Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных Центр программной инженерии

ОТЧЕТ

к лабораторной работе № 3 по дисциплине:

|  |
| --- |
| «Проектирование информационных систем» |
| Технический проект |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСТб-19-2 |  |  |  | Комогорцева Ю.В. |
| Проверил: | шифр группы |  | подпись |  | Фамилия ИО  Серышева И.А. |
|  | должность |  | подпись |  | Фамилия ИО |

Иркутск. 2022 г.

Содержание

[1 Введение 3](#_Toc121495485)

[2 Функциональность ПО 3](#_Toc121495486)

[3 Сценарии использования 4](#_Toc121495487)

[4 Постановка задачи 7](#_Toc121495488)

[Задача – Создание проекта 7](#_Toc121495489)

[Спецификация варианта использования «Подача заявки на создание проекта». 7](#_Toc121495490)

[Спецификация варианта использования «Рассмотрение заявки на создание». 8](#_Toc121495491)

[Требования к реализации задачи 9](#_Toc121495492)

[Задача – Участие в проекте 10](#_Toc121495493)

[Спецификация варианта использования «Создание команды для участия в проекте». 10](#_Toc121495494)

[Спецификация варианта использования «Подача заявки на вступление в команду для реализации проекта». 11](#_Toc121495495)

[Спецификация варианта использования «Рассмотрение заявки на участие в проекте». 12](#_Toc121495496)

[Спецификация варианта использования «Подведение итогов участия в проекте». 13](#_Toc121495497)

[Требования к реализации задачи 14](#_Toc121495498)

[Задача – Участие в хакатоне 15](#_Toc121495499)

[Задача – Проведение мероприятия 16](#_Toc121495500)

[5 Архитектура системы 17](#_Toc121495501)

[6 Логическая модель данных 20](#_Toc121495502)

[7 Физическая модель данных 21](#_Toc121495503)

[8 Пользовательский интерфейс 21](#_Toc121495504)

[Вывод 23](#_Toc121495505)

# Введение

Перед описанием бизнес-процессов и дальнейшим проектированием в системе, стоит упомянуть о необходимых видах пользователей в системе:

* Гость
* Участник – авторизованный пользователь в системе
* Участник команды – участник с правами на управление командой при реализации проекта.
* Администратор – авторизованный пользователь в системе с правами администратора для управления системой

**Модуль с проектами и модуль с пользователями является на данной итерации приоритетными, а именно функции:**

* **Просмотр и фильтрация проектов**
* **Подача заявки на создание проекта**
* **Рассмотрение заявки на создание проекта**
* **Авторизация пользователей**

# 1 Функциональность ПО

В таблице ниже определены функции, которые необходимо реализовать в системе.

|  |  |
| --- | --- |
| **Подсистема** | **Функции** |
| Проекты | Просмотр и фильтрация списка проектов |
| Просмотр информации о проекте |
| Создание проекта |
| Подача заявки на участие в проекте |
| Рассмотрение заявки на участие в проекте автором проекта |
| Фиксация результатов деятельности команды по проекту |
| Рулетка проектов |
| Хакатоны | Просмотр и фильтрация списка хакатонов |
| Просмотр информации о хакатоне |
| Подбор участников команды |
| Мероприятия | Просмотр списка конкурсов |
| Создание конкурса |
| Участие в конкурсе |
| Просмотр результатов конкурса |
| Голосование по результатам конкурса |
| Закрытие конкурса |
| Личный кабинет | Регистрация |
| Авторизация |
| Заполнение портфолио |
| Система достижений |
| Подписка на рассылку |
| Административная панель | Просмотр списка поданных на рассмотрение проектов |
| Рассмотрение проекта, который хотят создать пользователи |
| Создание хакатона вручную |
| Просмотр статистики площадки |
| Управление проектами |
| Управление источниками для агрегатора хакатонов |
| Рассмотрение конкурса, который хотят создать пользователи |
| Закрытие конкурса |

# 2 Сценарии использования

Было выделено 4 процесса, с помощью которых пользователь сможет достигнуть тех или иных целей, которые возможны в рамках системы «Биржа опыта»:

* создание проекта;
* участие в проекте;
* участие в хакатоне;
* проведение мероприятия.

Далее с помощью сценариев использования в нотации UML показан уровень общих процессов, а также детализация каждого из них. На рисунке 1 показаны общие сценарии, а на рисунках 2,3,4,5 показана детализация каждого сценария.



Рисунок 1 – Сценарии использования в системе «Биржа опыта»

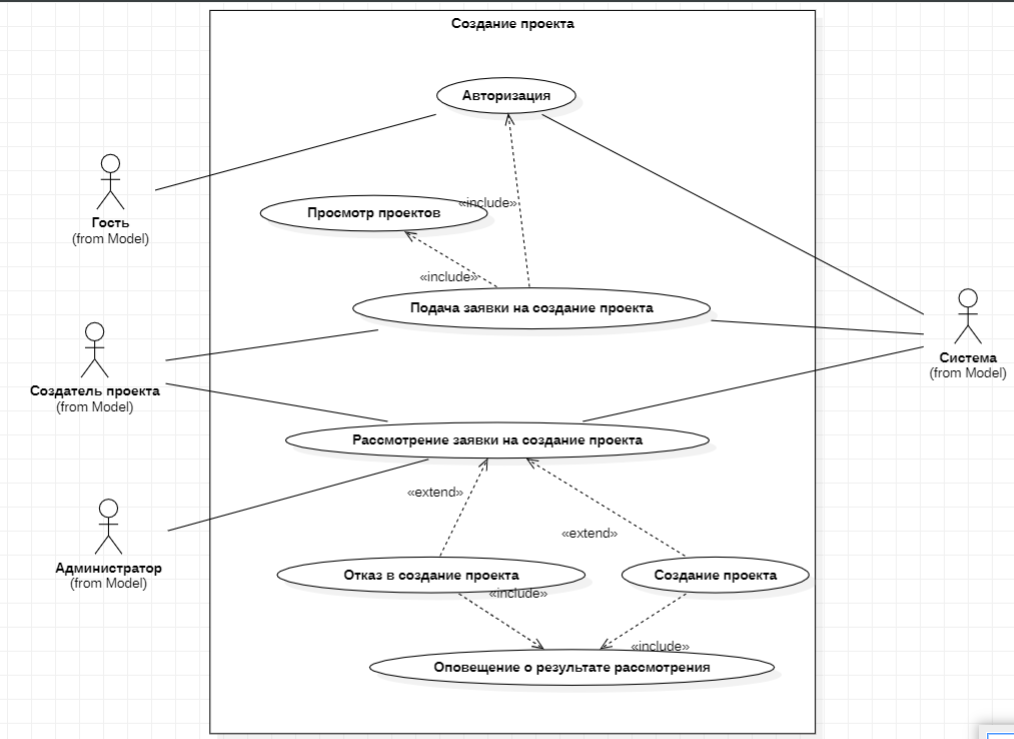


Рисунок 2 – Сценарий варианта использования «Создание проекта»

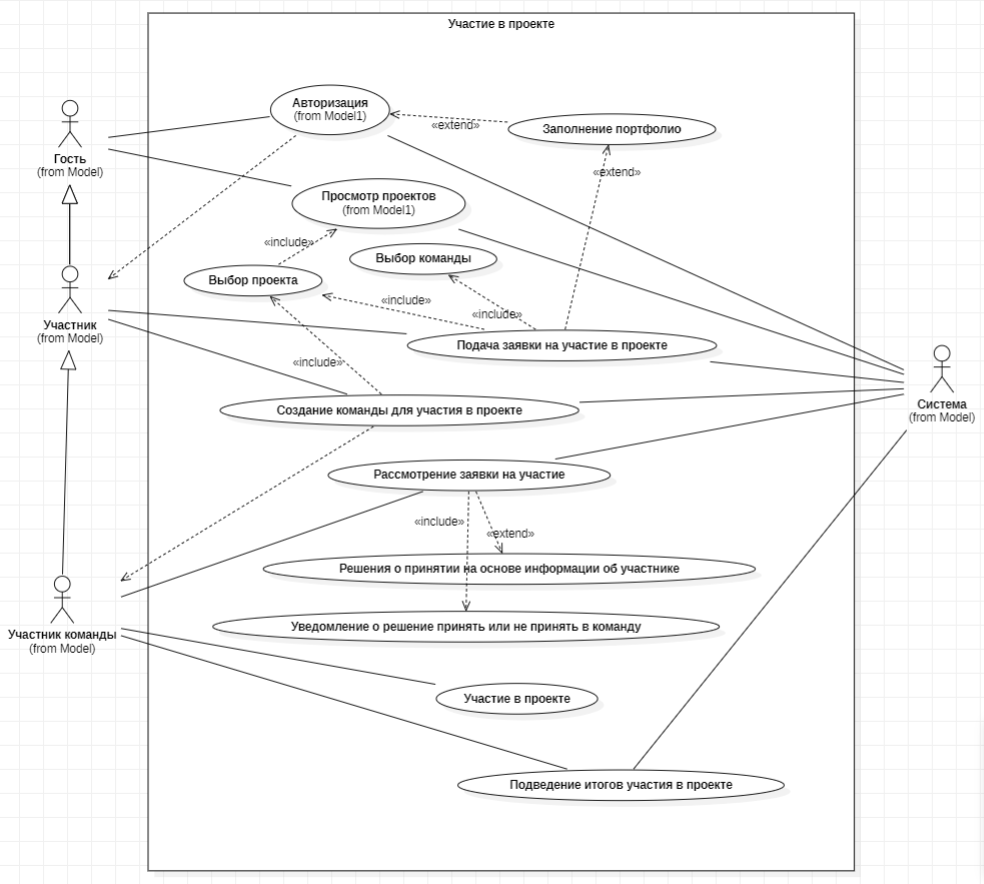


Рисунок 3 – Сценарий варианта использования «Участие в проекте»

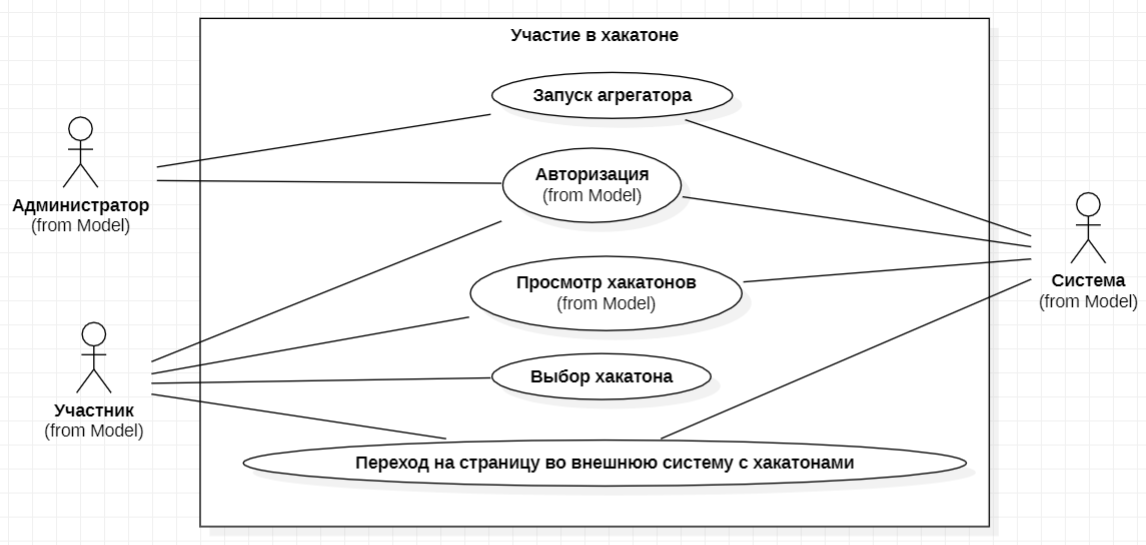


Рисунок 4 – Сценарий варианта использования «Участие в хакатоне»



Рисунок 5 – Сценарий варианта использования «Проведение мероприятия»

# 3 Постановка задачи

Описаны постановки задач и подзадач с помощью спецификаций вариантов использования. В процессе разработки необходимо ориентироваться на данные сценарии, ~~а именно~~ они показывают работу системы при определённых действиях пользователя.

## Задача – Создание проекта

Сценарий варианта использования «Создание проекта» показан на рисунке 2. Система должна позволять создавать проект только авторизованному пользователю. Создание проекта предполагает следующую последовательность действий: авторизованный пользователь подает заявку на создание проекта из страницы с проектами, затем администратор системы проверяет заявку и принимает решение о создании или отказе в создании проекта. Данная задача предполагает 2 пользователей (авторизованный участник и администратор) и реализацию как в серверной, так и в клиентской части.

### Спецификация варианта использования «Подача заявки на создание проекта».

**Цель:** подать заявку на создание проекта

**Активные субъекты:** Участник.

**Краткое описание:** Участник системы создает заявку на создание проекта.

**Предусловия:** Участник должен авторизоваться в системе.

**Постусловия:** должен быть выполнен вариант использования *«Рассмотрение заявки на создание проекта»*.

**Основной поток событий:**

1. Система публикует страницу с проектами, на которой находится кнопка «Создать проект». Если пользователь не авторизован после нажатия кнопки система предлагает авторизоваться.
2. Активный субъект нажимает кнопку создания проекта.
3. Система показывает пользователю информацию о том, что ему нужно создать заявку, которую рассматривает администратор проекта и публикует форму с полями: название, тип проекта, описание, фотографии, тэги проекта, максимальный размер команды. Поле тип проекта имеет значения «учебный» или «уникальный». Тэги для проекта выбираются из уже существующих в системе.
4. Активный субъект заполняет поля формы.
5. Система проверяет данные в форме на валидность.
6. Активный субъект нажимает кнопку «Отправить заявку».
7. Система уведомляет пользователя об успешной или не успешной отправке заявки.
8. Система завершает работу варианта использования.

**Альтернативные потоки событий:**

1. Система демонстрирует активному субъекту о неправильном заполнении полей в форме.
2. Система демонстрирует активному субъекту сообщение: «Ошибка при отправке заявки».

**Сценарий обработки ошибок:** при возникновении ошибки отправки заявки на создание поля формы сохраняют свои данные и позволяют пользователю заново отправить заявку.

### Спецификация варианта использования «Рассмотрение заявки на создание».

**Цель:** рассмотреть заявку на создание проекта

**Активные субъекты:** администратор, участник.

**Краткое описание:** Администратор рассматривает поданные заявки на создание проекта и принимает решение. В конце система должна уведомить участника о результате рассмотрения заявки.

**Предусловия:** Администратор должен быть авторизован в системе для просмотра поданных заявок и принятия решения по ним. Участник должен авторизоваться в системе, чтобы увидеть результат заявки.

**Постусловия:** нет.

**Основной поток событий:**

1. Система публикует страницу для управления, на которой находятся дашборды. Один из них отвечает за показ статистики по поданным и не рассмотренным заявкам.
2. Администратор нажимает на элемент дашборда со статистикой поданных заявок.
3. Система показывает администратору страницу с заявками на создание проекта. Каждая заявка имеет статус: отклонено, создано, не рассмотрено. Заявки показываются в таблице и если у них статус не рассмотрено, то строка таблицы имеет кнопки «создать» и «отклонить».
4. Администратор нажимает на не рассмотренную заявку.
5. Система открывает страницу проекта с данными из заявки.
6. Администратор читает идею проекта и принимает решение создавать проект или нет.
7. Администратор возвращается на страницу списка заявок.
8. Если администратор принимает решение о создании проекта, то должен нажать на кнопку «Создать». Если администратор не хочет создавать предложенный проект, то ему необходимо нажать на кнопку «отклонить».
9. После нажатия любой из кнопок «создать» и «отклонить» система меняет статус проекта и уведомляет участника о результате рассмотрения заявки по почте.
10. Система завершает работу варианта использования.

**Альтернативные потоки событий:** нет.

**Сценарий обработки ошибок:** нет.

### Требования к реализации задачи

Запросы, которые необходимо реализовать в REST API:

* /api/projects GET Получение списка проектов
* /api/project POST Создание проекта
* /api/admin GET Получение информации для дашбордов
* /api/admin/project/request GET Получение списка заявок, поданных на создание проекта
* /api/project/:id POST Изменение статуса проекта при рассмотрении проекта

Страницы, которые необходимо реализовать в web application:

* Страница «Проекты» с кнопкой создания проекта
* Страница «Подача заявки на создание проекта» с полями
* Страница «Управление системой» с дашбордами
* Страница «Заявки на создание проекта» со списком заявок

## Задача – Участие в проекте

Сценарий варианта использования «Участие в проекте» показан на рисунке 3. Система должна позволять участвовать в проекте только для авторизованного пользователя. Участие в проекте состоит из следующих подзадач: подача заявки на участие в проекте, рассмотрение заявки на участие в проекте, подведение итогов проекта. В задаче участвуют двое пользователей: участник, который хочет попасть в команду для реализации проекта, и участники, уже состоящие в команде. Реализация предполагается как в серверной, так и в клиентской части.

### Спецификация варианта использования «Создание команды для участия в проекте».

**Цель:** создание команды для участия в учебном проекте.

**Активные субъекты:** участник, который хочет создать команду.

**Краткое описание:** участник системы создает команду.

**Предусловия:** участник должен авторизоваться в системе.

**Постусловия:** может быть выполнен вариант использования *«Рассмотрение заявки на участие»*.

**Основной поток событий:**

1. Система публикует страницу с проектами.
2. Активный субъект выбирает проект для участия, открывая проект на отдельной странице с помощью кнопки «Подробнее» на карточке проекта.
3. Система публикует страницу с информацией о выбранном проекте. Если проект учебный, то на странице отображается список команд, которые исходя из статусов участия команды в проекте либо содержат, либо нет кнопку «Подать заявку». Страница проекта должна содержать кнопку «Создать команду».
4. Активный субъект нажимает кнопку «Создать команду».
5. Система показывает пользователю информацию о том, что ему нужно создать команду и публикует форму с полями: название команды, количество участников и роли в команде. Роли в команде выбираются из готового списка, который может редактировать только администратор.
6. Активный субъект заполняет поля формы.
7. Система проверяет данные в форме на валидность.
8. Активный субъект нажимает кнопку «Создать команду».
9. Система уведомляет пользователя об успешном или не успешном создании команды.
10. Система завершает работу варианта использования.

**Альтернативные потоки событий:**

1. Система демонстрирует активному субъекту о неправильном заполнении полей в форме.
2. Система демонстрирует активному субъекту сообщение: «Ошибка при создании команды».

**Сценарий обработки ошибок:** при возникновении ошибки о создании команды, формы сохраняют свои данные и позволяют пользователю заново отправить заявку.

### Спецификация варианта использования «Подача заявки на вступление в команду для реализации проекта».

**Цель:** подать заявку на создание проекта

**Активные субъекты:** участник, который хочет попасть в команду для реализации проекта.

**Краткое описание:** участник системы создает заявку на участие в проекте в рамках выбранной команды.

**Предусловия:** участник должен авторизоваться в системе.

**Постусловия:** должен быть выполнен вариант использования *«Рассмотрение заявки на участие»*.

**Основной поток событий:**

1. Система публикует страницу с проектами.
2. Активный субъект выбирает проект для участия, открывая проект на отдельной странице с помощью кнопки «Подробнее» на карточке проекта.
3. Система публикует страницу с информацией о выбранном проекте. Если проект учебный, то на странице отображается список команд, которые исходя из статусов участия команды в проекте либо содержат, либо нет кнопку «Подать заявку». Если проект уникальный, то на странице проекта указана только одна команда, у которой тоже может быть или не быть кнопка «Подать заявку».
4. Активный субъект выбирает команду и нажимает кнопку «Подать заявку».
5. Система показывает пользователю информацию о том, что ему нужно создать заявку, которую рассматривают участники выбранной команды и публикует форму с полями: желаемая роль в проекте, резюме в виде файла или ссылки. Желаемая роль в проекте выбирается из уже созданных и не занятых ролей в команде.
6. Активный субъект заполняет поля формы.
7. Система проверяет данные в форме на валидность.
8. Активный субъект нажимает кнопку «Отправить заявку».
9. Система уведомляет пользователя об успешной или не успешной отправке заявки.
10. Система завершает работу варианта использования.

**Альтернативные потоки событий:**

1. Система демонстрирует активному субъекту о неправильном заполнении полей в форме.
2. Система демонстрирует активному субъекту сообщение: «Ошибка при отправке заявки».

**Сценарий обработки ошибок:** при возникновении ошибки отправки заявки на создание поля формы сохраняют свои данные и позволяют пользователю заново отправить заявку.

### Спецификация варианта использования «Рассмотрение заявки на участие в проекте».

**Цель:** рассмотреть заявку на участие в проекте в определенной команде

**Активные субъекты:** участник, который хочет попасть в команду для реализации проекта, и участники, уже состоящие в команде.

**Краткое описание:** Участники, уже состоящие в проекте, рассматривают поданные заявки на участие в их команде и принимают решение. В конце система должна уведомить участника, который хочет попасть в команду для реализации проекта, о результате рассмотрения заявки.

**Предусловия:** Участники должны быть авторизованы в системе для просмотра поданных заявок и принятия решения по ним. Участник должен авторизоваться в системе, чтобы увидеть результат заявки.

**Постусловия:** должен быть выполнен вариант использования **«**Подведение итогов участия в проекте**».**

**Основной поток событий:**

1. Система публикует страницу с командами, в которых состоит участник, в рамках личного кабинета. Каждая команда имеет кнопку «Подробнее» и значок, появляющийся при наличии уведомления.
2. Участник, уже состоящий в выбранной команде, нажимает на кнопку «Подробнее» определенной команды.
3. Система показывает участнику, состоящему в команде, страницу с информацией о команде, ее участниках и желающих участвовать, а также сведениями об участии команды в проекте. Каждая заявка на участие имеет статус: отклонена, принята, не рассмотрена. Заявки показываются в таблице с полями: ФИО, желаемая роль, файл или ссылка на резюме. Если у заявки статус не рассмотрена, то строка таблицы имеет кнопки «Принять» и «Отклонить» заявку.
4. Участник, состоящий в команде, принимает решение о принятии пользователя в команду или нет. Участнику необходимо нажать на кнопку «Принять» или «Отклонить» заявку. Из всех, состоящих в команде участников, достаточно ~~сделать это~~ лишь одному.
5. После нажатия любой из кнопок «Принять» или «Отклонить» система меняет статус заявки на участие и уведомляет участника о результате рассмотрения заявки по почте.
6. Система завершает работу варианта использования.

**Альтернативные потоки событий:** нет.

**Сценарий обработки ошибок:** нет.

### Спецификация варианта использования «Подведение итогов участия в проекте».

**Цель:** зафиксировать результат участия в проекте

**Активные субъекты:** участники, уже состоящие в команде.

**Краткое описание:** Один из участников, уже состоящих в команде, фиксирует результат об участии в проекте.

**Предусловия:** Участник должен авторизоваться в системе, чтобы появилась возможность подведения итогов.

**Постусловия:** нет.

**Основной поток событий:**

1. Система публикует страницу с командами, в которых состоит участник, в рамках личного кабинета. Каждая команда имеет кнопку «Подробнее» и значок, появляющийся при наличии уведомления.
2. Участник, уже состоящий в выбранной команде, нажимает на кнопку «Подробнее» определенной команды.
3. Система показывает участнику, состоящему в команде, страницу с информацией о команде, ее участниках и желающих участвовать, а также сведениями об участии команды в проекте. Если команда еще не подводила итоги, то на странице команды присутствует кнопка «Сформировать итоги участия»
4. Участник, состоящий в команде, нажимает кнопку «Сформировать итоги участия»
5. Система публикует страницу ~~для~~ с формой подведения итогов со следующими полями: ссылка на решение, изображения результатов, отзыв в виде текста.
6. Активный субъект заполняет поля формы.
7. Система проверяет данные в форме на валидность.
8. Активный субъект нажимает кнопку «Сохранить».
9. Система уведомляет пользователя об успешном или не успешном сохранении результатов команды.
10. Система завершает работу варианта использования.

**Альтернативные потоки событий:** нет.

1. Система демонстрирует активному субъекту о неправильном заполнении полей в форме.
2. Система демонстрирует активному субъекту сообщение: «Ошибка при сохранении итогов».

**Сценарий обработки ошибок:** при возникновении ошибки о сохранении итогов, формы сохраняют свои данные и позволяют пользователю заново сохранить итоги.

### Требования к реализации задачи

Что необходимо реализовать в REST API:

* /api/projects/:id/teams POST Создание команды для участия в проекте
* /api/users/:id/teams GET Получение списка команд пользователя
* /api/users/:id/teams/:id POST Изменение статуса у заявки участника на выполнение проекта в определенной команде
* /api/events/:id/result POST Подведение итогов участия в проекте

Что необходимо реализовать в web application:

* Страница «Проекты» с кнопкой создания проекта
* Страница «Проект» с кнопкой создания команды и списком команд, выполняющих проект
* Страница «Команды» в личном кабинете, со списком команд, в которых состоит пользователь
* Страница «Команда» с информацией о команде и ее участниках, так же с поданными на вступление заявками
* Страница «Подача заявки на вступление в команду» с формой ввода
* Страница «Подведение итогов участия в проекте» с формой ввода

## Задача – Участие в хакатоне

Сценарий варианта использования «Участие в хакатоне» показан на рисунке 4. Система должна показывать хакатоны и переходить на сайты хакатонов всех пользователей. Участие в хакатоне не детализируется, потому что состоит из следующих действий пользователя: просмотр списка хакатонов, выбор интересующего, открытие страницы с информацией о хакатоне, нажатие на кнопку «Записаться на хакатон» и система открывает страницу хакатона на сайте организаторов. Реализация предполагается как в серверной, так и в клиентской части. В дальнейшем планируется добавить возможность собирать команду на хакатон в разрабатываемой системе, пока что на данной итерации проекта участие в хакатоне предполагает только переход в другую ИС.

Отдельно, в описание задачи необходимо определить откуда будут поступать хакатоны. Администратор может создать хакатон вручную, для этого в клиентской части необходимо разработать форму ввода, а на серверной стороне необходимо создать POST запрос для создания хакатона. Однако, предполагается что в следующих итерациях разработки будет создан агрегатор проектов. Определим первоначальные требования к агрегатору:

* Агрегатор должен быть реализован отдельным приложением, которое будет запускаться по инициативе из приложения REST API.
* Агрегатор необходимо реализовать на Python, с помощью библиотек:
* requests — для стягивания страниц из интернета
* beautifulsoup4 — для парсинга страниц
* sqlalchemy — для работ с БД
* Источниками для агрегатора должны являться:
* <https://leader-id.ru/>
* <https://hackathons.pro/>
* <https://www.хакатоны.рф/>
* <https://codenrock.com/>
* <https://pgenesis.ru/events>
* <https://devpost.com/>
* <https://хакатоны.рус/>
* Агрегатор должен сам напрямую работать с базой данных
* Агрегатор стягивает данные с сайтов-источников с помощью двух методов: через обращение к API и через парсинг HTML кода.

## Задача – Проведение мероприятия

Сценарий варианта использования «Проведение мероприятия» показан на рисунке 5. Система должна позволять участникам создавать мероприятия, которые, участвовать в проекте только для авторизованного пользователя. Участие в проекте состоит из следующих подзадач: подача заявки на участие в проекте, рассмотрение заявки на участие в проекте, подведение итогов проекта. В задаче участвуют двое пользователей: участник, который хочет попасть в команду для реализации проекта и участники, уже состоящие в команде. Реализация предполагается как в серверной, так и в клиентской части.

# **4 Архитектура системы**

Архитектура системы показана с помощью модели [C4](https://c4model.com/). Модель C4 была создана как способ помочь командам разработчиков программного обеспечения описывать и передавать архитектуру программного обеспечения как во время предварительных сессий проектирования, так и при ретроспективном документировании существующей кодовой базы.

Для реализации системы необходимо создавать именно web-приложение. ~~Благодаря этому, возможно~~ что позволит, обеспечить доступ с различных устройств из любой точки мира. Исходя из выбранного вида программного обеспечения, для реализации системы была выбрана архитектура «Клиент-Сервер». Это такая архитектура, которая состоит из двух физических машин в качестве сервера и клиента. Клиент содержит программу-клиент, которая предоставляет для пользователя интерфейс, посылает запросы на сервер и получает ответы на запросы. В то же время, сервер хранит данные и содержит программу, которая отвечает за логику обработки данных из поступивших запросов. Преимущества данной архитектуры: обеспечивает изоляцию операций управления данными, обработки данных и операций представления; легко модифицировать приложения; хорошая производительность и масштабируемость.

Модель состоит из 4х уровней, на данном этапе проектирования будут спроектированы 3. Четвертый уровень проектирования подразумевает диаграмму кода, будет проще нарисовать после частичной реализации.

Модель была разработана в сервисе [IcePanel](https://s.icepanel.io/ubakh5vkTlUYfh/1jwZ).

На рисунке 1 изображена диаграмма контекста системы , которая показывает ~~нам~~ что «Биржа опыта» должна взаимодействовать с системой Email для отправки писем пользователю, а также сайтами с хакатонами для агрегирования данных.

На рисунке 2 изображена схема контейнеров, которая отображает ~~части~~ системы «Биржа опыта». Система должна состоять из веб-приложения(клиент), REST API (сервер), агрегатора, а также самой БД. Веб приложение должно взаимодействовать с REST API через HTTP протокол в JSON формате.

На рисунке 3 показана схема компонентов веб-приложения: страницы, компоненты,хранилище и роутер.

На рисунке 4 находится схема компонентов REST API, а именно модулей: projects, hackatons, events, users. Все модули должны взаимодействовать с БД через библиотеку TypeORM. Так же модуль hackatons управляет агрегатором. С почтой взаимодействует модуль users.

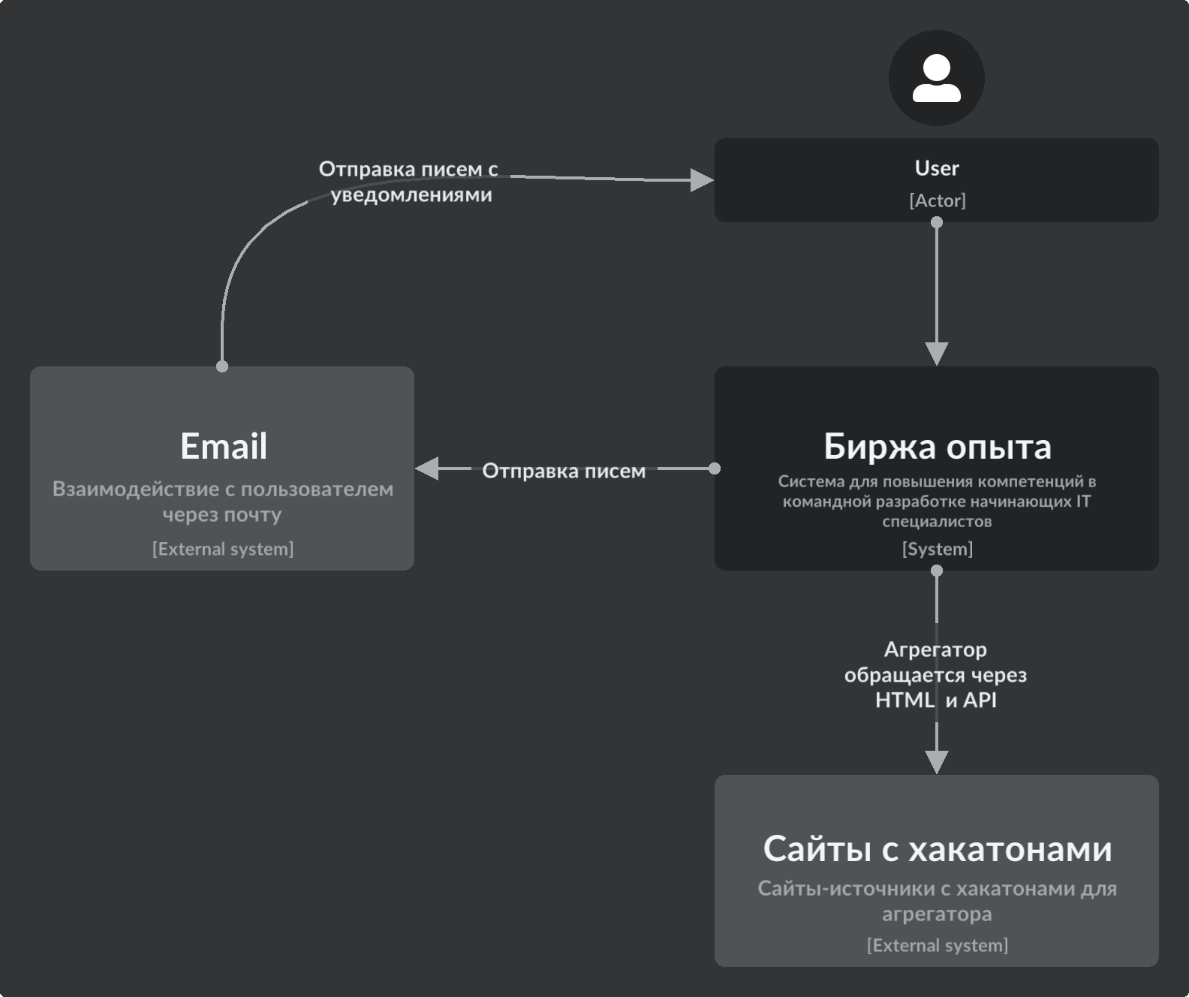


Рисунок 6 – Уровень 1. Схема системного контекста

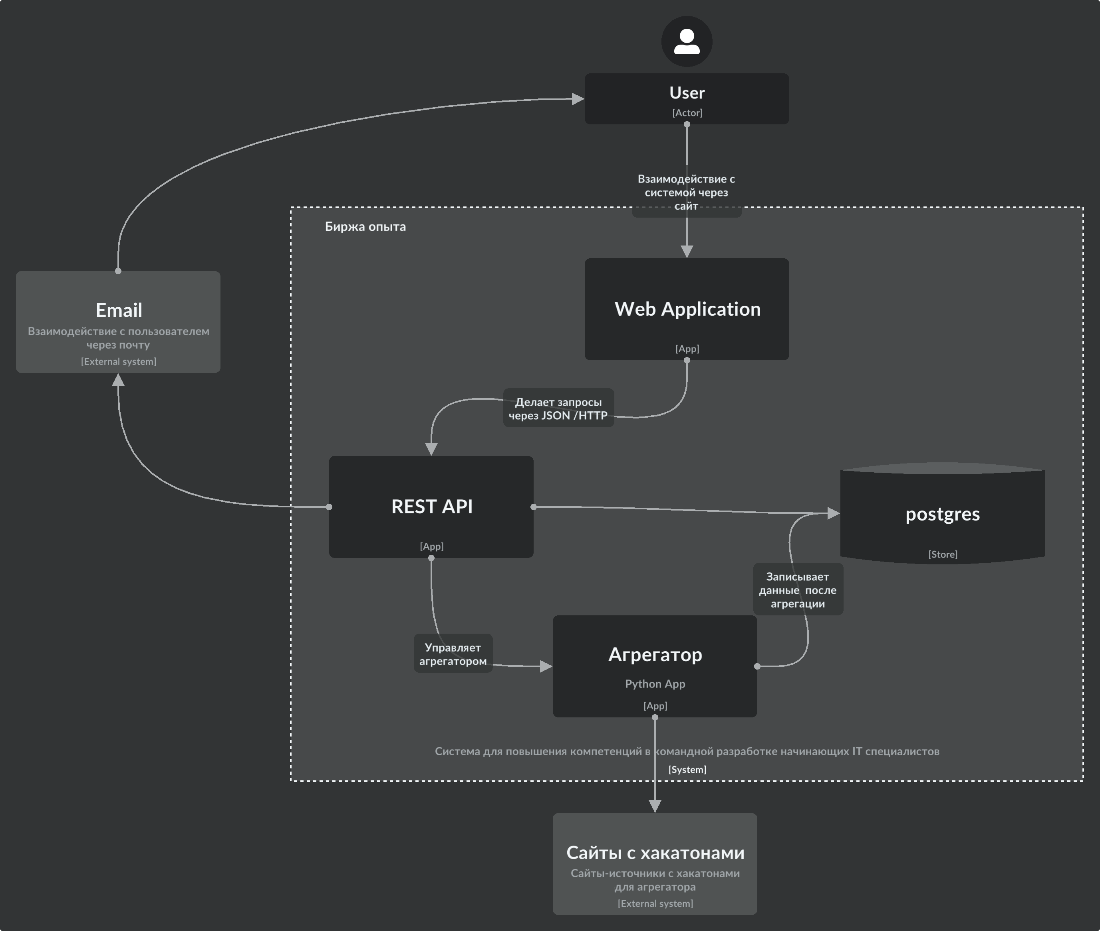


Рисунок 7 – Уровень 2. Схема контейнеров

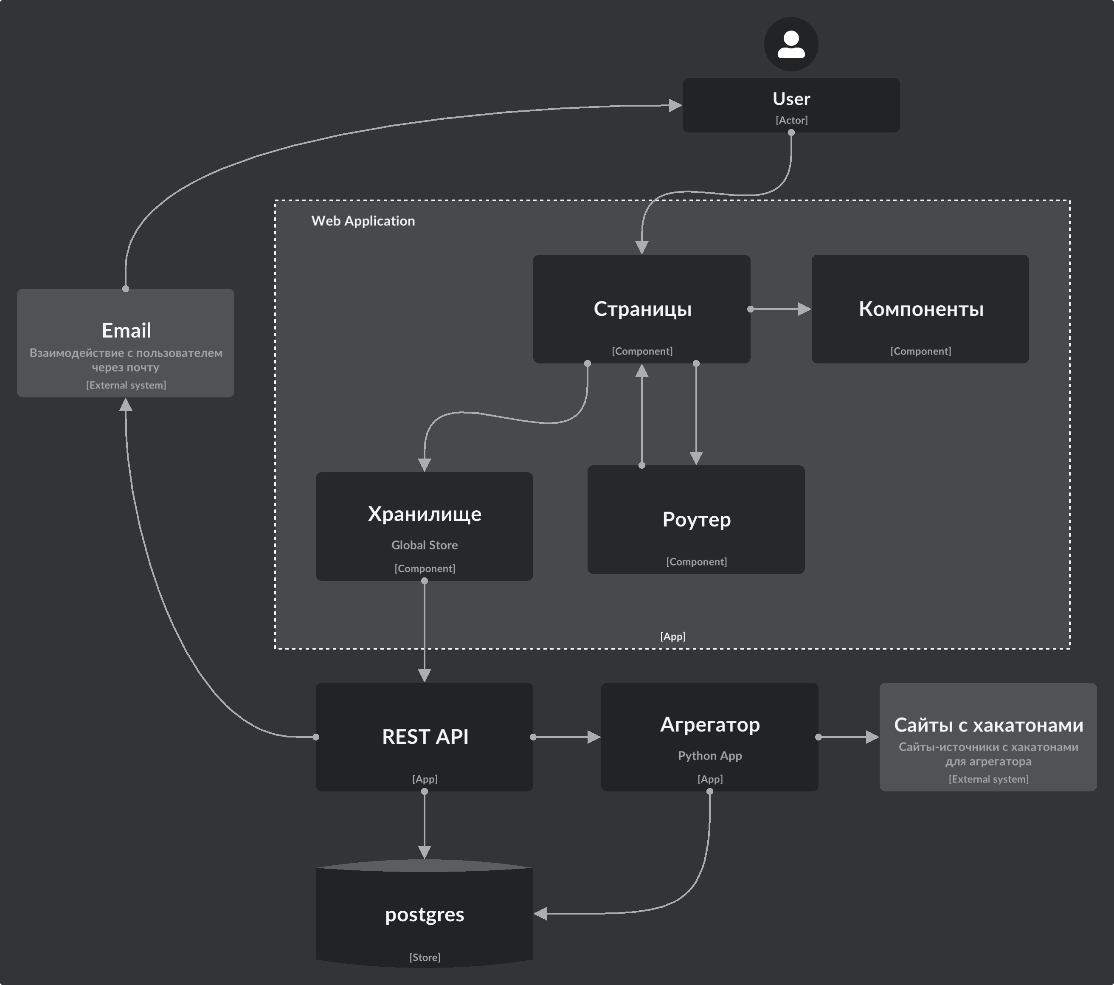


Рисунок 8 – Уровень 3. Схема компонентов Web Application

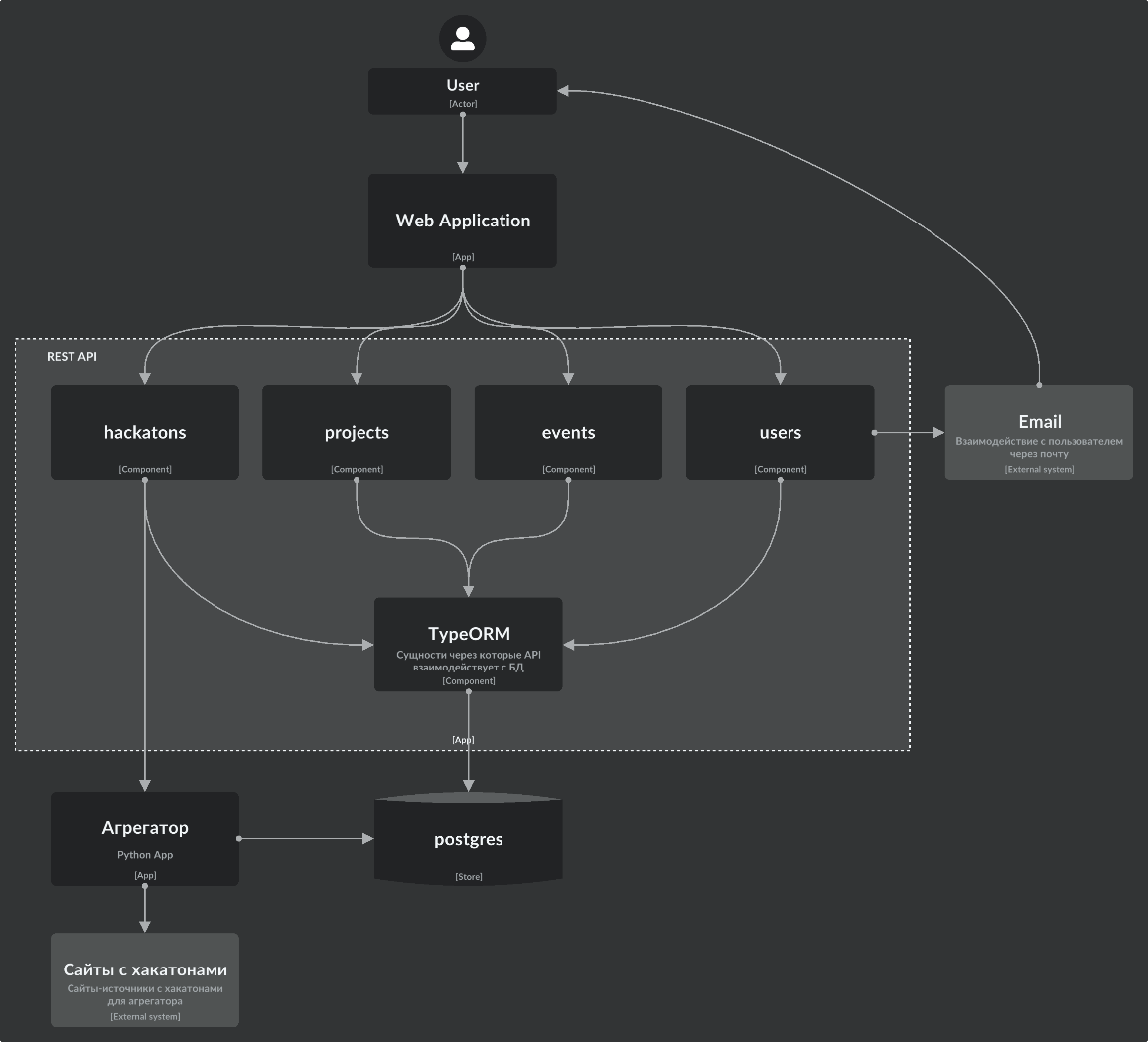


Рисунок 9 – Уровень 3. Схема компонентов REST API

# **5 Логическая модель данных**

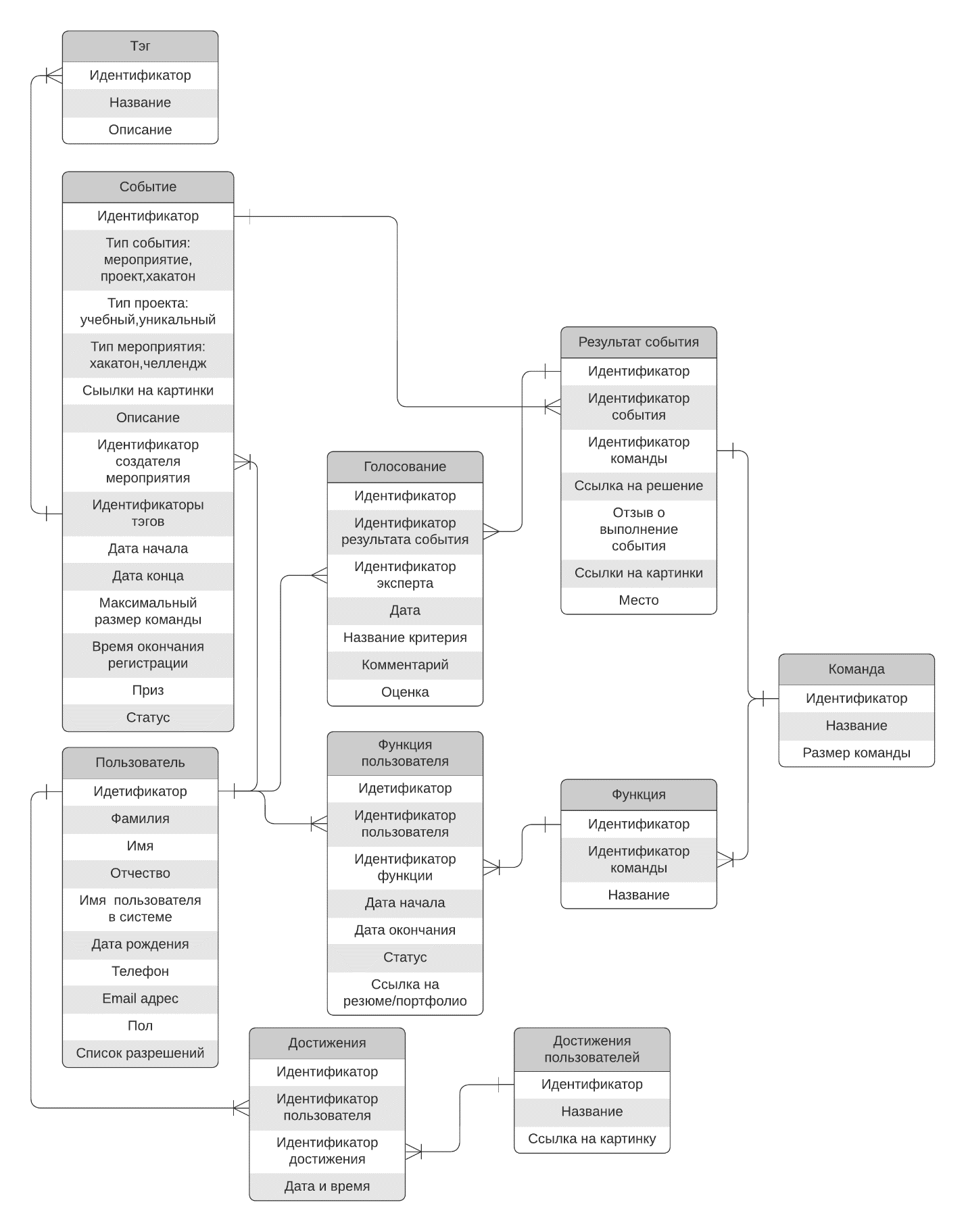


Рисунок 10 – Логическая модель данных

# **6** **Физическая модель данных**

Физическая модель данных была спроектирована с учетом СУБД PostgreSQL.

[Ссылка на диаграмму](https://drawsql.app/teams/ne-izvestno/diagrams/birzha-opyta)

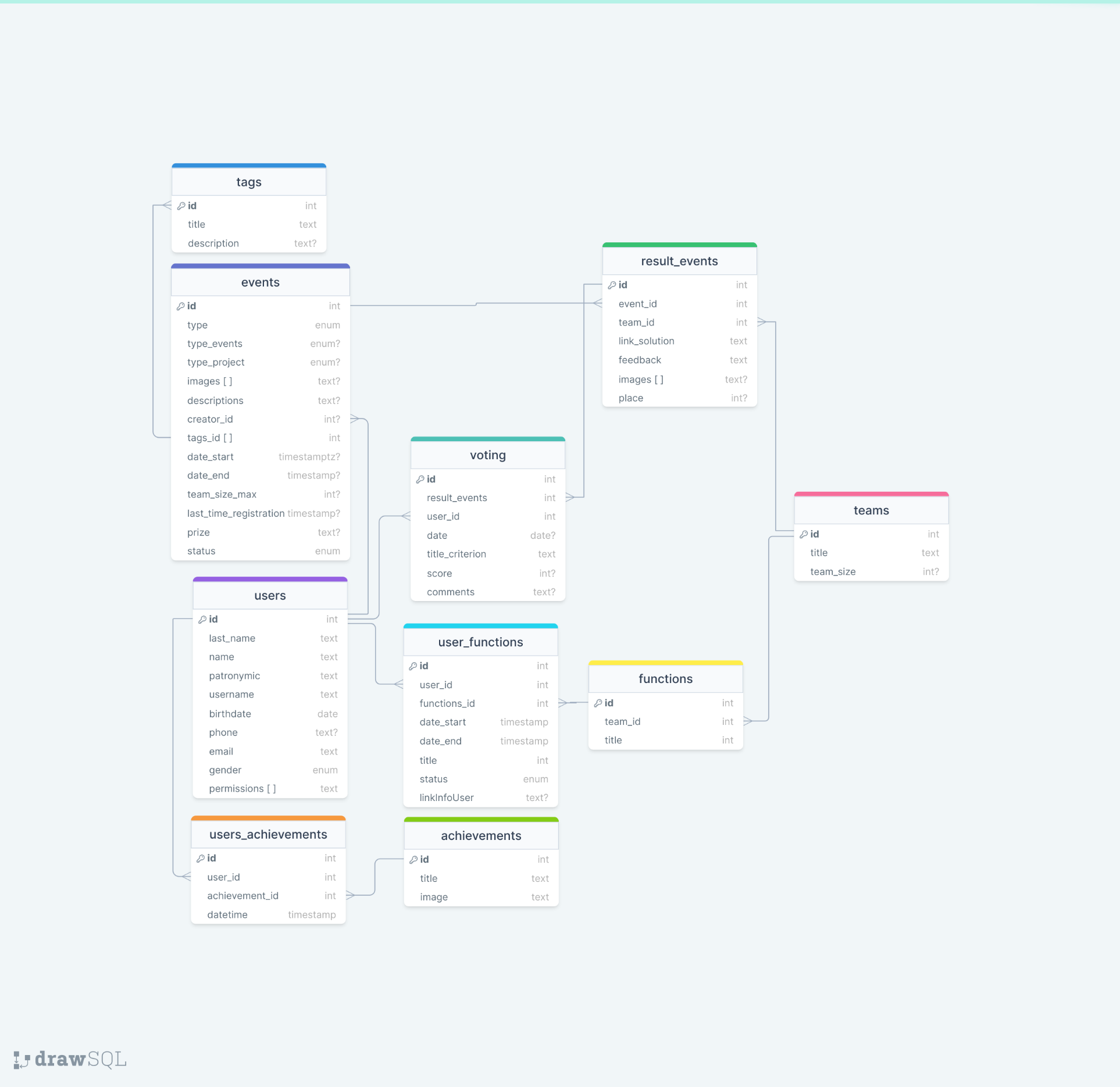


Рисунок 11 – Физическая модель данных

# **7 Пользовательский интерфейс**

Ниже приведены примеры макетов, сделанных в онлайн - сервисе Figma. Все макеты можно посмотреть по ссылке: [Макеты Биржа опыта](https://www.figma.com/file/MwEZrUat1v0jSNV96TtIvO/%D0%91%D0%B8%D1%80%D0%B6%D0%B0-%D0%BE%D0%BF%D1%8B%D1%82%D0%B0?node-id=0%3A1&t=ipHxg46Kslc25PPd-1)

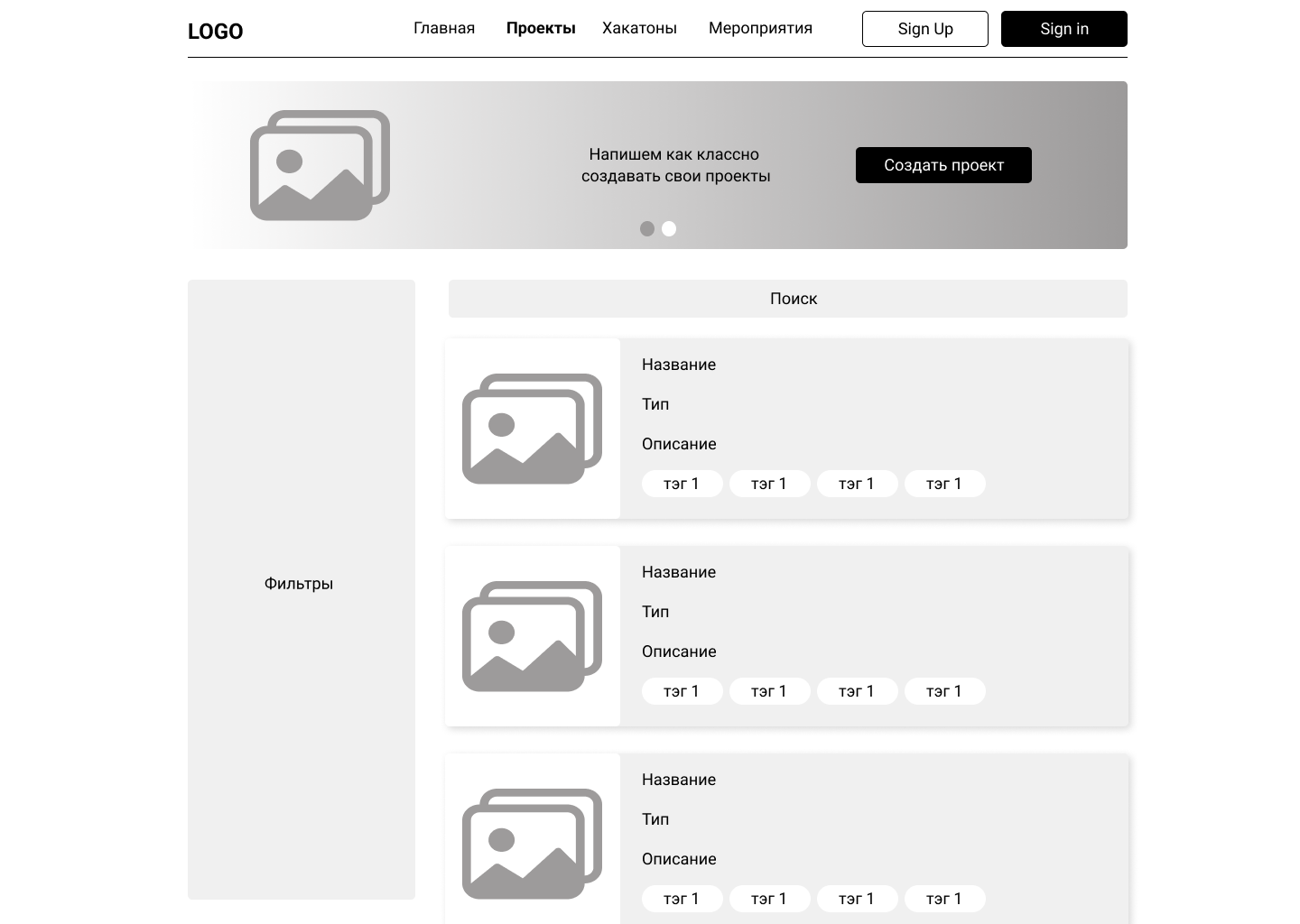


Рисунок 12 – Страница проектов

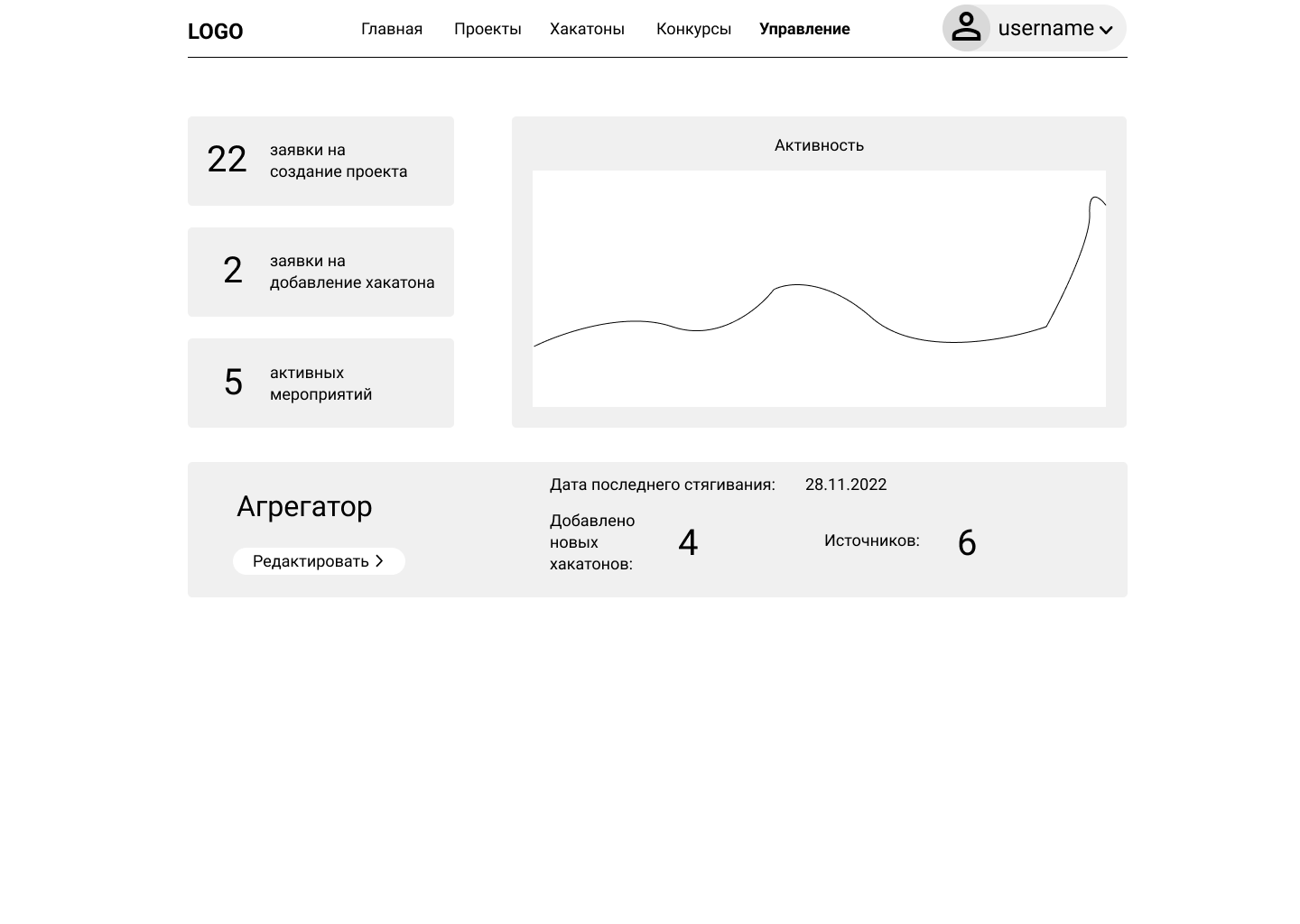


Рисунок 13 – Страница для управления системой

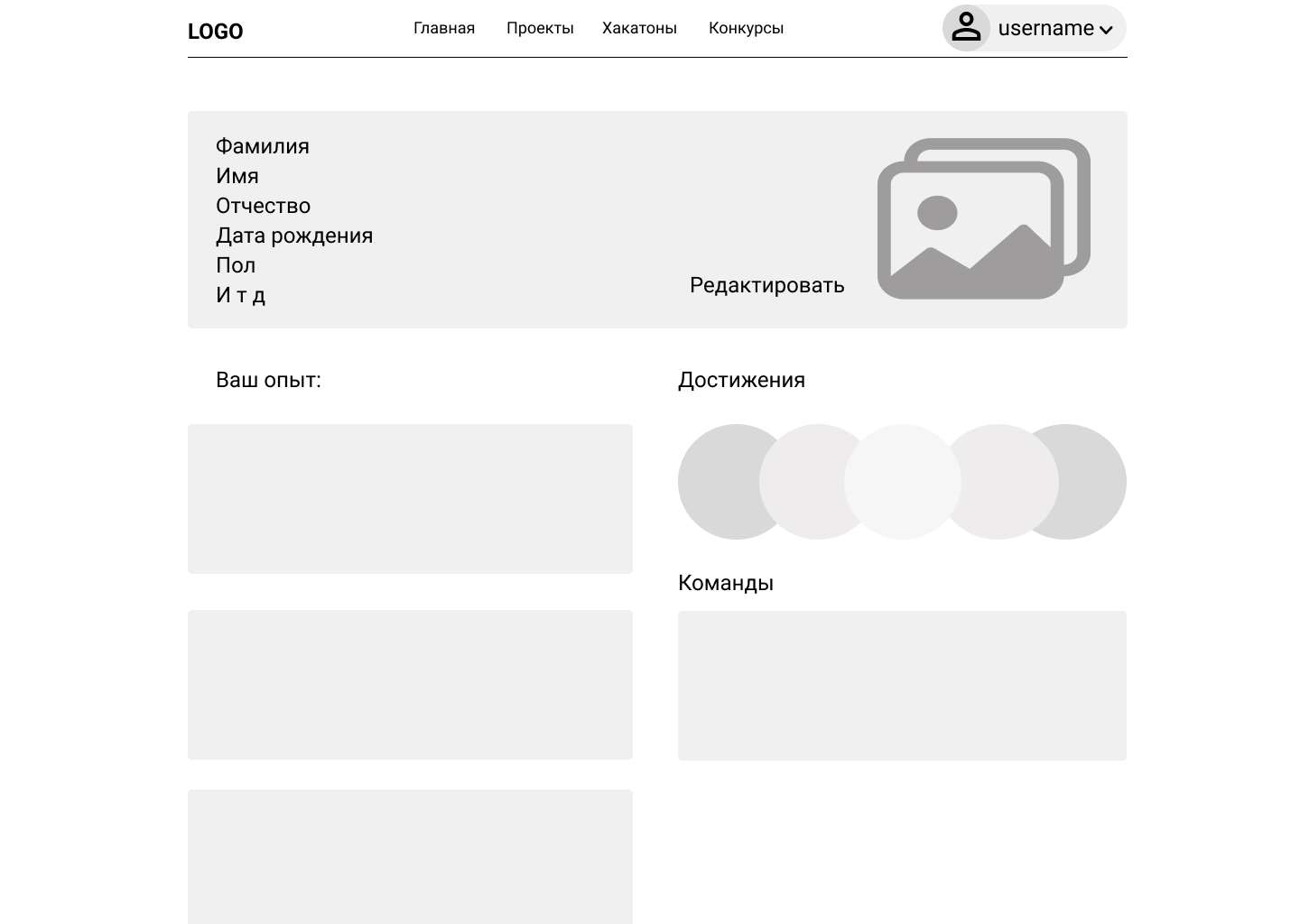


Рисунок 14 – Страница «Профиль»

# Вывод

Этап проектирования на данной итерации помог выявить какие необходимо добавить атрибуты в модель данных, что необходимо отобразить в макетах и как изменить структуру страниц, как необходимо реализовать систему и какие нужно учесть нюансы.

Необходимо понимать, что данный этап проектирования является первичным, и в следующих итерациях проекта (для курсовой) будет происходить проектирование на более детализированные уровни и для новых функций, которые будут стоять в приоритете. На данный момент, выполненного проектирования достаточно чтобы разработать MVP по первой итерации.