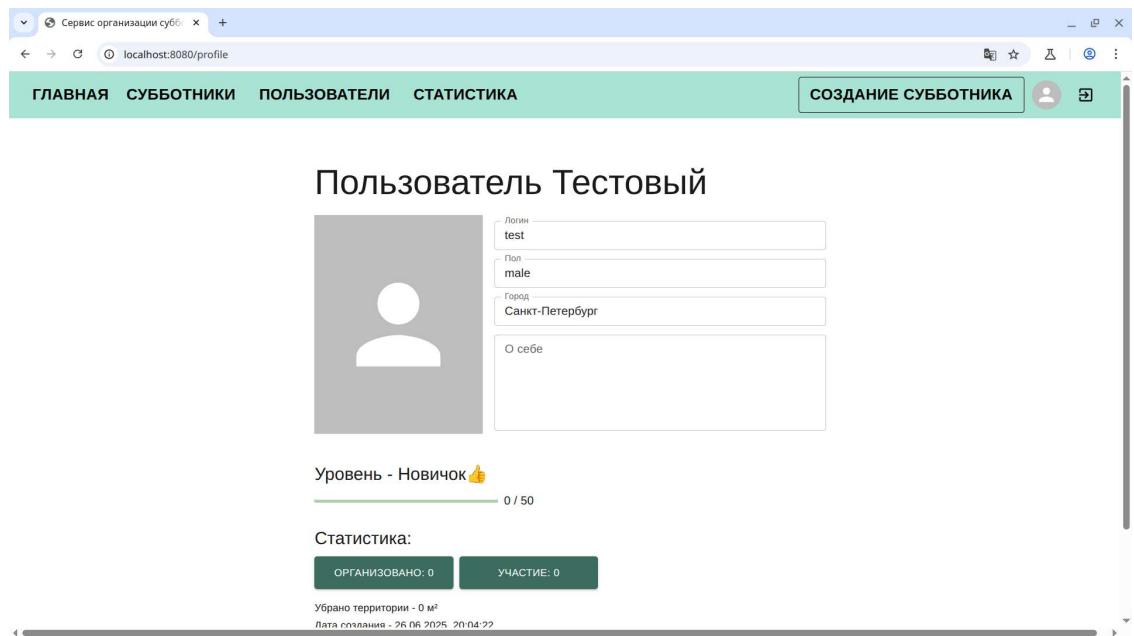


Рис. 11 – Страница пользователя

Субботники					
<input type="text"/> Найти		<input type="button" value="ПОСТРОИТЬ ГРАФИК"/>			
Название	Город	Адрес	Дата начала	Дата окончания	Организация
Очистка прудов в Парке Победы	Санкт-Петербург	Парк Победы	03.07.2025, 20:04:26	04.07.2025, 00:04:26	Союз субботников
Очистка мусора в Парке Горького	Москва	Парк Горького	04.07.2025, 20:04:26	05.07.2025, 00:04:26	Городская экология...
Посадка деревьев в Центральном ...	Таганрог	Центральный парк	05.07.2025, 20:04:26	06.07.2025, 00:04:26	Зеленый город
Уборка пляжа на Площади Кирова	Ижевск	Площадь Кирова	06.07.2025, 20:04:26	07.07.2025, 00:04:26	Чистый берег
Ремонт дорожек на Набережной Ф...	Нижний Новгород	Набережная Федоровского	07.07.2025, 20:04:26	08.07.2025, 00:04:26	Дорстрой
Покраска скамеек в Парке имени К...	Мурманск	Парк имени Кирова	08.07.2025, 20:04:26	09.07.2025, 00:04:26	Краски города
Установка кормушек для птиц на Н...	Калининград	Парк Юности	09.07.2025, 20:04:26	10.07.2025, 00:04:26	Птицы Подмосковья
Уборка мусора в Саду имени Алекс...	Санкт-Петербург	Сад имени Александра Нев...	10.07.2025, 20:04:26	11.07.2025, 00:04:26	Зеленая столица
Посадка цветов на Площади Ленина	Москва	Площадь Ленина	11.07.2025, 20:04:26	12.07.2025, 00:04:26	Цветочный город
Уборка мусора в Парке Дружбы	Ижевск	Парк Дружбы	12.07.2025, 20:04:26	13.07.2025, 00:04:26	Чистый город

Рис. 12 – Список субботников

Рис. 13 – Фильтрация субботников по месту проведения

Рис. 14 – Фильтрация субботников по дате

Сервис организации субботников

localhost:8080/cleandays/859

ГЛАВНАЯ СУББОТНИКИ ПОЛЬЗОВАТЕЛИ СТАТИСТИКА

СОЗДАНИЕ СУББОТНИКА

Информация о субботнике

1 Запланирован 26.06.2025 2 Проходит 03.07.2025 3 Завершен 04.07.2025

Название субботника
Очистка прудов в Парке Победы

Общая информация

Название локации — Парк Победы

Город — Санкт-Петербург Адрес — Парк Победы

Описание локации
Парк находится к востоку от вестибюля м. Парк Победы

Дата начала — 03.07.2025 Время начала — 20:04:26

Дата конца — 04.07.2025 Время конца — 00:04:26

Организатор — boriss

Организация — Союз субботников

Площадь, м² — 1000

Комментарии

Новое сообщение ➤

Рис. 15 – Информация о субботнике

Сервис организации субботников

localhost:8080/cleandays/859

ГЛАВНАЯ СУББОТНИКИ ПОЛЬЗОВАТЕЛИ СТАТИСТИКА

СОЗДАНИЕ СУББОТНИКА

ИТОГИ СУББОТНИКА

Соответствие условиям участия:

Условие 1
Участие во взносе за мусоровоз

Условие 2
Приди со своими инструментами

Дата создания - 26.06.2025, 20:04:26
Дата последнего изменения - 26.06.2025, 20:04:26

ID: 859

ИЗМЕНİТЬ УЧАСТИЕ В СУББОТНИКЕ РЕДАКТИРОВАТЬ ЗАВЕРШИТЬ СУББОТНИК (+10 ОПЫТА)

ИСТОРИЯ АКТИВНОСТИ СУББОТНИКА НАЗАД

Участие в субботнике
Очистка прудов в Парке Победы

Подтвердите соответствие требованиям:

Перчатки
 Хорошее настроение

Выберите статус вашего участия:

TOЧНО ПОЙДУ (+5 опыта)

ОПОЗДЯЮ (+4 опыта)

ВОЗМОЖНО, ПОЙДУ (+3 опыта)

НЕ ПОЙДУ

ОТМЕНА СОХРАНИТЬ

Рис. 16 – Изменение статуса участия в субботнике

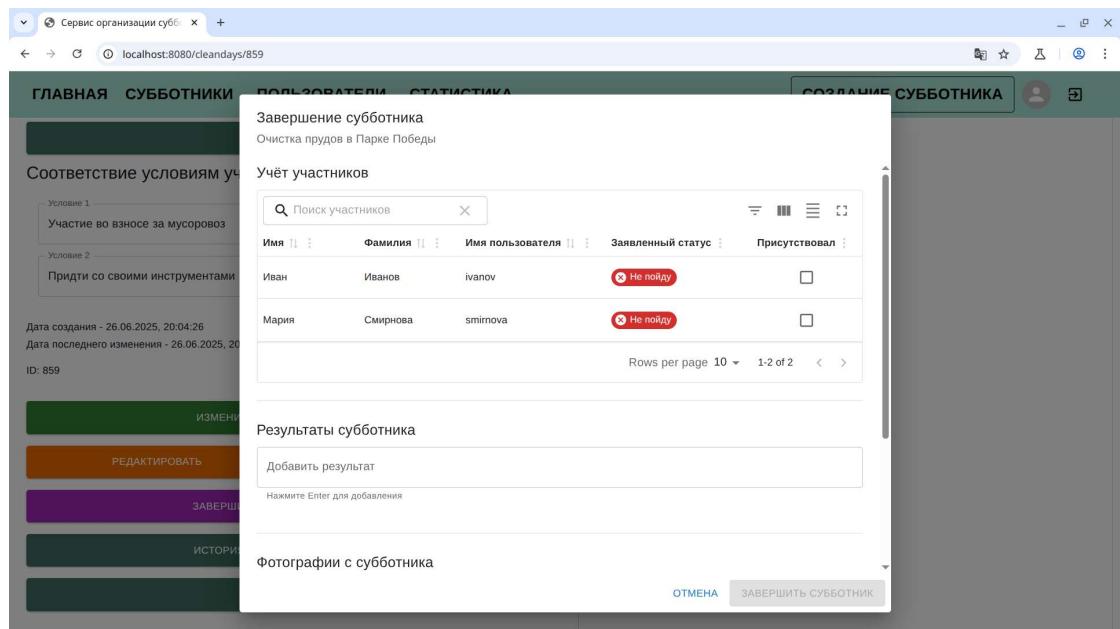


Рис. 17 – Меню завершения субботника

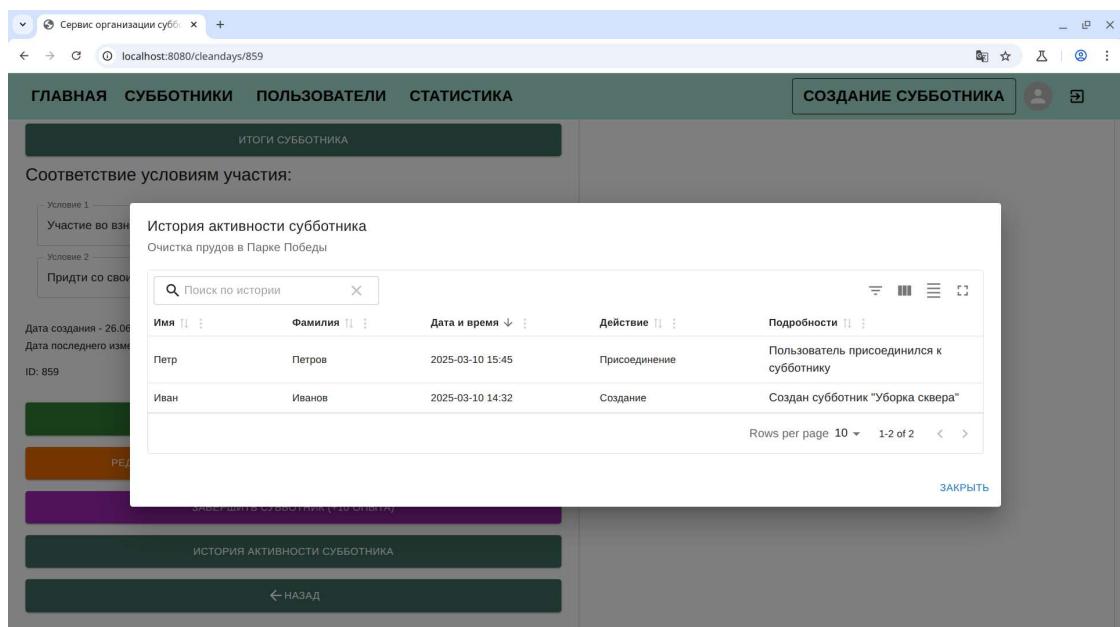


Рис. 18 – История активности субботника

Сервис организации субботников

localhost:8080/users

ГЛАВНАЯ СУББОТНИКИ ПОЛЬЗОВАТЕЛИ СТАТИСТИКА

СОЗДАНИЕ СУББОТНИКА

Пользователи

Найти ПОСТРОИТЬ ГРАФИК

Имя	Фамилия	Логин	Город	Посещённые субботники	Организованные субботники	Убрано, м ²	Уровень
Алексей	Романов	alek_rom	Мурманск	1	1	0	1
Анна	Петрова	annpar	Москва	1	1	0	1
Артём	Смирнов	art_smir	Калининград	0	0	0	1
Борис	Сухачёв	boriss	Санкт-Петербург	2	2	0	1
Дмитрий	Кузнецов	dmitrik	Ижевск	1	1	0	1
Иван	Сидоров	ivansid	Калининград	1	1	0	1
Елена	Морозова	lenamor	Санкт-Петербург	0	0	0	1
Мария	Иванова	mashaivan	Таганрог	1	1	0	1
Максим	Орлов	max_orlov	Санкт-Петербург	0	0	0	1
Наталья	Григорьева	natali_g	Ижевск	1	1	0	1

Строк на странице 10 1-10 из 20 |< < > >|

localhost:8080/users

Рис. 19 – Список пользователей

Сервис организации субботников

localhost:8080/users

ГЛАВНАЯ СУББОТНИКИ ПОЛЬЗОВАТЕЛИ СТАТИСТИКА

СОЗДАНИЕ СУББОТНИКА

Пользователи

Найти ПОСТРОИТЬ ГРАФИК

Имя	Фамилия	Логин	Город	Посещённые субботники	Организованные субботники	Убрано, м ²	Уровень
Алексей	Романов	alek_rom	Мурманск	1	0	1	
Анна	Петрова	annpar	Москва	1	0	1	
Артём	Смирнов	art_smir	Калининград	0	0	1	
Борис	Сухачёв	boriss	Санкт-Петербург	2	0	1	
Дмитрий	Кузнецов	dmitrik	Ижевск	1	0	1	
Иван	Сидоров	ivansid	Калининград	1	0	1	
Елена	Морозова	lenamor	Санкт-Петербург	0	0	1	
Мария	Иванова	mashaivan	Таганрог	1	0	1	
Максим	Орлов	max_orlov	Санкт-Петербург	0	0	1	
Наталья	Григорьева	natali_g	Ижевск	1	0	1	

Сортировка Город по возрастанию

Сортировка Город по убыванию

Очистить фильтр

Отфильтровать по Город

Скрыть Город колонку

Показать все колонки

Строк на странице 10 1-10 из 20 |< < > >|

Рис. 20 – Фильтрация и сортировка пользователей

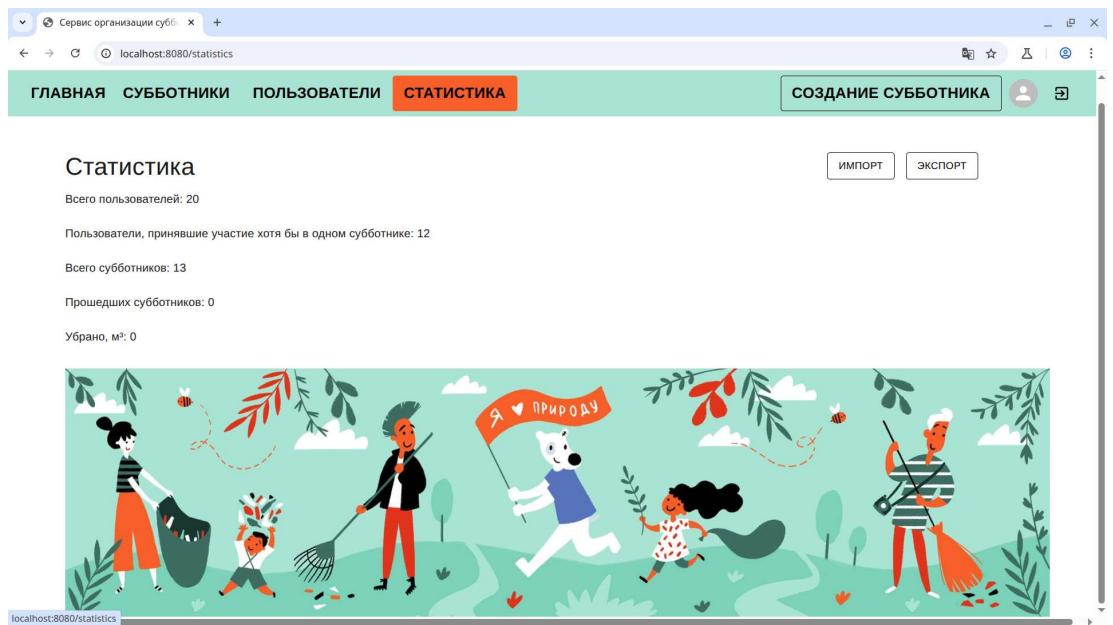


Рис. 21 – Статистика приложения

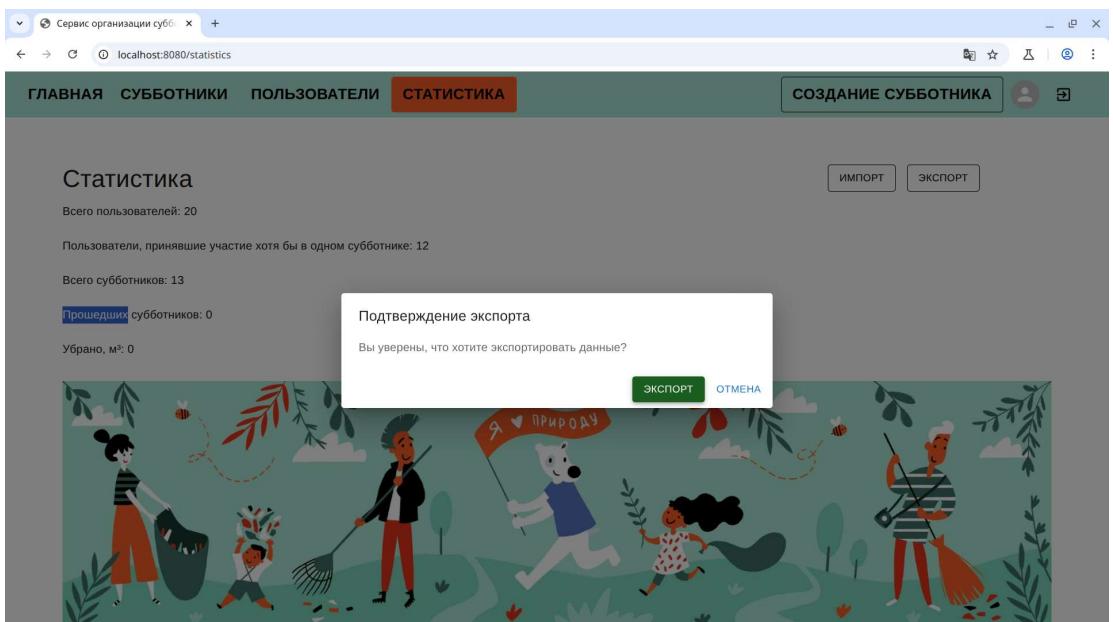


Рис. 22 – Экспорт базы данных

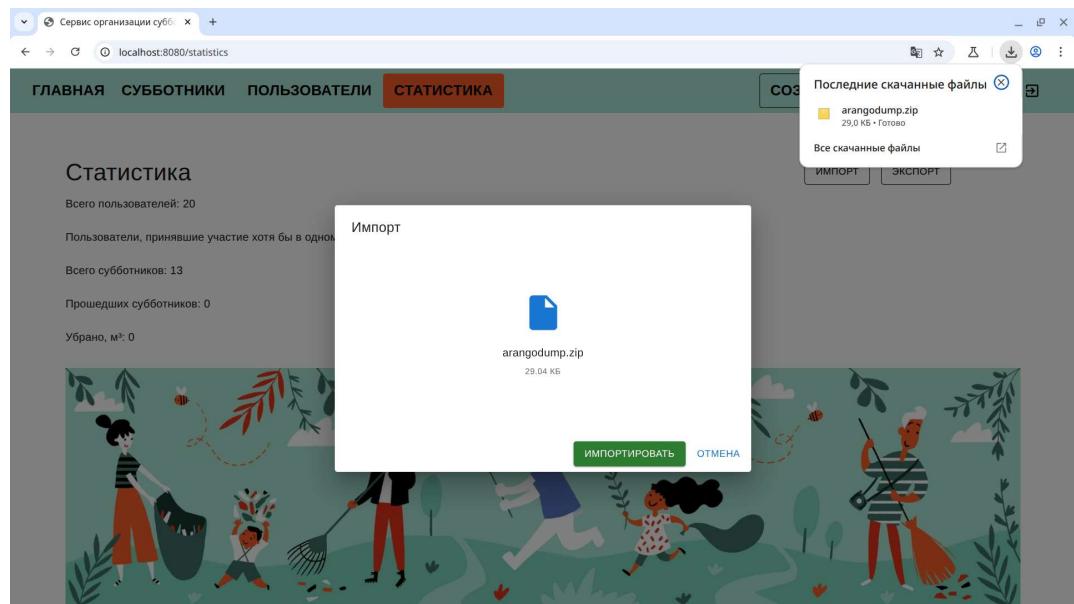


Рис. 23 – Импорт базы данных

ВЫВОДЫ

Достигнутые результаты

В ходе выполнения работы был разработан и реализован сервис для организации и проведения городских субботников, построенный на документно-графовой базе данных ArangoDB. В процессе работы была создана схема базы данных, включающая коллекции для пользователей, городов, локаций, субботников, требований, комментариев, логов и других сущностей, а также рёбер для моделирования связей между ними. Для удобства развертывания и эксплуатации системы использованы современные инструменты контейнеризации - Docker.

Фронтенд-часть приложения реализована на React и TypeScript с использованием современных библиотек для управления состоянием, запросами и стилизацией. Она обеспечивает интуитивно понятный интерфейс для пользователей и организаторов:

- Регистрация, авторизация и управление профилем пользователя;

- Просмотр списка субботников, подробной информации о каждом мероприятии, включая участников, требования, фотографии, комментарии и логи;
- Возможность присоединиться к субботнику, оставлять комментарии, просматривать статистику и историю участия;
- Для организаторов реализованы функции создания и редактирования субботников, управления требованиями и локациями.

Бэкенд реализован на Python с использованием FastAPI и инкапсулирует бизнес-логику приложения:

- Обработка запросов от клиента, валидация данных и безопасное взаимодействие с ArangoDB;
- Реализация CRUD-операций для всех ключевых сущностей (пользователи, субботники, локации, требования, комментарии и др.);
- Поддержка сложных выборок, фильтрации, сортировки и пагинации;
- Механизмы импорта и экспорта данных;
- Агрегация статистики для отображения на клиенте.

Взаимодействие между фронтеном и бэкендом осуществляется через REST API. Приложение поддерживает все основные сценарии работы с мероприятиями, пользователями и статистикой.

Недостатки и пути для улучшения полученного решения

В процессе разработки и тестирования были выявлены следующие аспекты, требующие доработки:

1. *Визуализация связанных сущностей*: в ряде случаев в интерфейсе отображаются внутренние идентификаторы вместо информативных наименований (например, для локаций или требований).
2. *Адаптивность интерфейса*: пользовательский интерфейс требует

доработки для оптимальной работы на мобильных устройствах и планшетах.

3. *Обработка ошибок:* возможна более детальная и информативная обработка ошибок на клиенте и сервере, а также расширение валидации пользовательских данных.
4. *Управление доступом:* не все административные функции защищены на уровне прав доступа, что требует внедрения более строгой ролевой модели.
5. *Масштабируемость:* при увеличении объёма данных и числа пользователей потребуется оптимизация запросов и архитектуры для обеспечения высокой производительности.

Будущее развитие решения

Для дальнейшего повышения ценности и удобства сервиса возможны следующие направления развития:

1. *Расширение функциональности для организаторов:* добавление инструментов массового оповещения участников, интеграция с картографическими сервисами, расширенные отчёты по мероприятиям.
2. *Улучшение пользовательского опыта:* внедрение адаптивной верстки, расширение возможностей фильтрации и поиска, улучшение визуализации статистики.
3. *Интеграция с внешними сервисами:* возможность авторизации через соцсети, интеграция с городскими порталами и системами оповещения.
4. *Развитие системы геймификации:* введение достижений, рейтингов, наград для активных участников и организаторов.
5. *Безопасность и аудит:* внедрение расширенного логирования действий пользователей, улучшение механизмов резервного копирования и восстановления данных.

Таким образом, созданный сервис является современной платформой для организации и поддержки городских субботников, обладающей потенциалом для дальнейшего развития и масштабирования.

ПРИЛОЖЕНИЯ

Документация по сборке и развертыванию приложения

1. Склонировать репозиторий: [nosql1h25-cleanday](#).
2. Перейти в корневую директорию проекта.
3. В .env файле в корне проекта ввести свои значения пароля базы данных *ARANGO_ROOT_PASSWORD* и секретного ключа *SECRET_KEY*.
4. Собрать docker-контейнер командой `*docker compose build --no-cache*`.
5. Запустить docker-контейнер с приложением командой `*docker compose up*`
6. Открыть веб-приложение в браузере по адресу: *http://localhost:8080*.

Инструкция для пользователя

Регистрация и вход

1. На главной странице нажмите кнопку «Зарегистрироваться»;
2. Заполните форму регистрации, указав имя, фамилию, логин, пароль и прочую информацию;
3. После регистрации войдите в систему, используя свой логин и пароль;
4. Для последующего входа используйте кнопку «Войти» на главной странице.

Просмотр и участие в субботниках

1. После входа в систему перейдите в раздел «Субботники»;
2. Просматривайте список доступных субботников с возможностью фильтрации и поиска;
3. Нажмите на интересующий субботник для просмотра подробной информации;
4. Для участия в субботнике нажмите на соответствующую кнопку и выберите статус участия:
 - «Точно пойду» - подтверждение участия;
 - «Опоздаю» - подтверждение участия с указанием опоздания;

- «Возможно пойду» - предварительное подтверждение;
 - «Не пойду» - отказ от участия;
5. Ознакомьтесь с требованиями к участию и подтвердите их соблюдение.

Создание собственного субботника

1. В верхней панели нажмите кнопку «Создание субботника»;
2. Заполните форму создания субботника:
 - Название;
 - Местоположение (адрес);
 - Организация;
 - Площадь территории (м^2);
 - Дата и время начала и окончания;
 - Рекомендуемое количество участников;
 - Описание мероприятия;
 - Теги мероприятия (уборка мусора, посадка растений и т.д.);
 - Требования к участникам;
3. Нажмите «Создать» для публикации вашего субботника.

Управление профилем

1. Для просмотра личного профиля нажмите на аватар в правом верхнем углу;
2. В профиле вы увидите:
 - Личную информацию (имя, фамилия, город и т.д.);
 - Текущий уровень и опыт;
 - Статистику участия в субботниках;
 - Убранную территорию (в м^2);
3. Для редактирования профиля нажмите на кнопку «Редактировать»;
4. Для просмотра организованных вами субботников нажмите кнопку «Организовано»;
5. Для просмотра субботников, в которых вы участвовали, нажмите кнопку

«Участие».

Просмотр статистики

1. Перейдите в раздел «Статистика» через главное меню;
2. На странице статистики вы увидите:
 - Общее количество пользователей в системе;
 - Количество пользователей с опытом участия в субботниках;
 - Общее количество субботников;
 - Количество прошедших субботников;
 - Общую площадь убранной территории;
3. Для экспорта статистики нажмите кнопку «Экспорт»;
4. Для импорта статистики нажмите кнопку «Импорт».

Завершение субботника (для организаторов)

1. После завершения мероприятия перейдите на страницу субботника;
2. Нажмите кнопку «Завершить субботник»;
3. Отметьте присутствие участников;
4. Добавьте результаты (например, «Собрано 10 мешков мусора»);
5. Загрузите фотографии с мероприятия;
6. Нажмите «Завершить» для подтверждения.

Выход из системы

1. Для выхода нажмите на значок выхода в правом верхнем углу;
2. Подтвердите выход из системы в появившемся диалоговом окне.

ЛИТЕРАТУРА

1. Ссылка на GitHub репозиторий. – [Электронный ресурс]. – URL: <https://github.com/moevm/nosql1h25-cleanday>.
2. Ссылка на макеты в Figma. – [Электронный ресурс]. – URL: <https://www.figma.com/design/RR5x78BQuwJct7bSUATjYD%D0%BE%D0%BF%D1%80%D0%BF%D0%BF%D0%BE%D0%BE%D1%80%D0%BF%D0%BF%D0%BE%D0%BF%D0%BE-%D0%B8-%D0%BD%D0%BE%D1%81%D0%BA%D0%BB?node-id=6035-100060>.
3. Документация React. – [Электронный ресурс]. – URL: <https://react.dev/reference/react> (дата обращения 01.06.2025).
4. Документация Material UI. – [Электронный ресурс]. – URL: <https://mui.com/material-ui/all-components> (дата обращения 01.06.2025).
5. Документация ArangoDB. – [Электронный ресурс]. – URL: <https://arangodb.com/> (дата обращения 01.06.2025).
6. Документация python-arango. – [Электронный ресурс]. – URL: <https://docs.python-arango.com/en/main/index.html> (дата обращения 01.06.2025).