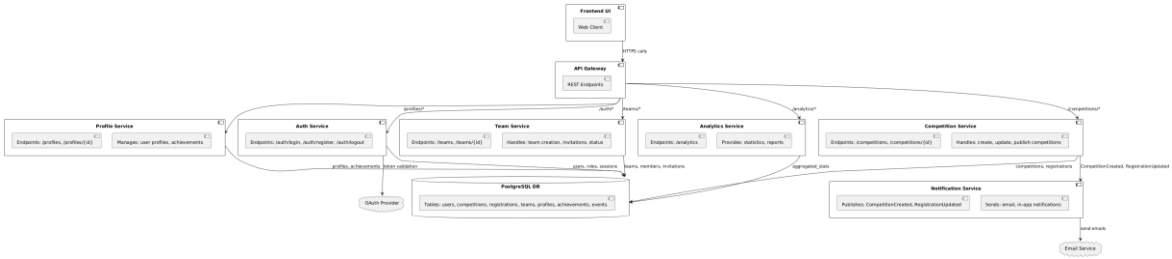
1. Диаграммы архитектуры

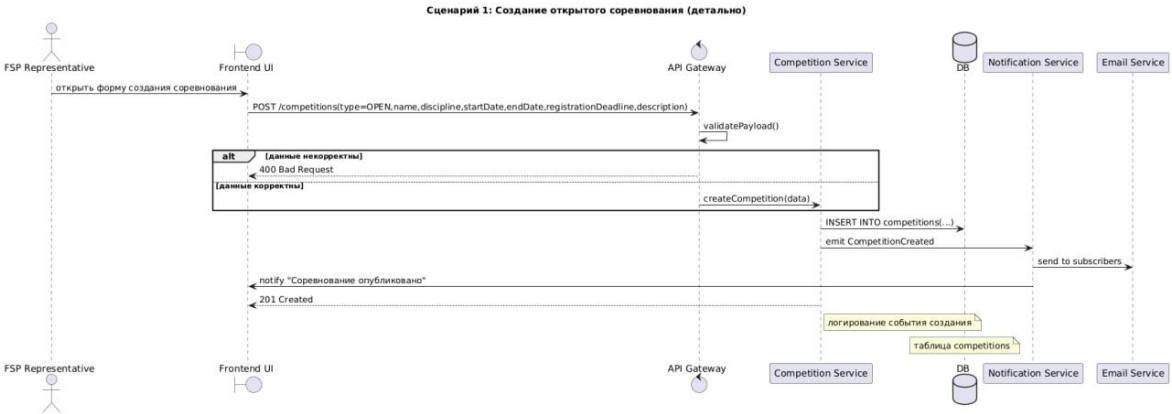
Мы использовали UML-диаграммы для отображения структуры системы.

Диаграмма компонентов нашей информационной системы:

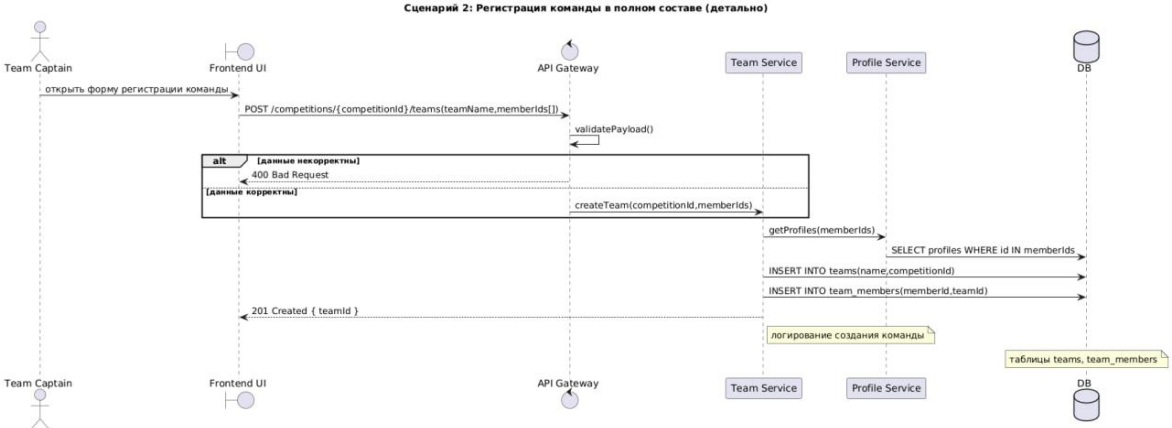


А также 10 диаграмм последовательности для каждого из сценариев:

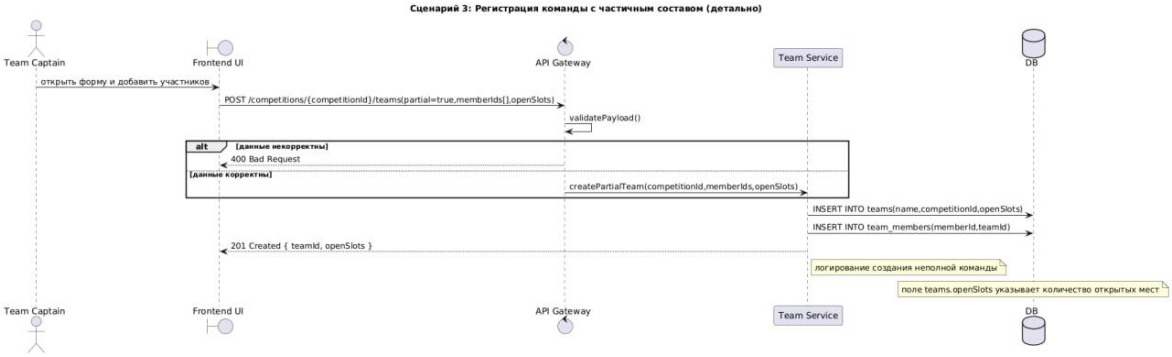
1. Сценарий 1:



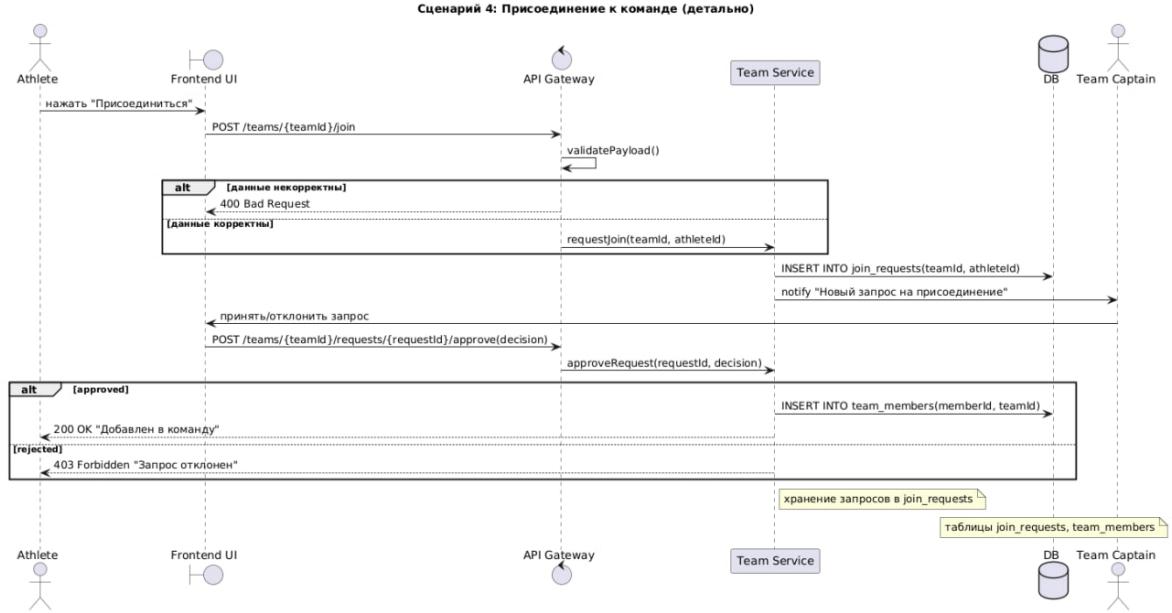
2. Сценарий 2:



3. Сценарий 3

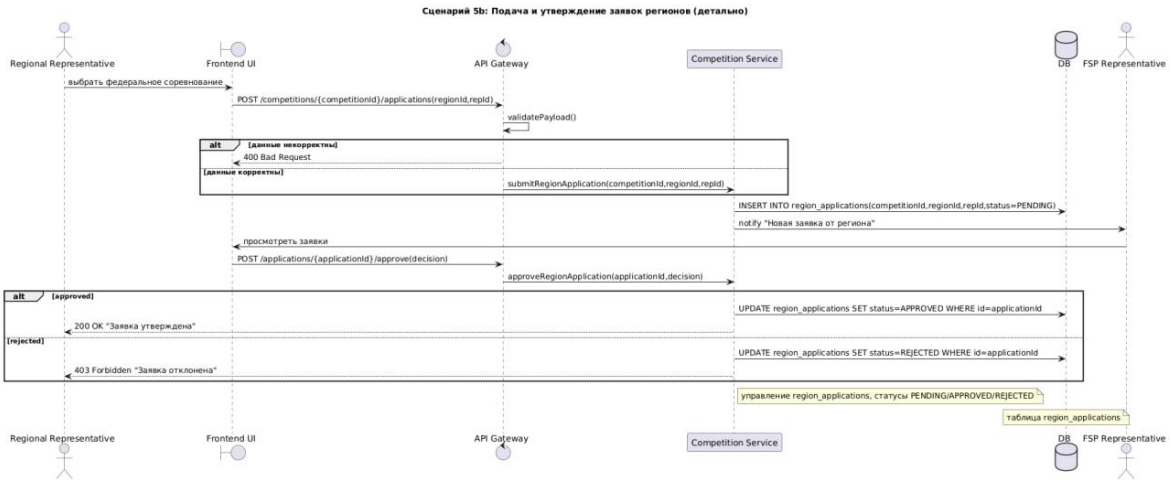


4. Сценарий 4

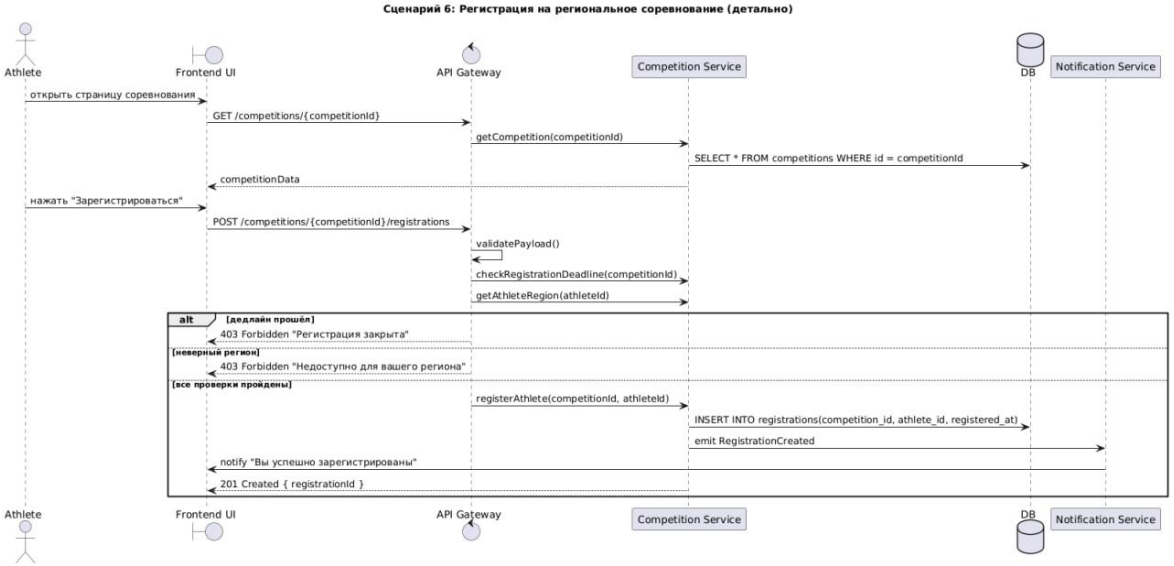


5. Сценарий 5a

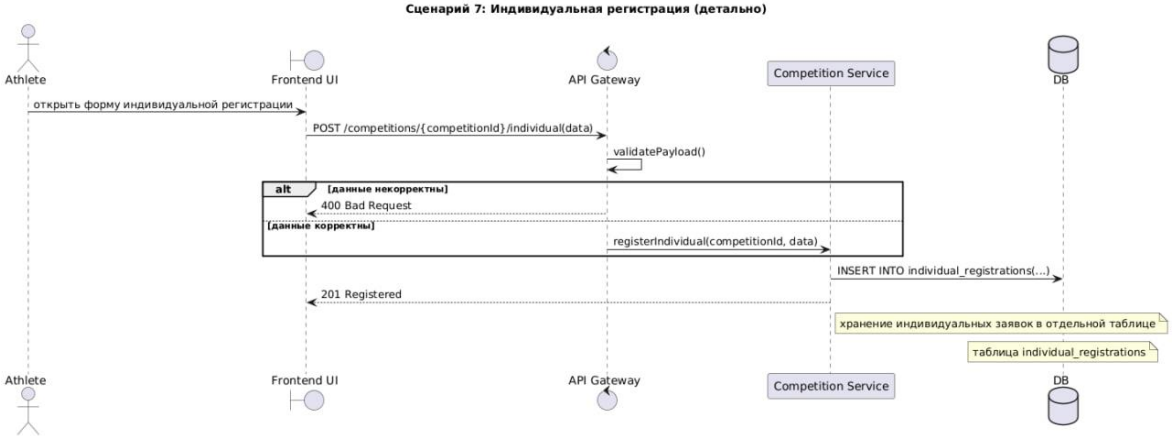
6. Сценарий 5b



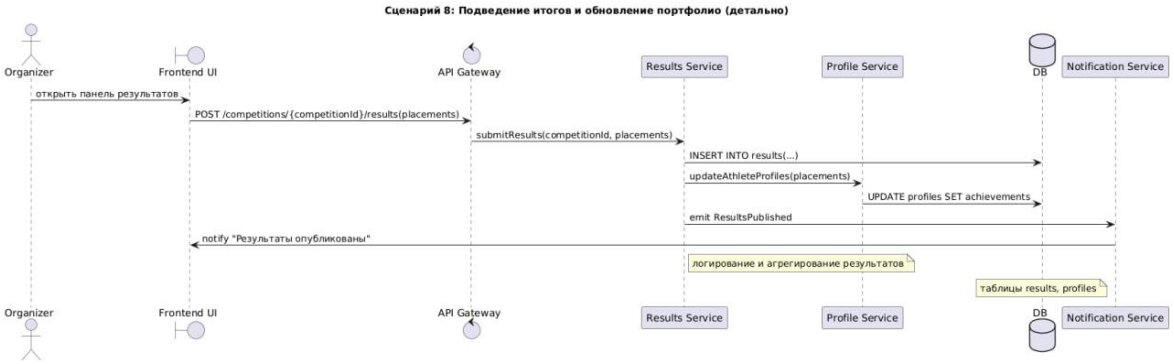
7. Сценарий 6



8. Сценарий 7

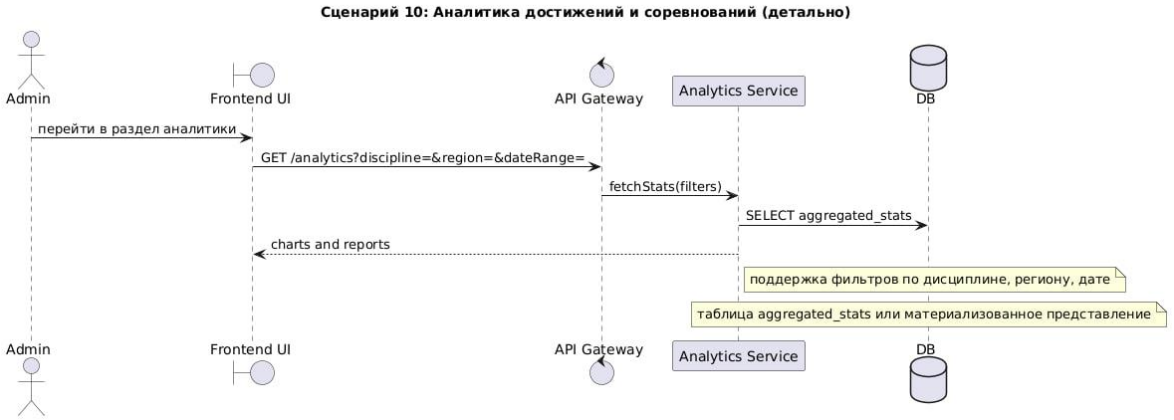


9. Сценарий 8



10.Сценарий 9

11.Сценарий 10



2. Функциональную схему базы данных (если применимо), включая:

- ER-диаграмму (сущности, атрибуты, связи между ними).
- Описание ключевых таблиц, их назначение, индексы, ограничения.
- Пояснение принципов хранения и обработки данных.
- Любые другие наработки (презентации, прототипы интерфейсов, скрипты).

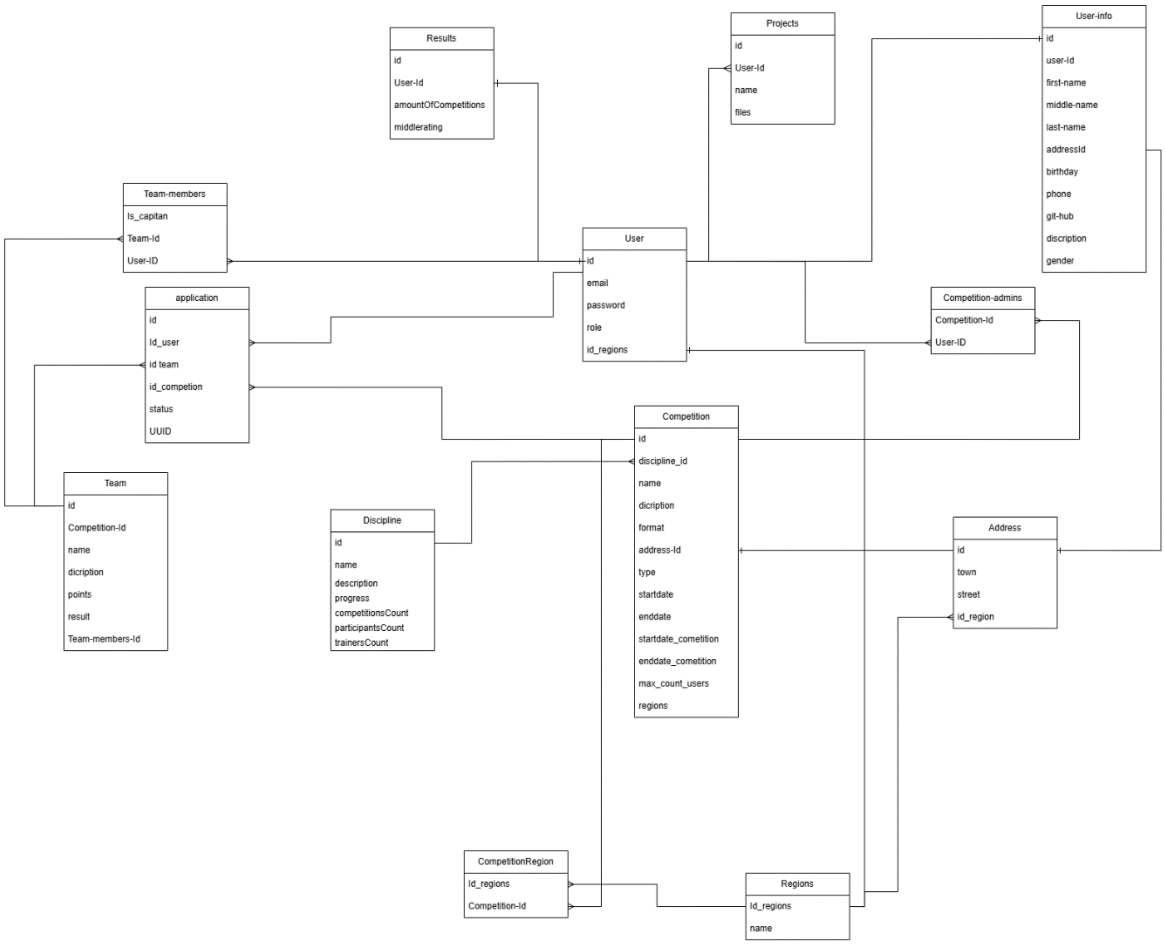


Таблица	Назначение
User	Основная таблица пользователей (аккаунты)
UserInfo	Расширенная информация о пользователе
Results	Сводная информация о результатах пользователя
Team	Команды, участвующие в соревнованиях
Teammembers	Промежуточная таблица между пользователями и командами

Competition	Информация о соревнованиях
CompetitionAdmins	Администраторы конкретных соревнований
Projects	Проекты, связанные с пользователями
Address	Таблица адресов, что участников, что и мест проведения соревнований
Discipline	Дисциплины соревнований
Regions	Регионы (например, для фильтрации или статистики)
Application	Заявки пользователей на участие в соревнованиях
CompetitionRegion	Промежуточная таблица между соревнованями и регионами

1. User (Пользователь)

Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Первичный ключ
email	STRING	UNIQUE, NOT NULL	Адрес электронной почты пользователя
password	STRING	NOT NULL	Хэш пароля пользователя
role	STRING	NOT NULL, DEFAULT 'user'	Роль в системе (user, admin)
id_regions	INT	NOT NULL, FOREIGN KEY id_regions → Regions.id	Внешний ключ на таблицу Regions

Связи:

1. 1.1 Один-к-одному с UserInfo (FOREIGN KEY userId → User.id)
2. 1.2 Один-к-одному с Results (FOREIGN KEY userId → User.id)
3. 1.3 Один-ко-многим с Projects (FOREIGN KEY userId → User.id)

4. 1.4 Один-ко-многим с CompetitionAdmins (FOREIGN KEY userId → User.id)
5. 1.5 Один ко многим cTeammembers (FOREIGN KEY userId → User.id)
6. 1.6 Многие-к-одному с Regions (FOREIGN KEY id_regions → Regions.id)
7. 1.7 Один-ко-многим с Application (FOREIGN KEY id_user → User.id)

2. UserInfo (Расширенная информация о пользователе)
Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Первичный ключ
User-id	INT	NOT NULL, FOREIGN KEY User-id → User.id	Ссылка на айди пользователя
firstName	STRING	NOT NULL	Имя пользователя
middleName	STRING	NULL ALLOW	Отчество пользователя
lastName	STRING	NOT NULL	Фамилия пользователя
birthday	DATE	NOT NULL	Дата рождения
phone	STRING	NULL ALLOW	Телефонный номер
gender	STRING	NULL ALLOW	Пол пользователя
github	STRING	NULL ALLOW	Ссылка на GitHub
description	STRING	NULL ALLOW	Краткое описание профиля
addressId	INT	NULL ALLOW, FOREIGN KEY addressId → Adress.id	Ссылка на адрес

Связи:

8. Один-к-одному с User (FOREIGN KEY userId → User.id)
9. Один-к-одному с Address (FOREIGN KEY addressId → Address.id)

3. Results (Результаты пользователя)
Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор записи
userId	INT	FK → User.id, NOT NULL	Ссылка на пользователя
amountOfCompetitions	INT	DEFAULT 0	Количество соревнований
middlerating	INT	DEFAULT NULL	Средний рейтинг участника

Связи:

10. Один-к-одному с User (FOREIGN KEY userId → User.id)

4. Team (Команда)
Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор команды
name	STRING	UNIQUE, NOT NULL	Название команды

description	STRING	NULL ALLOW	Описание команды
points	INT	DEFAULT 0	Накопленные баллы
result	INT	NULL ALLOW	Итоговое место/результат
competitionId	INT	FK → Competition.id, NOT NULL	Ссылка на соревнование
Team-members-Id	INT	FK → Team-members.Id, NOT NULL	Ссылка на участников

Связи:

11. Один-ко-многим с TeamMembers (FOREIGN KEY teamId → Team.id)
12. Один-ко-многим с Application (FOREIGN KEY id_team → Team.id)
13. Многие-к-одному с Competition (FOREIGN KEY competitionId → Competition.id)

5. Teammembers (Участники команд, связь User ↔ Team)

Атрибуты:

Имя	Тип	Ограничения	Описание
Is_capitan	BOOL	DEFAULT FALSE, NOT NULL	Проверка является ли участник капитаном
userId	INT	FK → User.id, NOT NULL	Ссылка на пользователя
teamId	INT	FK → Team.id, NOT NULL	Ссылка на команду

Связи:

14. Многие-к-одному с User (FOREIGN KEY userId → User.id)
15. Многие-к-одному с Team (FOREIGN KEY teamId → Team.id)

6. Competition (Соревнование)
Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор соревнования
disciplineId	INT	FK → Discipline.id, NOT NULL	Ссылка на дисциплину
name	STRING	UNIQUE, NOT NULL	Название соревнования
description	STRING	NULL ALLOW	Описание соревнования
format	STRING	NOT NULL	Формат проведения
type	STRING	NOT NULL, DEFAULT 'open'	Тип соревнования
startDate	DATETIME	NOT NULL	Дата и время начала регистрации
endDate	DATETIME	NOT NULL	Дата и время окончания регистрации
startdate_cometition	DATETIME	NOT NULL	Дата и время начала соревнований
enddate_cometition	DATETIME	NOT NULL	Дата и время окончания соревнований
maxParticipants	INT	NOT NULL	Максимальное число участников
addressId	INT	FK → Address.id, NOT NULL	Адрес проведения
regions	INT	FK → CompetitionRegion.id_regions, NOT NULL	Разрешенные регионы

Связи:

16. Один-к-одному с CompetitionAdmins (FOREIGN KEY competitionId → Competition.id)
17. Один-ко-многим с Application (FOREIGN KEY id_competition → Competition.id)
18. Один-ко-многим с CompetitionRegion (FOREIGN KEY competitionId→ Competition.id)
19. Один-к-одному с Address (FOREIGN KEY addressId → Address.id)
20. Многие-к-одному с Discipline (FOREIGN KEY disciplineId → Discipline.id)

7. CompetitionAdmins (Администраторы соревнований)
Атрибуты:

Имя	Тип	Ограничения	Описание
userId	INT	FK → User.id, NOT NULL	Администраторы
competitionId	INT	FK → Competition.id, NOT NULL	Соревнование

Связи:

21. Многие-к-одному с User (FOREIGN KEY userId → User.id)
22. Многие-к-одному с Competition (FOREIGN KEY competitionId → Competition.id)

8. Projects (Личные достижения)
Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCRE	Идентификатор проекта

		MENT, NOT NULL	
userId	INT	FK → User.id, NOT NULL	Автор проекта
name	STRING	NOT NULL	Название достижения
files	STRING	NULLABLE	Путь или ссылка на файлы

Связи:

23. Многие-к-одному с User (FOREIGN KEY userId → User.id)

9. Address (Адреса)

Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор адреса
id_region	INT	FK → regions.id_regions, NOT NULL	Регион/область
town	STRING	NOT NULL	Город/населённый пункт
street	STRING	NULL ALLOW	Улица/адрес

Связи:

24. Один-ко-многим с Competition (FOREIGN KEY addressId → Address.id)
25. Один-ко-многим с UserInfo (FOREIGN KEY addressId → Address.id)
26. Многие-к-одному с Regions (FOREIGN KEY id_regions → regions.id_regions

10. Discipline (Справочник дисциплин)

Атрибуты:

Имя	Тип	Ограничения	Описание

id	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор дисциплины
name	STRING	UNIQUE, NOT NULL	Название дисциплины
description	TEXT	NULL ALLOW	Описание дисциплины
competitionsCount	INT	DEFAULT 0	Кол-во проведённых соревнований
participantsCount	INT	DEFAULT 0	Кол-во участников за всё время
trainersCount	INT	DEFAULT 0	Кол-во тренеров/организаторов
progress	INT	CHECK (0–100)	Прогресс в процентах (от 0 до 100)

Связи:

27. Один-ко-многим с Competition (FOREIGN KEY discipline_id → Discipline.id)

11. Regions (Справочник регионов)

Атрибуты:

Имя	Тип	Ограничения	Описание
Id_regions	INT	PK, AUTO_INCREMENT, NOT NULL	Идентификатор региона
name	STRING	UNIQUE, NOT NULL	Название региона

Связи:

28. Один-ко-многим с User (FOREIGN KEY id_regions → Regions.id_regions)
29. Один-ко-многим с CompetitionRegion (FOREIGN KEY id_regions → Regions.id_regions)

30. Один-ко-многим с Adress (FOREIGN KEY id_regions → Adress.id_regions)

12. CompetitionRegion (Регионы соревнований)

Атрибуты:

Имя	Тип	Ограничения	Описание
competitionId	INT	FK → Competition.id, NOT NULL	Ссылка на соревнование
regionId	INT	FK → Regions.id, NOT NULL	Ссылка на регион

Связи:

31. Многие-к-одному с Competition → (FOREIGN KEY competitionId → Competition.id)
32. Многие-к-одному с Regions → (FOREIGN KEY id_regions → Regions.id)

13. Application (заявки на участие)

Атрибуты:

Имя	Тип	Ограничения	Описание
id	INT	PK, AUTO_INCREMENT, NOT NULL	Первичный ключ — уникальный идентификатор заявки

id_user	INT	NOT NULL, FOREIGN KEY → User.id	Внешний ключ на пользователя, который подал заявку
id_team	INT	NOT NULL, FOREIGN KEY → Team.id	Внешний ключ на команду, в которую подана заявка
id_competion	INT	NOT NULL, FOREIGN KEY → Competition.id	Внешний ключ на соревнование, на которое подана заявка
status	STRING	NOT NULL	Статус заявки
UUID	STRING	NULL ALLOW	Уникальный индефикатор

Связи:

33. Многие-к-одному с User (FOREIGN KEY id_user → User.id)
34. Многие-к-одному с Team (FOREIGN KEY id_team → Team.id)
35. Многие-к-одному с Competitions (FOREIGN KEY id_competion → Competition.id)

Пояснение принципов хранения и обработки данных

Система управления базами данных

В качестве системы управления базами данных используется PostgreSQL — объектно-реляционная СУБД, обеспечивающая хранение структурированных данных с поддержкой транзакций, индексов, ограничений целостности и связей между таблицами. Хранение данных реализовано в виде реляционной схемы с использованием нормализованных таблиц.

Серверная логика и ORM

Взаимодействие с базой данных осуществляется с использованием ORM-библиотеки Sequelize, работающей в среде Node.js. Модели Sequelize описываются в JavaScript и отображаются на соответствующие таблицы в базе данных PostgreSQL. Каждая модель включает определение полей (атрибутов), их типов, ограничений, а также связей с другими таблицами.

Принципы хранения данных

Данные хранятся в таблицах, каждая из которых соответствует конкретной сущности предметной области (пользователи, соревнования, команды и др.).

Каждая таблица имеет первичный ключ (PRIMARY KEY) для идентификации записей.

Для обеспечения уникальности и ссылочной целостности используются уникальные ограничения (UNIQUE) и внешние ключи (FOREIGN KEY).

Внешние ключи используются для организации связей между таблицами по принципам: один-к-одному, один-ко-многим.

Принципы обработки данных

Создание, чтение, обновление и удаление данных выполняется с использованием методов Sequelize (create, findOne, findAll, update, destroy).

Ассоциации между моделями определяются с помощью методов hasOne, belongsTo, hasMany, belongsToMany.

Обработка ассоциаций производится на уровне ORM, что позволяет выполнять связанные запросы с использованием операторов include и through.

Для валидации данных и реализации бизнес-логики используются встроенные ограничения (allowNull, defaultValue, validate), а также пользовательские хуки (beforeCreate, afterUpdate и др.).

Вся логика взаимодействия с базой данных инкапсулирована в модели и контроллерах, что обеспечивает модульность и повторное использование кода.

Обоснование выбора СУБД PostgreSQL

В качестве системы управления базами данных в проекте выбрана PostgreSQL. Данный выбор обусловлен следующими причинами:

1. Поддержка реляционной модели данных

Проект использует нормализованную реляционную структуру с четкими связями между сущностями (один-к-одному, один-ко-многим). PostgreSQL является полнофункциональной объектно-реляционной

СУБД, идеально подходящей для хранения и управления подобными структурами данных.

2. Расширенные возможности работы с ограничениями целостности

PostgreSQL поддерживает сложные типы ограничений:

первичные ключи (PRIMARY KEY),

внешние ключи (FOREIGN KEY),

уникальные ограничения (UNIQUE),

проверки значений (CHECK),

каскадное удаление (ON DELETE CASCADE),

что позволяет реализовать строгую валидацию данных на уровне СУБД.

3. Совместимость с ORM Sequelize

Выбор PostgreSQL также обусловлен её полной совместимостью с используемой ORM-библиотекой Sequelize. Это обеспечивает:

Автоматическую генерацию схем на основе JS-моделей;

Упрощённую реализацию CRUD-операций;

Гибкую работу с ассоциациями;

Механизмы миграций и версионирования структуры БД.

4. Широкая поддержка и масштабируемость

PostgreSQL является промышленным решением с открытым исходным кодом, которое активно поддерживается и развивается. Она масштабируется как вертикально, так и горизонтально, и может использоваться как на этапе прототипирования, так и в продуктивной среде.