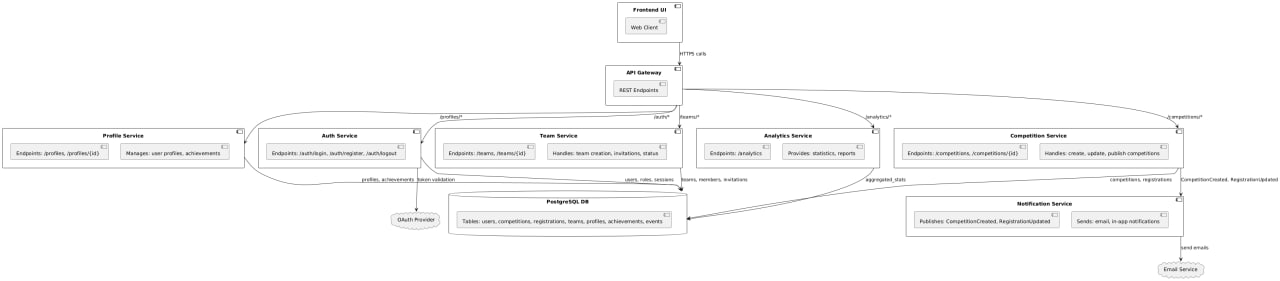
1. Диаграммы архитектуры

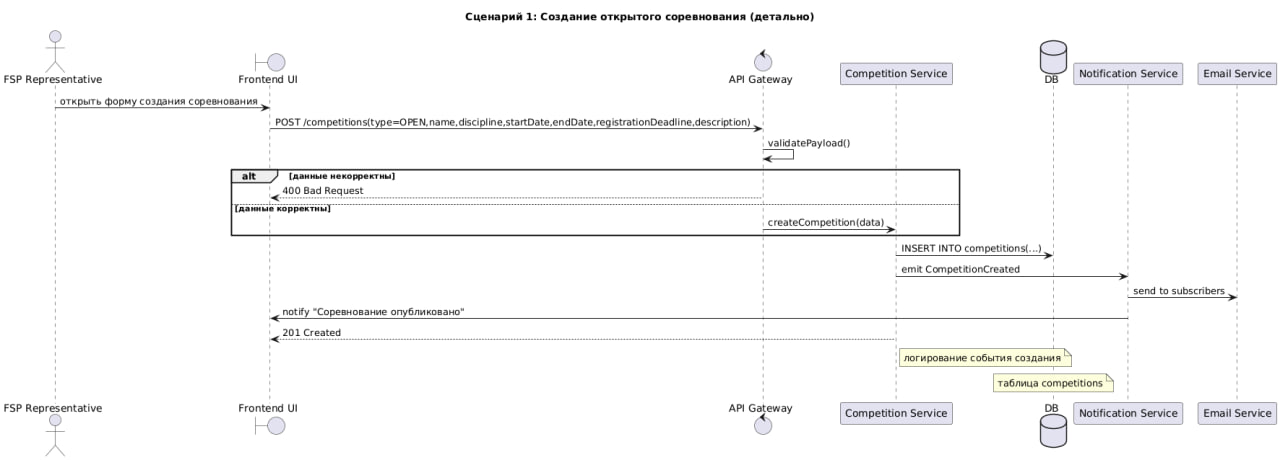
Мы использовали UML-диаграммы для отображения структуры системы.

Диаграмма компонентов нашей информационной системы:

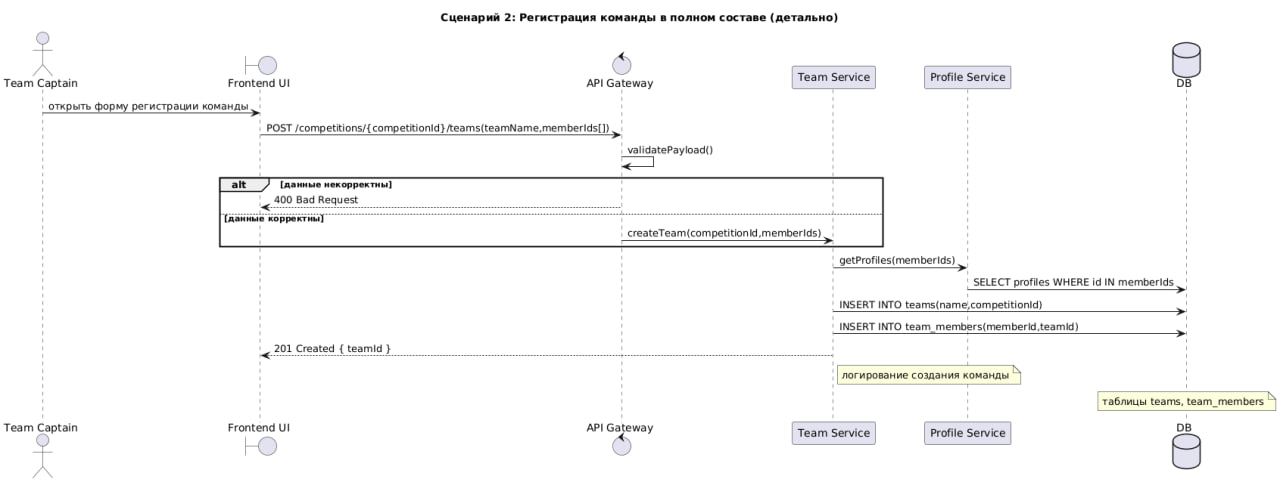


А также 10 диаграмм последовательности для каждого из сценариев:

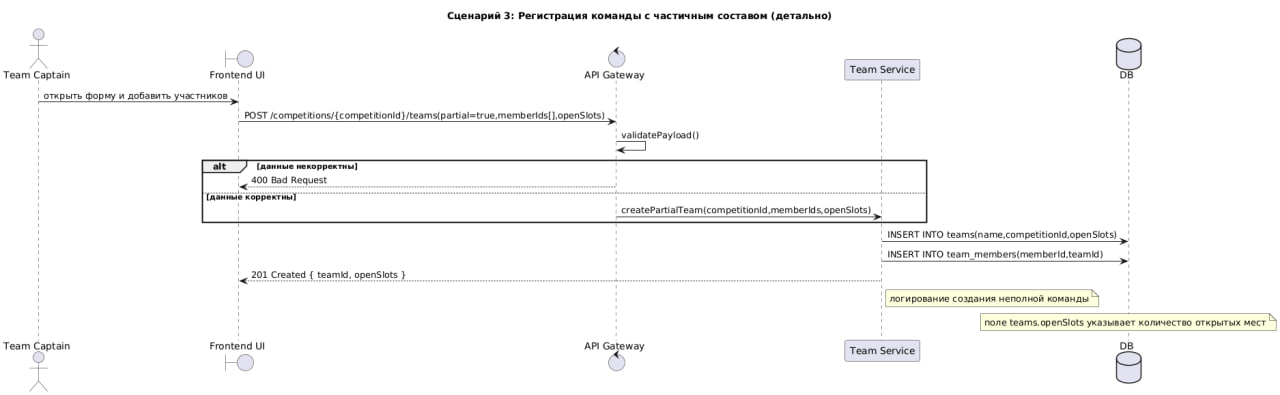
1. Сценарий 1:



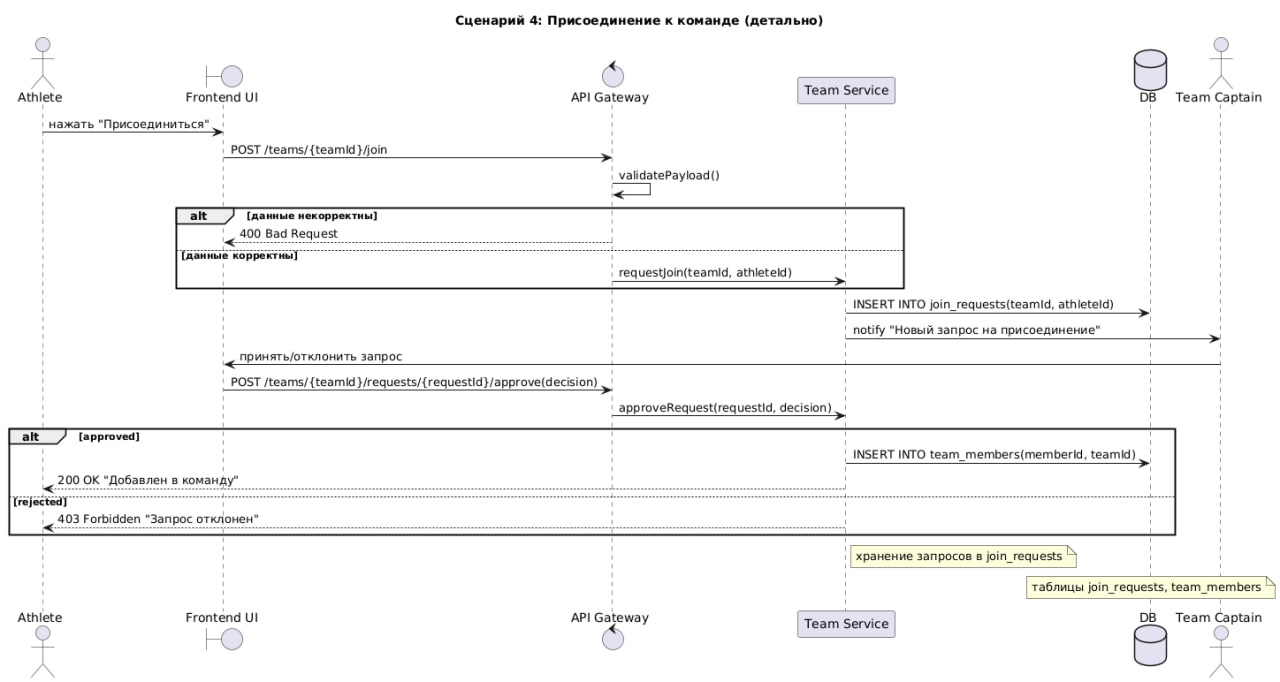
1. Сценарий 2:



1. Сценарий 3



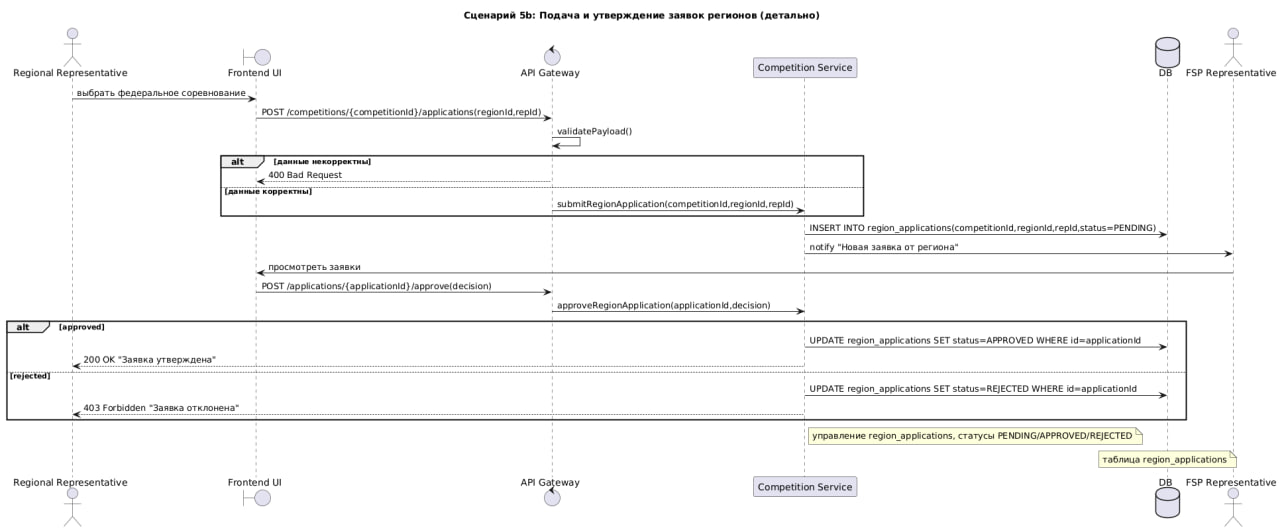
1. Сценарий 4



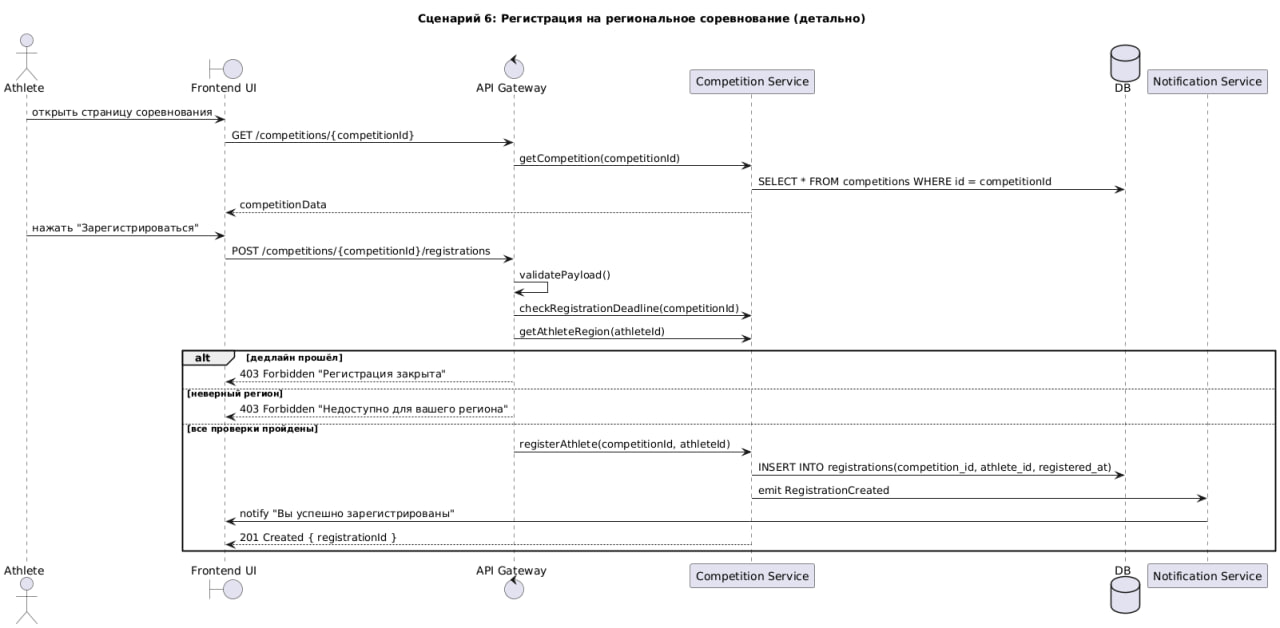
1. Сценарий 5a



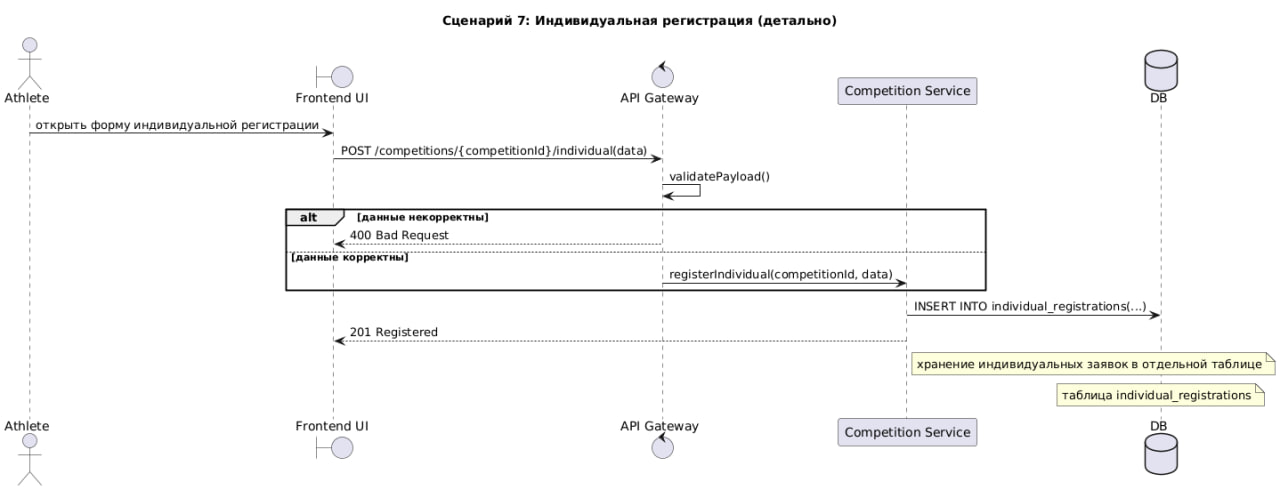
1. Сценарий 5b



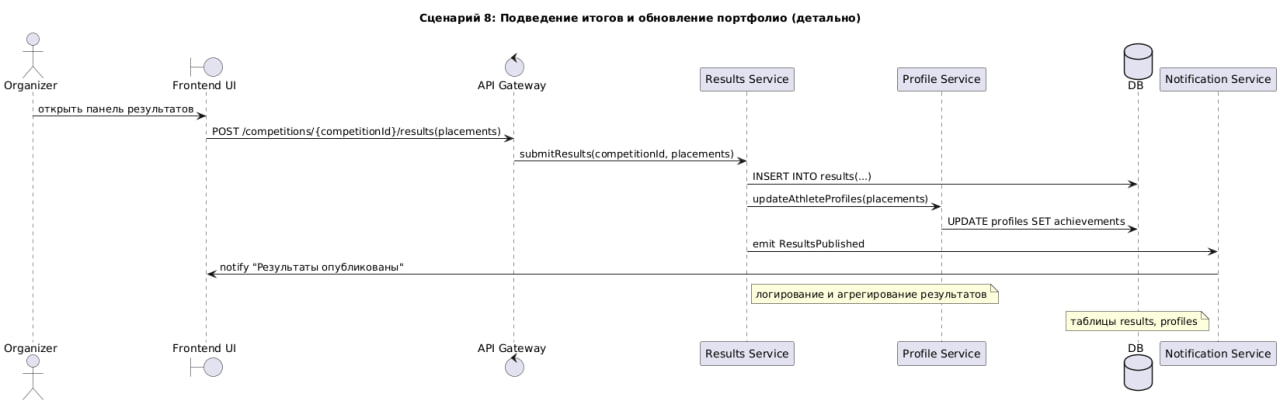
1. Сценарий 6



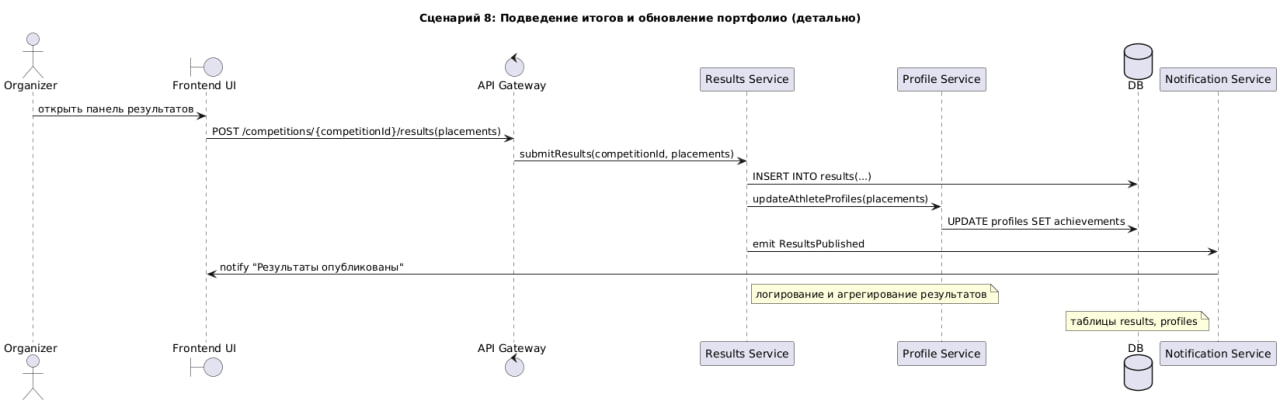
1. Сценарий 7



1. Сценарий 8



1. Сценарий 9

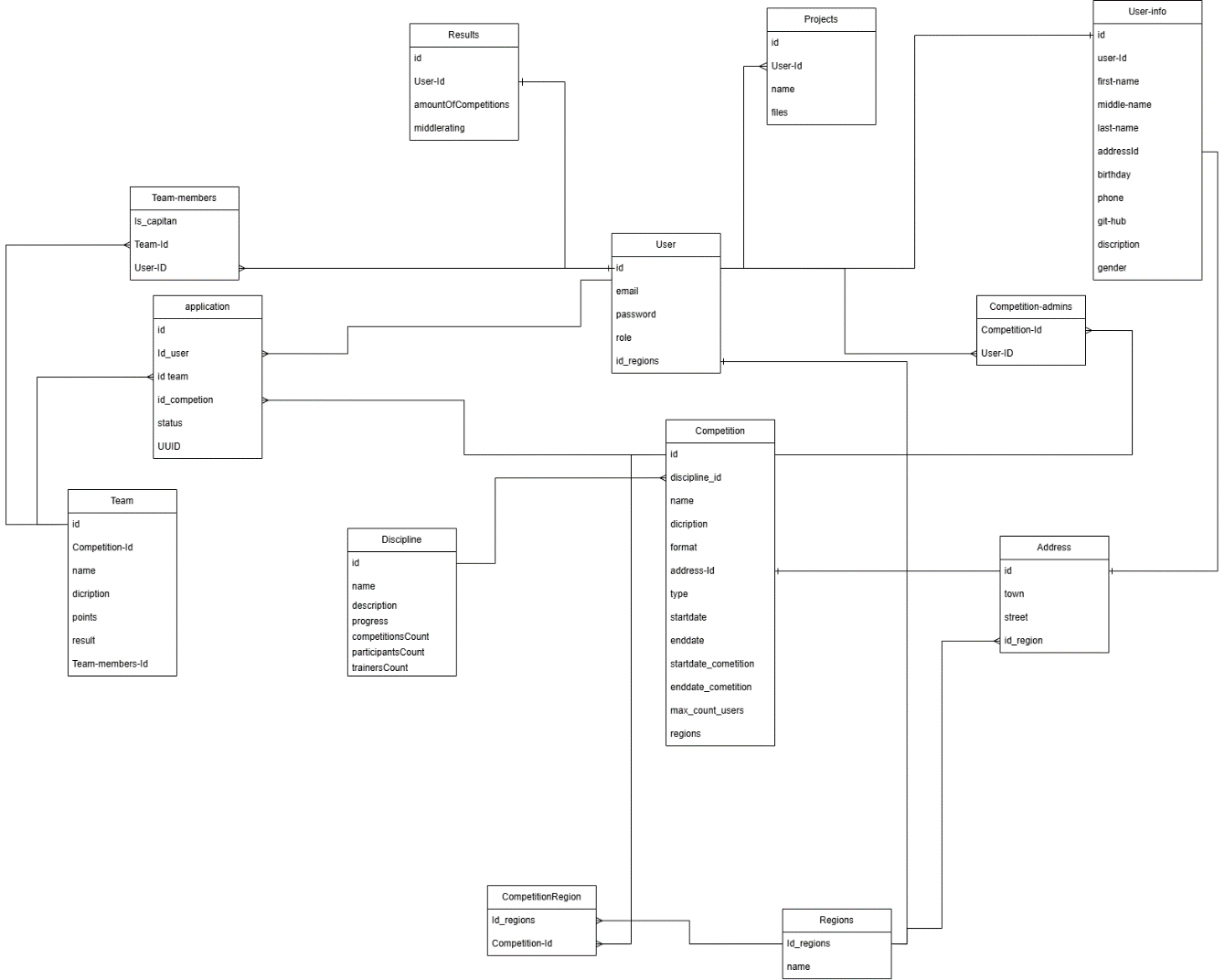


1. Сценарий 10



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 сTeammembers (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:

1. Один-к-одному с User (FOREIGN KEY userId → User.id)
2. 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:

1. 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:

1. Один-ко-многим с TeamMembers (FOREIGN KEY teamId → Team.id)
2. Один-ко-многим с Application (FOREIGN KEY id\_team → Team.id)
3. 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:

1. Многие-к-одному с User (FOREIGN KEY userId → User.id)
2. 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:

1. Один-к-одному с CompetitionAdmins (FOREIGN KEY competitionId → Competition.id)
2. Один-ко-многим с Application (FOREIGN KEY id\_competition → Competition.id)
3. Один-ко-многим с CompetitionRegion (FOREIGN KEY competitionId→ Competition.id)
4. Один-к-одному с Address (FOREIGN KEY addressId → Address.id)
5. 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:

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

# **8. Projects (Личные достижения)**

Атрибуты:

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Ограничения | Описание |
| id | INT | PK, AUTO\_INCREMENT, NOT NULL | Идентификатор проекта |
| userId | INT | FK → User.id, NOT NULL | Автор проекта |
| name | STRING | NOT NULL | Название достижения |
| files | STRING | NULLABLE | Путь или ссылка на файлы |

Связи:

1. 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:

1. Один-ко-многим с Competition (FOREIGN KEY addressId → Address.id)
2. Один-ко-многим с UserInfo (FOREIGN KEY addressId → Address.id)
3. 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:

1. 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:

1. Один-ко-многим с User (FOREIGN KEY id\_regions → Regions.id\_regions)
2. Один-ко-многим с CompetitionRegion (FOREIGN KEY id\_regions → Regions.id\_regions)
3. 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:

1. Многие-к-одному с Competition → (FOREIGN KEY competitionId → Competition.id)
2. 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:

1. Многие-к-одному с User (FOREIGN KEY id\_user → User.id)
2. Многие-к-одному с Team (FOREIGN KEY id\_team → Team.id)
3. 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.

**Миграции и управление схемой**

Изменения в структуре базы данных реализуются с использованием миграций Sequelize, что позволяет контролировать версии схемы БД и обеспечивать ее синхронизацию между средами разработки, тестирования и эксплуатации.

Инициализация базы данных осуществляется программно при запуске проекта или с помощью специальных скриптов миграции.

**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.