

# Barbells

DOCUMENTAÇÃO

PROJETO FINAL DE BACK-END

MARIANA PEREIRA FIGUEIREDO TAVARES

# Sumário

- Planeamento do projeto
- Estrutura de dados
- Sobre a API
- Rotas
- Instalação
- Deploy

# Planeamento do projeto

Objetivo da aplicação

- Plataforma de registo de treinos e métricas pessoais
- O utilizador pode subscrever a plataforma e registar os treinos efetuados, medições do seu peso e da sua percentagem de gordura corporal

# Planeamento do projeto


## Definição dos requisitos funcionais

O utilizador deve poder

- Registar-se, autenticar-se, obter o seu perfil, terminar sessão e eliminar a sua conta
- Registar e gerir as suas pesagens e medições de gordura corporal e obter um histórico das mesmas
- Registar e gerir os seus treinos e obter um histórico dos mesmos
- Procurar treinos por intervalo de datas
- Obter o volume de treino por treino e por intervalo de datas e grupo muscular
- Criar novos exercícios e fazer a gestão dos mesmos
- Procurar exercícios por nome
- Filtrar exercícios por grupo muscular
- Criar templates de treino e fazer a gestão das mesmas

# Planeamento do projeto

Definição das regras de negócio

O utilizador não deve poder 

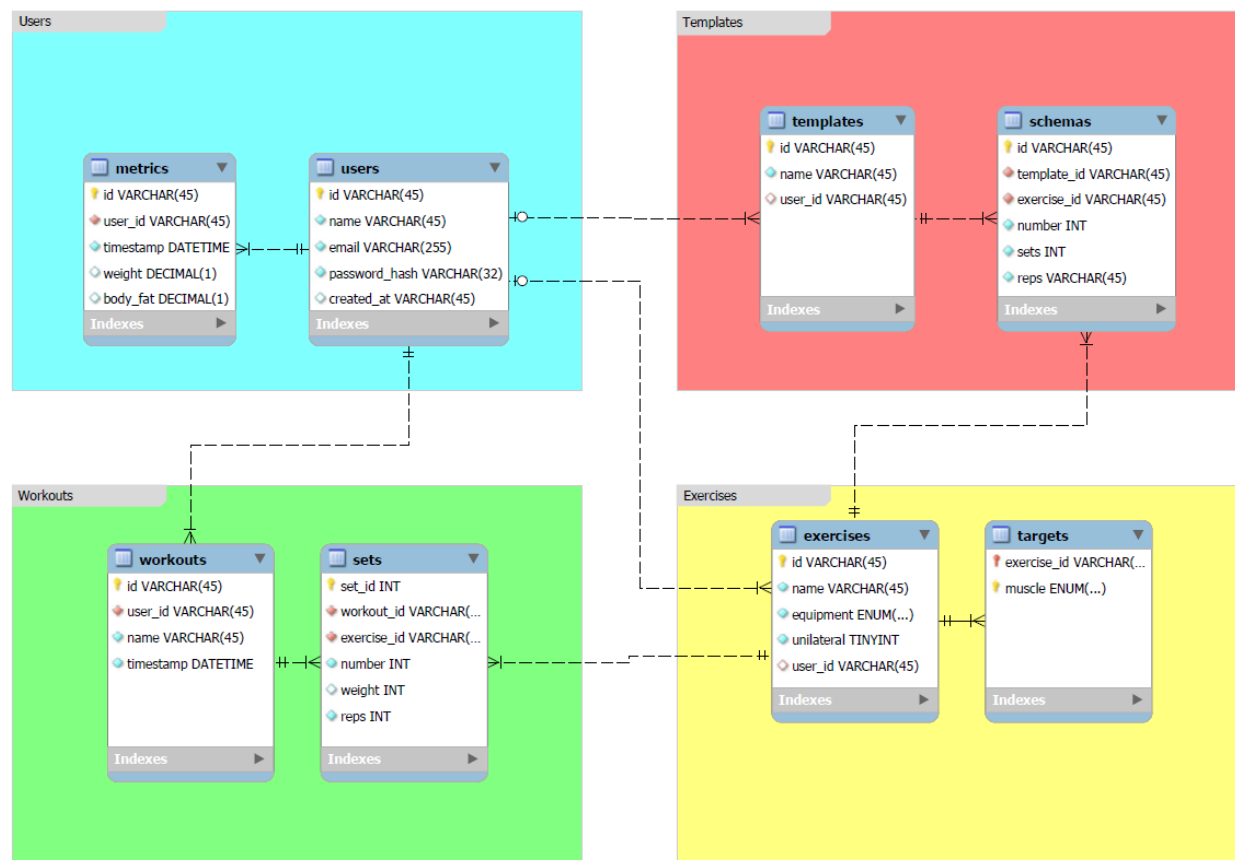
- Registrar-se com um email duplicado
- Criar um exercício duplicado
- Criar uma template de treino duplicada
- Editar ou eliminar exercícios e templates predefinidos da aplicação
- Obter e gerir dados de outros utilizadores

# Planeamento do projeto

Escolha das tecnologias a utilizar

- NodeJS
- Fastify
- Prisma
- Zod
- Vitest
- Supertest
- Docker

# Estrutura de dados



# Sobre a API

- Todas as rotas (exceto a de registo e de login) requerem autenticação
  - ✓ Enviar token de autenticação obtido na rota de login no Authorization Bearer Header
- Todas as rotas respondem em formato JSON



usersRoutes

metricsRoutes

workoutsRoutes

setsRoutes

exercisesRoutes

templatesRoutes

schemasRoutes

/register

- Cria um novo utilizador
- method: 'POST'

```
{  
  name: string,  
  email: string,  
  password: string (min 6 caracteres)  
}
```

### /login

- Autentica o utilizador
- Retorna o token de autenticação
- method: 'POST'

```
{  
  email: string,  
  password: string (min 6 caracteres)  
}
```

### /logout

- Termina sessão do utilizador
- method: 'POST'

/me

- Obtém conta do utilizador autenticado
- method: 'GET'
- Retorna user:

```
{  
  id: string;  
  name: string;  
  email: string;  
  createdAt: Date;  
}
```

/me

- Elimina conta do utilizador autenticado
- method: 'DELETE'

/metrics

- Cria um registo de peso ou gordura corporal
- method: 'POST'

```
{  
  timestamp: date (opcional: adiciona dia e hora atual por defeito)  
  weight: number (opcional)  
  bodyFat: number (opcional)  
}
```

### /metrics/history

- Obtém o histórico dos registo de peso e gordura corporal do utilizador autenticado
- method: 'GET'
- Retorna metrics:

```
{  
  id: string;  
  userId: string;  
  timestamp: Date;  
  weight: number | null;  
  bodyFat: number | null;  
}[]
```

### /metrics/history

- Apaga todo o histórico dos registo de peso e gordura corporal do utilizador autenticado
- method: 'DELETE'

/metrics/:metricsId

- Obtém registo de peso/ou e gordura corporal por id através de rota dinâmica
- method: 'GET'
- Retorna metric:

```
{  
  id: string;  
  userId: string;  
  timestamp: Date;  
  weight: number | null;  
  bodyFat: number | null;  
}
```

/metrics/:metricId

- Edita registo de peso e/ou gordura corporal por id através de rota dinâmica
- method: 'PUT'

```
{  
  timestamp: date (opcional: adiciona dia e hora atual por defeito)  
  weight: number (opcional)  
  bodyFat: number (opcional)  
}
```

/metrics/:metricsId

- Apaga registo de peso e/ou gordura corporal por id através de rota dinâmica
- method: 'DELETE'



## /workouts

- Regista um treino
- Cria o treino e os respetivos sets
- method: 'POST'

```
{
  name: string (opcional),
  timestamp: date (opcional: adiciona dia e hora atual por defeito),
  sets: {
    create: {
      number: number (número de ordem de cada set)
      exerciseId: string (uuid),
      weight: number (opcional),
      reps: number()
    }[]
  }
}
```

### /workouts/history

- Obtém histórico de treinos
- method: 'GET'
- Retorna workouts:

```
{  
  id: string;  
  name: string | null;  
  timestamp: Date;  
  userId: string;  
}[]
```

### /workouts/history

- Elimina todo o histórico de treinos
- method: 'DELETE'

### /workouts/total

- Obtém número total de treinos realizados
- method: 'GET'
- Retorna count: number

### /workouts/:workoutId

- Obtém um treino por id através de rota dinâmica
- method: 'GET'
- Retorna workout:

```
{  
  id: string;  
  name: string | null;  
  timestamp: Date;  
  userId: string;  
}
```

/workouts/:workoutId

- Edita um treino por id através de rota dinâmica
- Os sets de treino possuem rota própria para sua gestão
- method: 'PUT'

```
{  
  name: string (opcional)  
  timestamp: date (adiciona dia e hora atual por defeito)  
}
```

/workouts/:workoutId

- Apaga um treino por id através de rota dinâmica
- method: 'DELETE'

/workouts/:workoutId/volume?muscle=

- Obtém o volume total de um treino por id através de rota dinâmica
- method: 'GET'
- Retorna volume: number
- Obtém o volume de um treino por grupo muscular se submetido via query

muscle: 'abs' | 'back' | 'biceps' | 'calfs' | 'chest' | 'glutes' | 'hamstrings' | 'quadriceps' | 'shoulders' | 'triceps'

/workouts/search-by-date?from=&to=&muscle=

- Obtém os treinos realizados por intervalo de tempo
- method: 'GET'
- Retorna workouts:

```
{  
  id: string;  
  name: string | null;  
  timestamp: Date;  
  userId: string;  
}[]
```

- Data final e inicial submetida via query

from: string

to: string

/workouts/search-by-date/volume?from=&to=&muscle=

- Obtém o volume de treino total por intervalo de tempo
- method: 'GET'
- Retorna volume: number
- Obtém o volume de treino total por intervalo de tempo e por grupo muscular se submetido via query

from: string

to: string

muscle: 'abs' | 'back' | 'biceps' | 'calfs' | 'chest' | 'glutes' | 'hamstrings' | ' quadriceps' | 'shoulders' | ' triceps'

## Rotas para atualizar o conteúdo de um treino

/sets

- Cria um novo set
- method: 'POST'

```
{  
  workoutId: string (uuid),  
  exerciseId: string (uuid),  
  number: number (número de ordem de cada set)  
  weight: number (opcional)  
  reps: number  
}
```



## Rotas para atualizar o conteúdo de um treino

/sets/:setId

- Obtém um set por id através de rota dinâmica
- method: 'GET'
- Retorna set:

```
{  
  id: string;  
  workoutId: string;  
  exerciseId: string;  
  number: number (número de ordem do set);  
  weight: number | null;  
  reps: number;  
}
```

## Rotas para atualizar o conteúdo de um treino

/sets/:setId

- Edita um set por id através de rota dinâmica
- method: 'PUT'
  - {
  - number: number (número de ordem do set)
  - weight: number (opcional)
  - reps: number
  - }

/sets/:setId

- Apaga um set por id através de rota dinâmica
- method: 'DELETE'

## /exercises

- Cria um exercício com os respetivos targets (músculos-alvo)
- method: 'POST'

```
{
  name: string,
  equipment: 'assisted' | 'barbell' | 'bodyweight' | 'cable' | 'machine',
  unilateral: boolean,
  targets: {
    create: {
      muscle: 'abs' | 'back' | 'biceps' | 'calves' | 'chest' | 'glutes' | 'hamstrings' |
'quadriceps' | 'shoulders' | 'triceps'
    }[]
  }
}
```

/exercises/:exerciseld

- Edita um exercício por id através de rota dinâmica

- method: 'PUT'

```
{  
  name: string,  
  equipment: 'assisted' | 'barbell' | 'bodyweight' | 'cable' | 'machine',  
  unilateral: boolean  
}
```

/exercises/:exerciseld

- Apaga um exercício por id através de rota dinâmica

- method: 'DELETE'

/exercises/:exerciseld

- Obtém um exercício por id através de rota dinâmica
- method: 'GET'
- Retorna exercise:

```
{  
  id: string;  
  name: string;  
  equipment: 'assisted' | 'barbell' | 'bodyweight' | 'cable' | 'machine';  
  unilateral: boolean;  
  userId: string | null;  
}
```

/exercises/all

- Obtém todos os exercícios
- method: 'GET'
- Retorna exercises:

```
{  
  id: string;  
  name: string;  
  equipment: $Enums.Equipment;  
  unilateral: boolean;  
  userId: string | null;  
}[]
```

/exercises/search-by-name?query=

- Procura exercícios por nome via query
- method: 'GET'

query: string

- Retorna exercises:

```
{  
  id: string;  
  name: string;  
  equipment: $Enums.Equipment;  
  unilateral: boolean;  
  userId: string | null;  
}[]
```

/exercises/search-by-target?muscle=

- Procura exercícios por grupo muscular via query
- method: 'GET'

muscle: 'abs' | 'back' | 'biceps' | 'calfs' | 'chest' | 'glutes' | 'hamstrings' | ' quadriceps' | 'shoulders' |  
' triceps'

- Retorna exercises:

```
{  
  id: string;  
  name: string;  
  equipment: $Enums.Equipment;  
  unilateral: boolean;  
  userId: string | null;  
}[]
```





templatesRoutes []

/templates

- Cria um template
- Cria o treino e os respetivos schemas (sets de treino de uma template)
- method: 'POST'

```
{
  name: string,
  schemas: {
    create: {
      number: number (número de ordem de cada exercício)
      exerciseld: string (uuid)
      sets: number,
      reps: string
    }[]
  }
}
```

/templates/all

- Obtém todos os templates
- method: 'GET'
- Retorna templates:

```
{  
  id: string;  
  name: string;  
  userId: string | null;  
} []
```

/templates/:templateId

- Obtém um template por id através de rota dinâmica
- method: 'GET'
- Retorna template:

```
{  
  id: string;  
  name: string;  
  userId: string | null;  
} []
```

/templates/:templateId

- Edita um template por id através de rota dinâmica
- Os schemas possuem rota própria para sua gestão
- method: 'PUT'

```
{  
  name: string  
}
```

/templates/:templateId

- Apaga um template por id através de rota dinâmica
- method: 'DELETE'

## Rotas para atualizar o conteúdo de um template

/schemas

- Cria um novo schema
- method: 'POST'

```
{  
  number: number (número de ordem do exercício)  
  templateId: string (uuid),  
  exerciseId: string (uuid),  
  sets: number,  
  reps: string  
}
```

/schemas/:schemaId

- Obtém um schema por id através de rota dinâmica
- method: 'GET'

## Rotas para atualizar o conteúdo de um template

/schemas/:schemald

- Edita um schema por id através de rota dinâmica
  - method: 'PUT'
- ```
{  
  number: number (número de ordem do exercício)  
  exerciseld: string (uuid),  
  sets: number,  
  reps: string  
}
```

/schemas/:schemald

- Apaga um schema por id através de rota dinâmica
- method: 'DELETE'

# Instalação

<https://github.com/mpftavares/barbells-api>

1. Instalar as dependências do projeto
  - `npm i`
2. Criar e configurar ficheiro `.env`
  - `PORT,JWT_SECRET,DATABASE_URL`
3. Criar container
  - `npm run docker`
4. Criar base de dados
  - `npm run migrate`
5. Iniciar o servidor
  - `npm run start`

**Deploy** 

Rotas disponíveis em:

<https://barbells-p120.onrender.com/>