САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа 2, Лабораторная работа 1
Работа с TypeORM, Реализация boilerplate

Выполнила:

Жижилева Арина

К3342

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача ДЗ2:

Реализовать все модели данных, спроектированные в рамках ДЗ1

Реализовать набор из CRUD-методов для работы с моделями данных средствами Express + TypeScript

Реализовать API-эндпоинт для получения пользователя по id/email

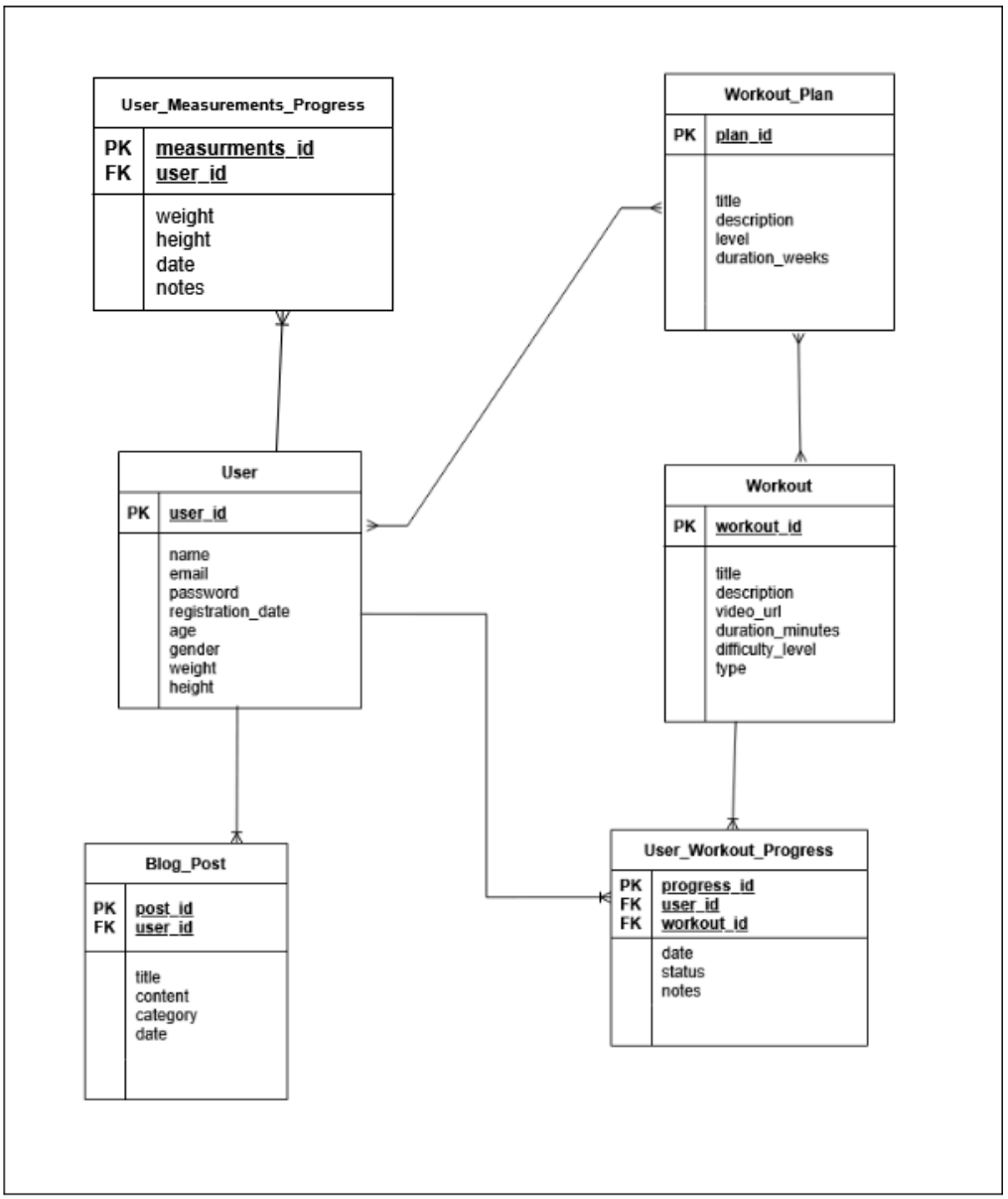
Задача ЛР1:

Нужно написать свой boilerplate на express + TypeORM + typescript.

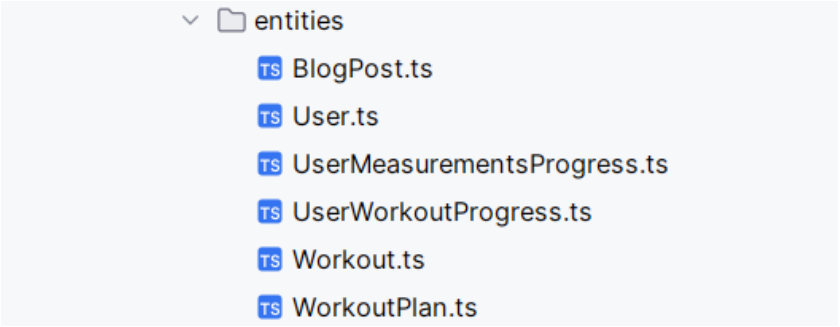
Должно быть явное разделение на:

- модели
- контроллеры
- роуты

Ход работы



Создание моделей



Пример кода из user.ts

```
import { Entity, PrimaryGeneratedColumn, Column, OneToMany } from "typeorm";
import { BlogPost } from "./BlogPost";
import { UserMeasurementsProgress } from "./UserMeasurementsProgress";
import { UserWorkoutProgress } from "./UserWorkoutProgress";

@Entity()
export class User {
 @PrimaryGeneratedColumn()
 user_id!: number;

 @Column()
 name!: string;

 @Column({ unique: true })
 email!: string;

 @Column()
 password!: string;

 @Column()
 registration_date!: Date;

 @Column()
 age!: number;

 @Column()
```

```
gender!: string;

@Column("float")
weight!: number;

@Column("float")
height!: number;

@OneToMany(() => BlogPost, (post) => post.user)
blogPosts!: BlogPost[];

@OneToMany(() => UserMeasurementsProgress, (progress) => progress.user)
measurementsProgress!: UserMeasurementsProgress[];

@OneToMany(() => UserWorkoutProgress, (progress) => progress.user)
workoutProgress!: UserWorkoutProgress[];
}
```

Установка зависимостей

npm install
npm install -g ts-node-dev typescript

Создание бд в postgres

1. через win+r services.msc запускаю postgres.
2. в cmd psql -U postgres
3. CREATE DATABASE fitness_app;

Файл для подключения к бд:

data-source.ts

```
import "reflect-metadata";
import { DataSource } from 'typeorm';
import { User } from "../entities/User";
import { BlogPost } from "../entities/BlogPost";
import { UserMeasurementsProgress } from "../entities/UserMeasurementsProgress";
import { WorkoutPlan } from "../entities/WorkoutPlan";
import { Workout } from "../entities/Workout";
import { UserWorkoutProgress } from "../entities/UserWorkoutProgress";

export const AppDataSource = new DataSource({
  type: "postgres",
  host: "localhost",
  port: 5432,
  username: "postgres",
  password: "Arina2992",
  database: "fitness_app",
  synchronize: true,
  logging: false,
  entities: [User, BlogPost, UserMeasurementsProgress, WorkoutPlan, Workout, UserWorkoutProgress],
  migrations: [],
  subscribers: [],
});
```

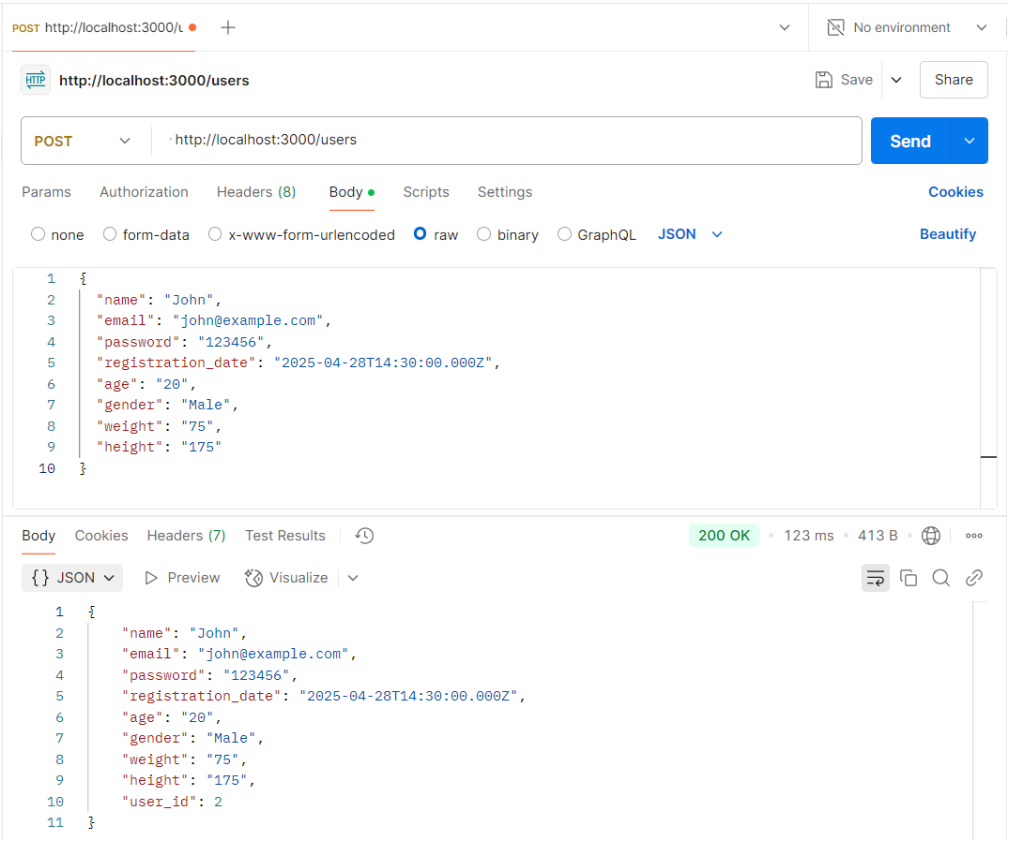
Запуск проекта в pycharm терминале командой npm run dev

```
postgres=# \c fitness_app
Вы подключены к базе данных "fitness_app" как пользователь "postgres".
fitness_app=# \dt
                     ╤яшёюъ юЄэю°хэшщ
 ╤їхьр  |            ╚ь              |   ╥шя    | ┬ырфхыхЎ
--------+----------------------------+----------+----------
 public | blog_post                  | Єрсышўр | postgres
 public | user                       | Єрсышўр | postgres
 public | user_measurements_progress | Єрсышўр | postgres
 public | user_workout_progress      | Єрсышўр | postgres
 public | workout                    | Єрсышўр | postgres
 public | workout_plan               | Єрсышўр | postgres
(6 ёЄЁюъ)
```

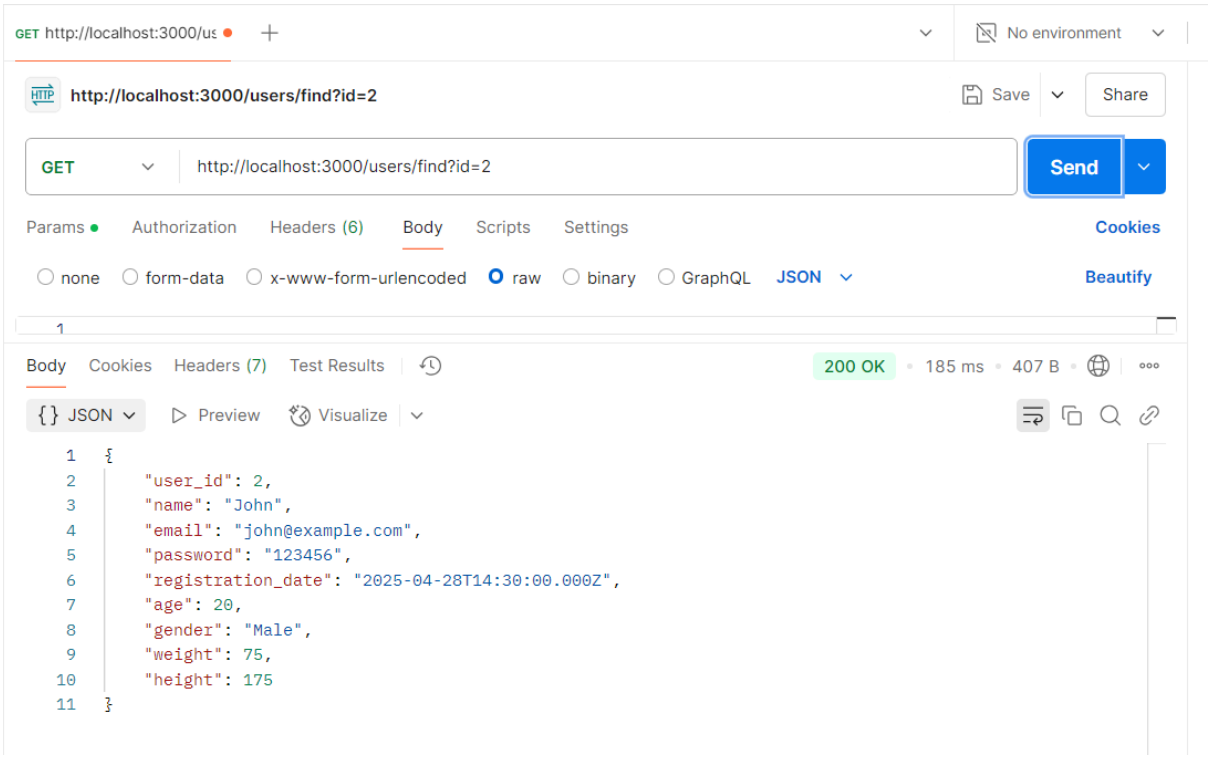
Файловая структура проекта:

```
src
  controllers
  database
  entities
  repositories
  routes
  scripts_postman
  app.ts
  server.ts
package.json
package-lock.json
tsconfig.json
```

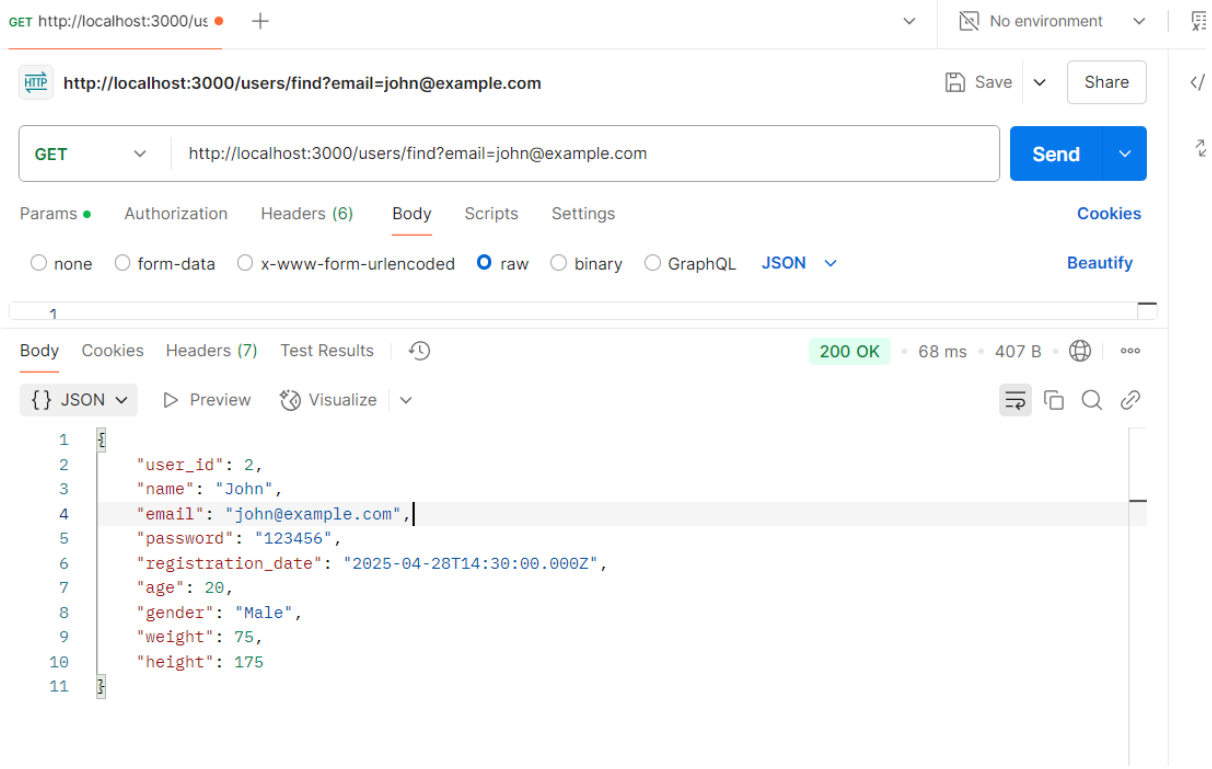
Регистрация



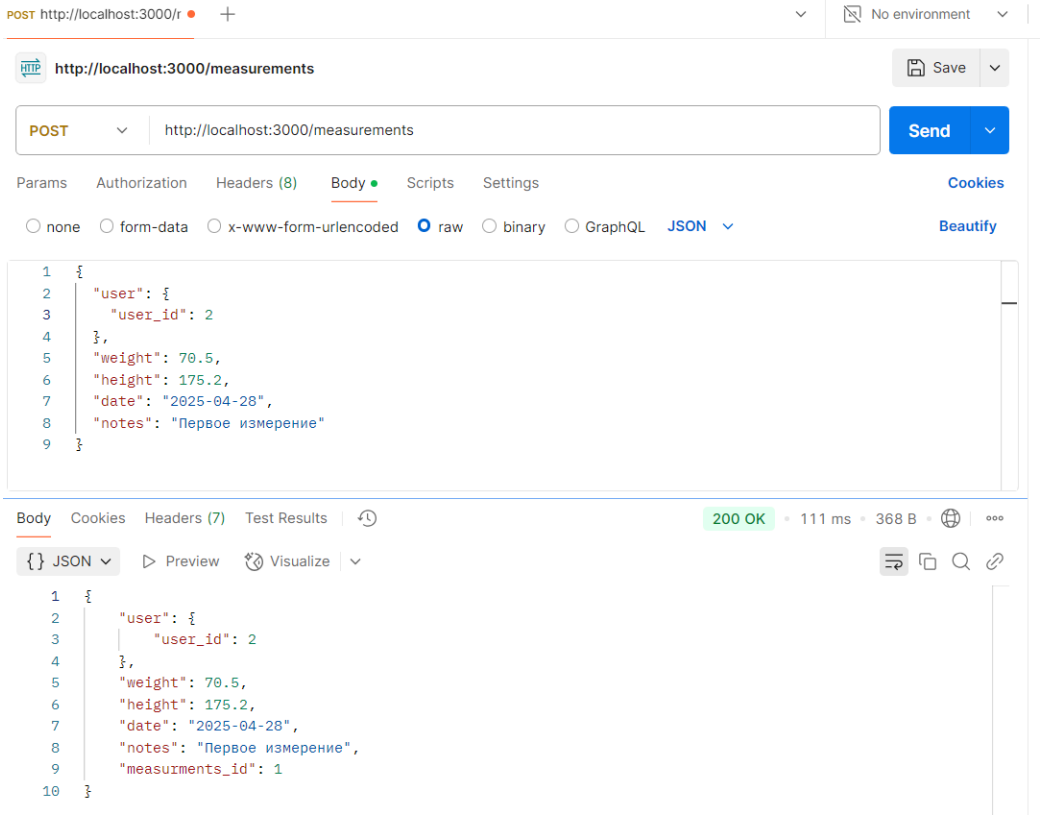
Получение пользователя по id:



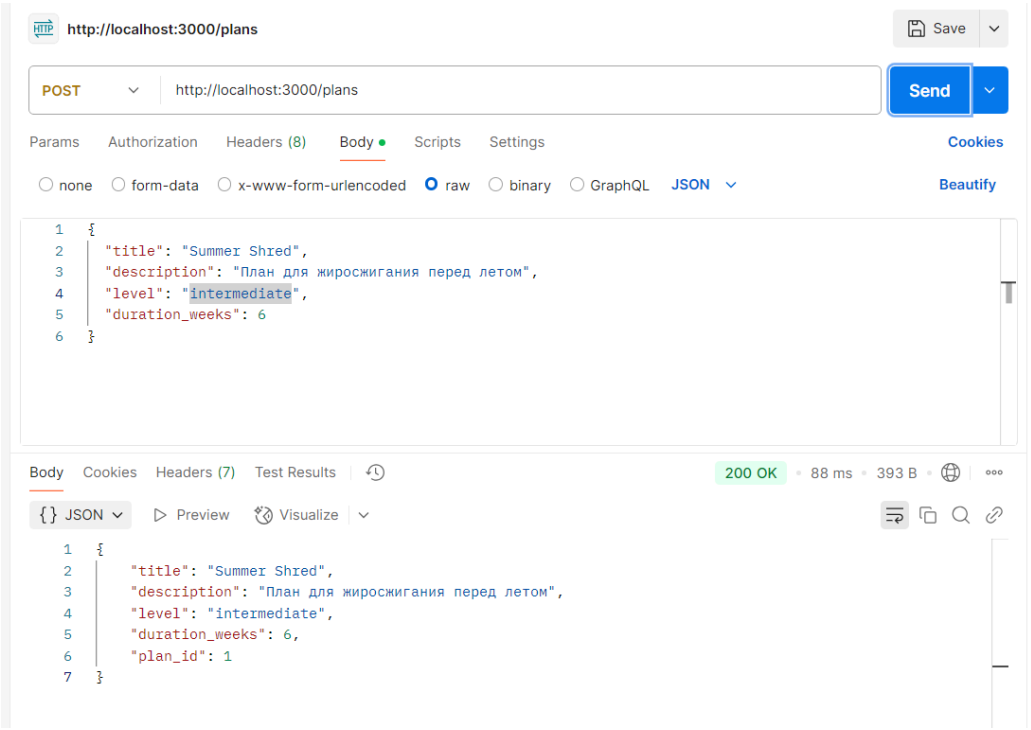
Получение пользователя по email:



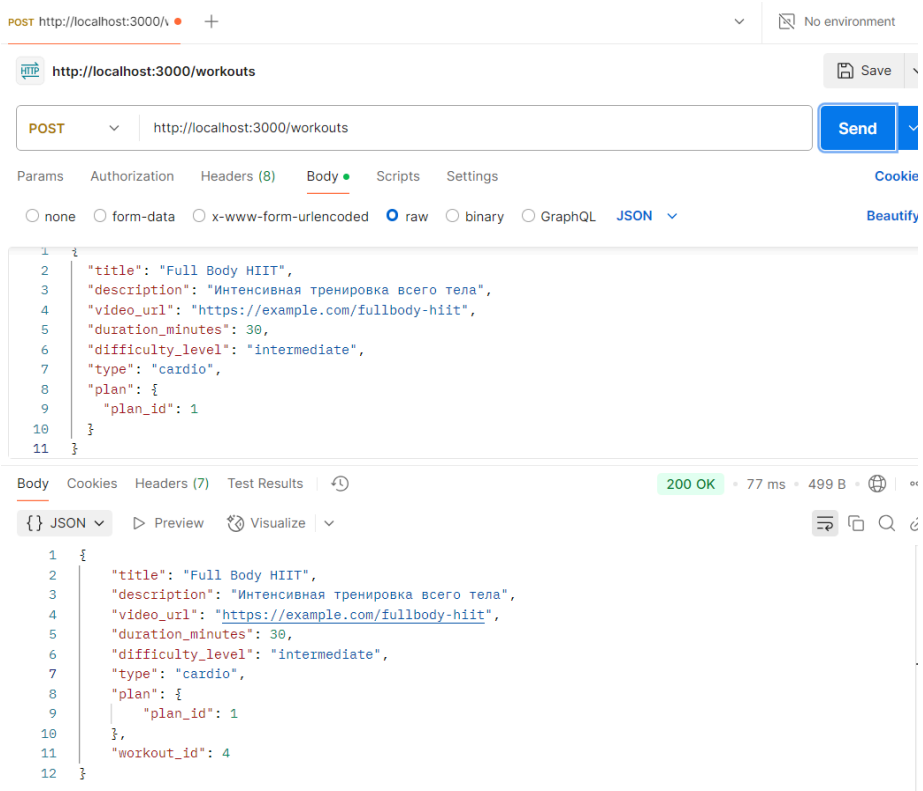
Measurements



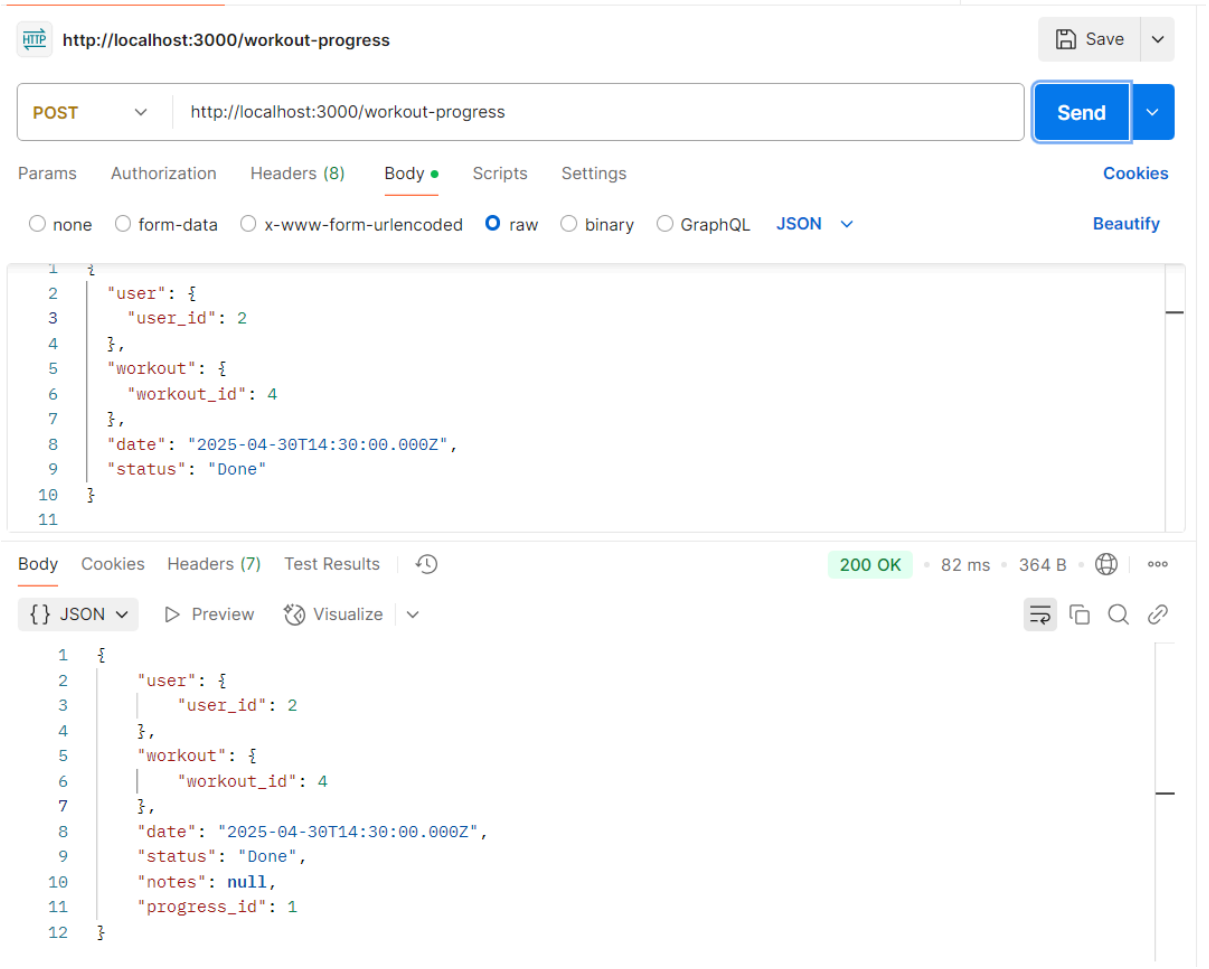
Создание плана тренировок



Создание тренировки



Отслеживание прогресса


http://localhost:3000/workout-progress
Save
POST
http://localhost:3000/workout-progress
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"user": {
"user_id": 2
},
"workout": {
"workout_id": 4
},
"date": "2025-04-30T14:30:00.000Z",
"status": "Done"
}
Body
Cookies
Headers (7)
Test Results
200 OK
82 ms
364 B
{} JSON
Preview
Visualize
{
"user": {
"user_id": 2
},
"workout": {
"workout_id": 4
},
"date": "2025-04-30T14:30:00.000Z",
"status": "Done",
"notes": null,
"progress_id": 1
}