

Федеральное государственное автономное образовательное учреждение  
высшего образования

Национальный Исследовательский Университет ИТМО

Курсовая работа

Дисциплина: “Информационные системы и базы данных”

Тема: “Информационная система спортивного плавания”.

Выполнили: Аллаяров Игорь Олегович и Пушкин Антон Сергеевич

Факультет: Программной инженерии и компьютерной техники

Преподаватель: Гаврилов Антон Валерьевич

Город Санкт-Петербург

2024 год

<b>1-я часть</b>	<b>2</b>
Описание предметной области	2
<b>2-я часть</b>	<b>7</b>
Сущности	7
ER-диаграмма	7
Даталогическая модель	8
<b>3-я часть</b>	<b>10</b>
База данных	10
<b>4-я часть</b>	<b>15</b>
GITHUB	15
Примеры кода:	15

## 1-я часть

# Описание предметной области

### Предметная область:

Система призвана предоставлять информацию и возможности людям, которые профессионально занимаются плаванием.

### Представляемая информация разделена на следующие предметные области:

- Спортшкола
- Бассейн
- Спортсмен
- Тренер
- Соревнование
- Результат
- Тренировка
- Задание

, где:

**Спортшкола** содержит сведения о местоположении школы(город), информацию об используемых бассейнах, тренерском составе и

составе участников школ, а также статус каждого “школьника” или же разряд

**Бассейн** содержит информацию о местоположении(город), о датах проведения соревнований и о размерах бассейна:

*длина чаши, число дорожек.*

**Спортсмен** содержит сведения о самом спортсмене:

ФИО, город проживания, разряд, тип плавания, спортшкола, возраст, год рождения, актуальный допуск, тренер, участие в соревнованиях.

**Тренер** содержит сведения о лице, занимающем должность тренера: ФИО, город проживания, разряд, спортшкола, возраст, год рождения, тренируемые спортсмены.

**Соревнование** содержит информацию о событии, его названии, уровне соревнования, месте проведения(объект и город).

**Результат** содержит итоговые результаты заплыва:

*время дистанции, время реакции, очки Fina*

и характеристики самого заплыва:

*дистанция, стиль плавания*

**Тренировка** предоставляет возможность записи на подготовительное занятие и содержит информацию о месте и дате проведения, дистанции и тренера, проводящего занятие.

**Задание** содержит информацию для проведения самостоятельной тренировки. То есть данный раздел позволяет заниматься самостоятельно без тренера. Виды тренировок учитывают расстояние и стиль плавания. Сами задания учитывают режим и отдых

**Медосмотр** содержит общие сведения о медосмотре:

*дата проведения*

и предоставляет возможность прохождения медосмотра и сдачи теста на допинг. Также раздел “медосмотр” будет хранить результаты самого медосмотра: срок действия, допуск к соревновательной деятельности и результат теста на допинг

## **Основная идея и ее актуальность**

Будучи спортсменом, который принимает участие в различных соревнованиях данная система удобна тем, что просто содержит всю актуальную информацию о ближайших событиях, будь то соревнования или медосмотр.

Возможность занятий(самостоятельных тренировок или с тренером) - большой плюс, потому как личные программы тренировок составлены тренерами, а значит не придется весь год тренироваться по программе с ютуба, которая вовсе может быть неправильно составлена. В свою очередь занятие с личным тренером будет легко организовывать за счет сервиса.

Следующий плюс, который является и актуальной проблемой. Возможность просмотра профиля другого спортсмена с его результатами. Таких сервисов нет, а возможность знать все про своего соперника позволяет грамотно готовиться к соревнованиям.

Наш сервис - все самое необходимое(и даже больше) для пловца в одном месте

Ниже представлено более детальное техническое описание:

## **Основные требования к функциям системы:**

- Поиск и фильтрация спортсменов в информационной системе.
- Просмотр рейтинга лучших спортсменов по фильтрам.
- Просмотр профиля и таблицы результатов спортсмена.
- Просмотр расписания других спортсменов (при разрешении).
- Добавление и редактирование спортсменов и информации о них.
- Добавление и редактирование медосмотров для спортсмена.
- Добавление и редактирование соревнований и информации о них.
- Добавление и редактирование результатов участия в соревнованиях спортсмена.

- Добавление и редактирование бассейнов и информации о них.
- Добавление и редактирование спортшкол и информации о них.
- Составление и редактирование расписания спортсмена (его плана тренировок с заданиями).

### **Описание (области и акторы):**

Предметная область представляет собой информационный сервис спортивного плавания.

Актор системы может иметь одну из двух ролей: спортсмен или тренер.

#### **Спортсмен**

Спортсмен имеет спортшколу, один или несколько тренеров, расписание тренировок, результаты. Расписание тренировок спортсмена представляет собой множество тренировок по датам проведения. Указать участие спортсмена в соревновании можно при актуальном допуске.

#### **Тренер**

Тренер имеет спортшколу, тренируемых им спортсменов. Тренировки составляет тренер, которые привязываются к спортсмену.

#### **Спортшкола**

Спортшкола как сущность содержит некоторое количество спортсменов, приписанных к ней. Также спортшкола имеет один или несколько бассейнов, в которых базируется, и тренеров.

#### **Соревнования**

Соревнования имеют объект проведения и город проведения. Каждому соревнованию соответствуют список результаты по каждой дистанции.

#### **Результат**

Результат является артефактом участия спортсмена в соревнованиях. Каждому результату соответствует 1 спортсмен.

#### **Задания**

Заданий может быть несколько, все они составляют тренировку. Задания являются соответствующими спортсмену.

## **Медосмотр**

Медосмотры связываются со спортсменами, при актуальности (по дате) устанавливая атрибут допуска.

## 2-я часть

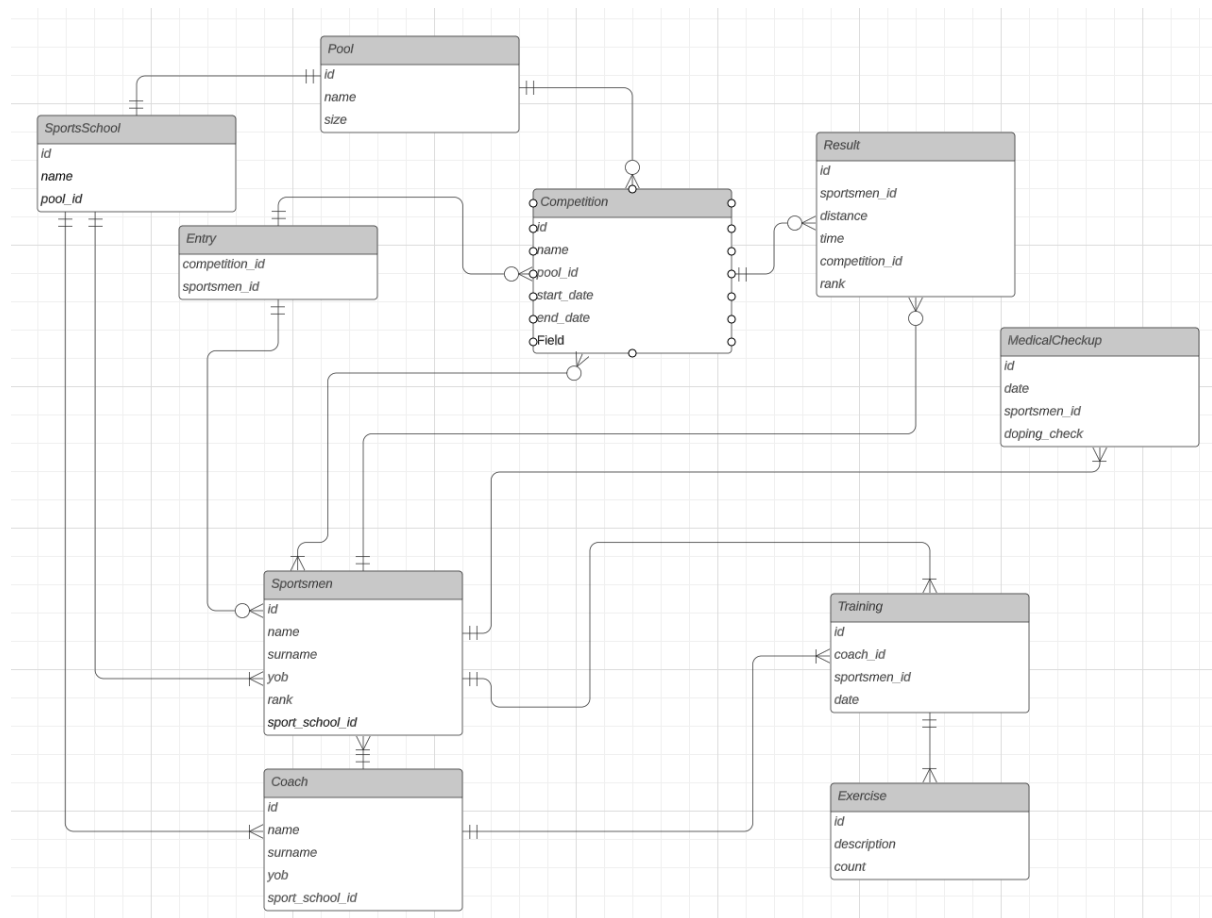
### Сущности

- SportsSchool - стержневая
- Pool - стержневая
- Sportsmen - стержневая
- Coach - стержневая
- Competition - стержневая
- Result - ассоциация (Competition, Sportsmen)
- Training - характеристика (Sportsmen)
- Exercise - характеристика (Training)
- MedicalCheckup - характеристика (Sportsmen)
- Entry - ассоциация (Competition, Sportsmen)

### ER-диаграмма

[https://lucid.app/lucidchart/ab4c7485-2146-4c6d-932a-e497cd763524/edit?viewport\\_loc=21%2C-103%2C1492%2C777%2C0\\_0&invitationId=inv\\_de910859-](https://lucid.app/lucidchart/ab4c7485-2146-4c6d-932a-e497cd763524/edit?viewport_loc=21%2C-103%2C1492%2C777%2C0_0&invitationId=inv_de910859-)

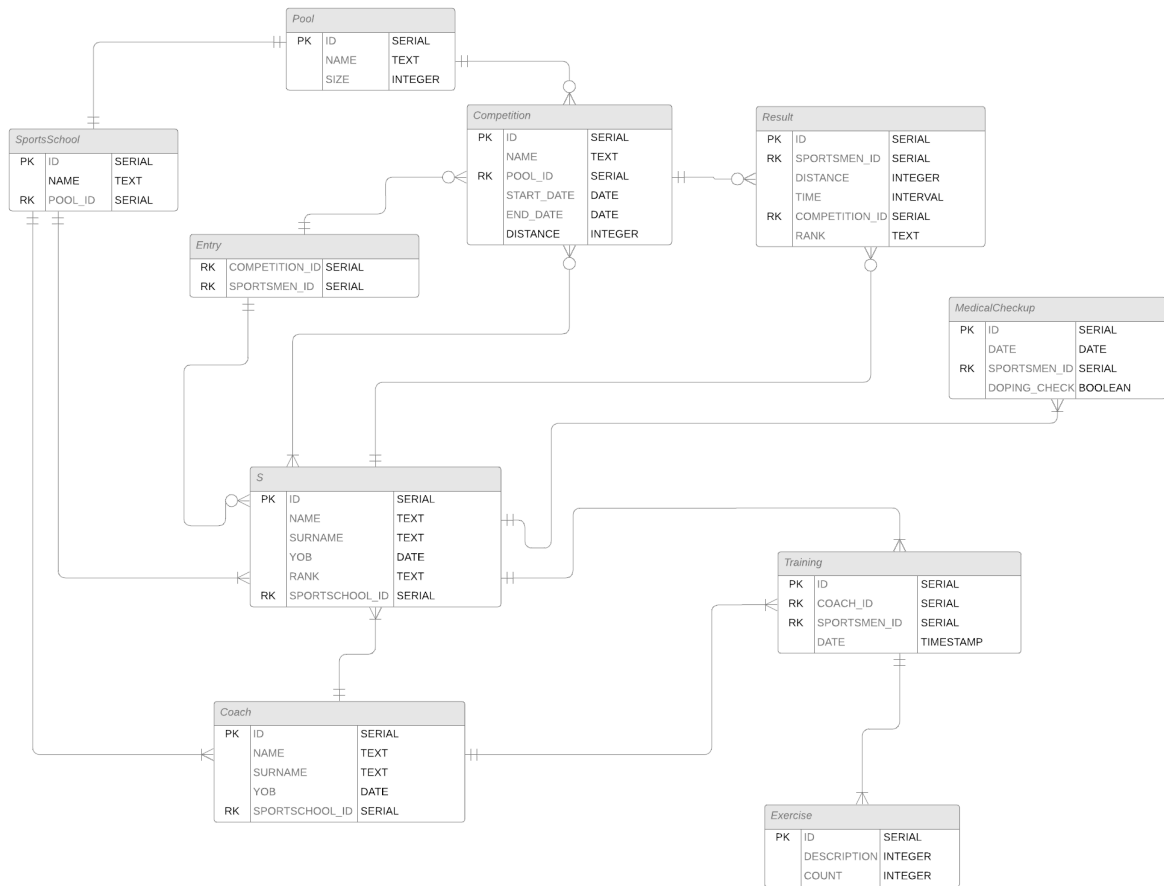
[41d4-488d-85cd-312f57481cf](#)



## Даталогическая модель

[https://lucid.app/lucidchart/aa61980d-0752-4018-9f70-363b38aa1f9f/edit?view\\_items=3UjBy3yesbFv&invitationId=inv\\_c178fb4e-c856-48e9-89c0-9e23f8de0f18](https://lucid.app/lucidchart/aa61980d-0752-4018-9f70-363b38aa1f9f/edit?view_items=3UjBy3yesbFv&invitationId=inv_c178fb4e-c856-48e9-89c0-9e23f8de0f18)





## 3-я часть

# База данных

### Создание таблиц

```
DO $$
BEGIN
    IF NOT EXISTS (SELECT 1 FROM pg_type WHERE typname = 'pool_size') THEN
        CREATE TYPE POOL_SIZE AS ENUM ('_50', '_25');
    END IF;

    IF NOT EXISTS (SELECT 1 FROM pg_type WHERE typname = 'rank') THEN
        CREATE TYPE RANK AS ENUM ('MS', 'CMS', 'I_A', 'II_A', 'III_A', 'I_Y',
        'II_Y', 'III_Y', 'NR');
    END IF;

    IF NOT EXISTS (SELECT 1 FROM pg_type WHERE typname = 'distance') THEN
        CREATE TYPE DISTANCE AS ENUM (
            '_50_BUTTERFLY', '_50_BACKSTROKE', '_50_BREASTSTROKE',
            '_50_FREESTYLE',
            '_100_BUTTERFLY', '_100_BACKSTROKE', '_100_BREASTSTROKE',
            '_100_MEDLEY',
            '_100_FREESTYLE', '_200_BUTTERFLY', '_200_BACKSTROKE',
            '_200_BREASTSTROKE',
            '_200_FREESTYLE', '_200_MEDLEY', '_400_FREESTYLE', '_400_MEDLEY',
            '_800_FREESTYLE', '_1500_FREESTYLE'
        );
    END IF;
END $$;

CREATE TABLE IF NOT EXISTS POOL(
    ID SERIAL PRIMARY KEY,
    NAME TEXT NOT NULL,
    POOL_SIZE POOL_SIZE
);

CREATE TABLE IF NOT EXISTS SPORTS_SCHOOL(
    ID SERIAL PRIMARY KEY,
    NAME TEXT NOT NULL,
    POOL_ID SERIAL REFERENCES POOL(ID)
);

CREATE TABLE IF NOT EXISTS COMPETITION(
    ID SERIAL PRIMARY KEY,
    NAME TEXT NOT NULL,
    POOL_ID SERIAL REFERENCES POOL(ID),
    START_DATE DATE NOT NULL,
    END_DATE DATE NOT NULL,
    DISTANCE DISTANCE
);
```

```

);

CREATE TABLE IF NOT EXISTS SPORTSMAN(
    ID SERIAL PRIMARY KEY,
    FULL_NAME TEXT NOT NULL,
    YOB DATE NOT NULL,
    RANK RANK,
    SPORTSCHOOL_ID SERIAL REFERENCES SPORTS_SCHOOL(ID)
);

CREATE TABLE IF NOT EXISTS ENTRY(
    COMPETITION_ID SERIAL REFERENCES COMPETITION(ID),
    SPORTSMAN_ID SERIAL REFERENCES SPORTSMAN(ID)
);

CREATE TABLE IF NOT EXISTS COACH(
    ID SERIAL PRIMARY KEY,
    FULL_NAME TEXT NOT NULL,
    YOB DATE NOT NULL,
    SPORTSCHOOL_ID SERIAL REFERENCES SPORTS_SCHOOL(ID)
);

CREATE TABLE IF NOT EXISTS RESULT(
    ID SERIAL PRIMARY KEY,
    SPORTSMAN_ID SERIAL REFERENCES SPORTSMAN(ID),
    DISTANCE DISTANCES CHECK(DISTANCE IN get_competition_distant(ID),
    TIME INTERVAL DEFAULT '0 MINUTE_SECOND',
    COMPETITION_ID SERIAL REFERENCES COMPETITION(ID),
    RANK RANK
);
--
CREATE TABLE IF NOT EXISTS MEDICAL_CHECKUP(
    ID SERIAL PRIMARY KEY,
    DATE DATE NOT NULL,
    SPORTSMAN_ID SERIAL REFERENCES SPORTSMAN(ID),
    DOPING_CHECK BOOLEAN
);

CREATE TABLE IF NOT EXISTS TRAINING(
    ID SERIAL PRIMARY KEY,
    COACH_ID SERIAL REFERENCES COACH(ID),
    SPORTSMAN_ID SERIAL REFERENCES SPORTSMAN(ID)
);

CREATE TABLE IF NOT EXISTS TRAINING_EVENT(
    SPORTSMAN_ID SERIAL REFERENCES SPORTSMAN(ID),
    COACH_ID SERIAL REFERENCES COACH(ID),
    TRAINING_ID SERIAL REFERENCES TRAINING(ID)
);

CREATE TABLE IF NOT EXISTS EXERCISE (
    ID SERIAL PRIMARY KEY,
    DESCRIPTION TEXT,

```

```
COUNT INTEGER DEFAULT 4,  
TRAINING_ID SERIAL REFERENCES TRAINING(ID)  
);
```

## Удаление таблиц

```
DROP TABLE EXERCISE;  
DROP TABLE TRAINING_EVENT;  
DROP TABLE TRAINING;  
DROP TABLE MEDICAL_CHECKUP;  
DROP TABLE RESULT;  
DROP TABLE COACH;  
DROP TABLE ENTRY;  
DROP TABLE SPORTSMAN;  
DROP TABLE COMPETITION;  
DROP TABLE SPORTS_SCHOOL;  
DROP TABLE POOL;
```

## Вставка данных

```
INSERT INTO pool (name, pool_size) VALUES  
('ПОЛЕТ', '_25'::POOL_SIZE),  
('Океан', '_50'::POOL_SIZE),  
('AquaSwim', '_50'::POOL_SIZE),  
('SwimCity', '_25'::POOL_SIZE),  
('Дом плавания', '_25'::POOL_SIZE),  
('Атлантика', '_50'::POOL_SIZE);
```

```
INSERT INTO sports_school (name, pool_id) VALUES  
('Школа7 города Новосибирск', 1),  
('Школа Спортивных Чемпионов', 2),  
('СДЮСШОР "Радуга"', 3),  
('Спортивный клуб "Динамо"', 4),  
('Школа плавания Эдуарда Гурьянова', 5),  
('Школа олимпийского резерва по плаванию', 6);
```

```
INSERT INTO sportsman (full_name, yob, rank, sportschool_id) VALUES  
('Коромысло Иван Филиппович', '2003-03-22', 'MS'::RANK, 6),  
('Фэлпс Федор Джонсович', '2000-12-02', 'III_Y'::RANK, 1),  
('Петров Баттерфляй Иванович', '2006-06-03', 'I_A'::RANK, 2),  
('Аллаяров Игорь Олегович', '2003-10-02', 'MS'::RANK, 4),  
('Брасов Алексей Кроллович', '2002-1-01', 'II_A'::RANK, 3),  
('Бревно Максим Невсплываевич', '2003-11-21', 'CMS'::RANK, 5);
```

```
INSERT INTO coach (full_name, yob, sportschool_id) VALUES  
('Победилов Василий Напенсиивич', '1788-03-08', 1),
```

```

('Рекодров Олег Побитович', '1988-09-01', 3),
('Старов Михаил Фелпсович', '1968-08-15', 4),
('Кроль Геннадий Баттерфляевич', '1980-08-19', 5),
('Бесов Никита Кукушкинович', '1986-06-06', 6);

INSERT INTO medical_checkup (date, sportsman_id, doping_check) VALUES
('2017-05-27', 1, 'False'),
('2023-09-01', 2, 'True'),
('2014-06-03', 3, 'False'),
('2023-11-03', 4, 'False'),
('2021-12-31', 5, 'False'),
('2014-01-1', 6, 'False');

INSERT INTO competition (name, pool_id, start_date, end_date, distance)
VALUES
('Кубок золотого поршня', 1, '2023-09-09', '2023-09-15',
'_50_BUTTERFLY'::DISTANCE),
('Плаваем вместе', 5, '2021-01-02', '2021-01-10',
'_100_BREASTSTROKE'::DISTANCE),
('Чемпионат и первенство ДВФО', 1, '2023-10-01', '2023-10-03',
'_1500_FREESTYLE'::DISTANCE),
('Чемпионат России по плаванию', 2, '2023-05-20',
'2023-10-17', '_200_BREASTSTROKE'::DISTANCE),
('Золотой граммофон', 3, '2017-03-17', '2020-02-12',
'_400_MEDLEY'::DISTANCE),
('Российский чемпионат "Мне не холодно"', 4, '2022-06-03', '2022-07-03',
'_50_BACKSTROKE'::DISTANCE);

INSERT INTO entry (competition_id, sportsman_id) VALUES
(1, 1),
(1, 2),
(1, 3);
INSERT INTO entry (competition_id, sportsman_id) VALUES
(2, 1),
(2, 2),
(2, 4);

INSERT INTO result (sportsman_id, distance, time, competition_id, rank)
VALUES
(1, '_50_BUTTERFLY'::DISTANCE, '00:22:30', 1, 'I_A'::RANK),
(2, '_50_BUTTERFLY'::DISTANCE, '00:23:15', 1, 'II_A'::RANK),
(3, '_50_BUTTERFLY'::DISTANCE, '00:24:00', 1, 'III_A'::RANK);

INSERT INTO result (sportsman_id, distance, time, competition_id, rank)
VALUES
(1, '_100_BACKSTROKE'::DISTANCE, '00:11:45', 2, 'I_A'::RANK),
(2, '_100_BACKSTROKE'::DISTANCE, '00:12:30', 2, 'II_A'::RANK),
(4, '_100_BACKSTROKE'::DISTANCE, '00:13:15', 2, 'III_Y'::RANK);

```

```

INSERT INTO training (coach_id, sportsman_id) VALUES
  (1, 1),
  (2, 2),
  (3, 3);

INSERT INTO training_event (sportsman_id, coach_id, training_id) VALUES
  (1, 1, 1),
  (2, 2, 2),
  (3, 3, 3);

INSERT INTO exercise (description, count, training_id) VALUES
  ('Подтягивания', 10,1),
  ('Приседания', 20,1),
  ('Отжимания', 15,1),
  ('Плавание вольным стилем', 400,2),
  ('Бег 100 метров', 1,2);

```

## Удаление данных

```

DELETE FROM EXERCISE;
DELETE FROM TRAINING_EVENT;
DELETE FROM TRAINING;
DELETE FROM MEDICAL_CHECKUP;
DELETE FROM RESULT;
DELETE FROM COACH;
DELETE FROM ENTRY;
DELETE FROM SPORTSMAN;
DELETE FROM COMPETITION;
DELETE FROM SPORTS_SCHOOL;
DELETE FROM POOL;

```

## Функция

```

CREATE OR REPLACE FUNCTION get_competition_distant (RESULT_ID
INTEGER) RETURNS DISTANCE AS
$$
BEGIN
SELECT DISTANCE FROM COMPETITION WHERE
RESULT_ID=COMPETITION_ID;
END
$$
LANGUAGE "plpgsql";

```

## Индексы

```
CREATE INDEX FIND_SPORTSMAN_SCHOOL ON SPORTSMAN  
(SPORTSCHOOL_ID);
```

## **Триггеры**

```
CREATE TRIGGER training_event_insert_trigger AFTER INSERT ON  
"TRAINING" FOR EACH ROW EXECUTE PROCEDURE  
training_event_insert_trigger_fun();
```

## 4-я часть

# GITHUB

### Back-end

<https://github.com/Anth0o0ny/db-coursework.git>

### Front-end

[https://github.com/Xswinger/course\\_work\\_front](https://github.com/Xswinger/course_work_front)

## Примеры кода:

### Back-end

Примеры некоторых классов:

Сущности:

```
public class Sportschool {
    private int id;
    private String name;
    private PoolSize poolSize;

    public Sportschool() {
    }

    public Sportschool(int id, String name, PoolSize poolSize) {
        this.id = id;
        this.name = name;
        this.poolSize = poolSize;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```



```

public PoolSize getPoolSize() {
    return poolSize;
}

public void setPoolSize(PoolSize poolSize) {
    this.poolSize = poolSize;
}
}

```

## Котроллер:

```

@RestController
@RequestMapping("/sportschool")
public class SchoolController {

    @Autowired
    private DB db;

    @PostMapping("/getSchoolByName")
    public List<Sportschool> getSchoolByName(@RequestBody Map<String, String>
requestBody) {
        String name = requestBody.get("name");
        return db.getSchoolByName(name);
    }
}

```

## Перечисления:

```

public enum PoolSize {

    _50 ("50 метров"),
    _25 ("25 метров");

    private final String title;

    PoolSize(String title) {
        this.title = title;
    }

    @JsonValue
    public String getTitle() {
        return title;
    }

    @JsonCreator
    public static PoolSize fromTitle(String title) {
        for (PoolSize poolSize : values()) {
            if (poolSize.title.equals(title)) {
                return poolSize;
            }
        }
        return null;
    }
}

```

## Запросы:

```
@Component
public class DB {

    @Autowired
    JdbcTemplate template;

    public DB() {
    }

    public List<Sportschool> getSchoolByName(String name) {
        String sql = "SELECT s.ID, s.NAME, p.POOL_SIZE " +
            "FROM SPORTS_SCHOOL s " +
            "JOIN POOL p ON s.POOL_ID = p.ID " +
            "WHERE s.NAME = ?";
        return template.query(sql, new Object[]{name}, new
        BeanPropertyRowMapper<>(Sportschool.class));
    }
}
```

## Front-end

### Страница авторизации:

Вход

Логин

Пароль

Роль

Войти


Еще не зарегистрировались?

### Профиль:

Профиль

Карточка

Аллаяров Игорь Олегович



Статус: Тренер

Возраст: 27 лет

Спортшкола: ССК

Спортсмены

Аллаяров Игорь Олегович

Аллаяров Игорь Олегович

Результаты спортсменов

Чемпионат ИТМО

Спортсмен	Дистанция	Результат	Ранк	FINA
Аллаяров Игорь Олегович	50 брасс	31.50	I	450

Чемпионат ИТМО

Спортсмен	Дистанция	Результат	Ранк	FINA
Аллаяров Игорь Олегович	50 брасс	31.50	I	450

Чемпионат ИТМО

Спортсмен	Дистанция	Результат	Ранк	FINA
Аллаяров Игорь Олегович	50 брасс	31.50	I	450

Страница тренировок:

Тренировки

Введите ФИО тренера

Поиск

Нет результатов

Популярные тренировки

Спортсмен - Аллаяров Игорь Олегович

4 по 50 кроль ускорение

4 по 50 кроль ускорение

4 по 50 кроль ускорение

Тренер - Каменьщиков Павел Валерьевич

Спортсмен - Аллаяров Игорь Олегович

4 по 50 кроль ускорение

4 по 50 кроль ускорение

4 по 50 кроль ускорение

Тренер - Каменьщиков Павел Валерьевич

Страница составления тренировок:

Мои тренировки

Тренировки

Добавить тренировку

Задание №1

0

Создать

+

-