

# *Aprofundamento em Express.js*

Labenu\_



# Sumário

Labenu\_



O que vamos  
ver hoje? 🙄

- Ordenação de rotas
- Documentação



# Relembrando...

Labenu\_



# Introdução

- Na aula de introdução a API's e Express.js, revimos a API do Labefy e entendemos as **exigências** de cada endpoint
- Conhecemos a lib Express.js e como ela nos ajuda a **criar** endpoints
- Também vimos a **sintaxe** usada pelo Express.js e as principais propriedades dos parâmetros **req** e **res**
- Hoje, vamos conhecer alguns detalhes do Express.js e construção de endpoints, além da importância de uma boa **documentação**



# Express: Ordenação das Rotas

Labenu\_



# Ordenação das Rotas

Observe os endpoints a seguir e tente imaginar: qual seria a resposta da requisição **http://localhost:3003/test/hello** ?

```
app.get('/test/:number', (req: Request, res: Response) => {  
  res.send(  
    `Seu número da sorte é ${Number(req.params.number) + 3}!`  
  )  
})  
  
app.get('/test/hello', (req: Request, res: Response) => {  
  res.send(`Olá, mundo!`)  
})
```



# Ordenação das Rotas

- Embora a resposta desejada seja "Olá, mundo", o que recebemos no lugar é "Seu número da sorte é NaN!"
- Se estivéssemos desenvolvendo uma API complexa, poderíamos demorar muito tempo para perceber que nossa requisição está sendo enviada para o endpoint errado
- Isso acontece porque o nosso servidor, ao receber uma requisição, percorre todos os endpoints, na ordem em que foram declarados, procurando por uma correspondência



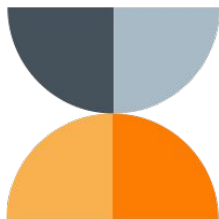


# Ordenação das Rotas

Para evitar erros como esse, devemos simplesmente priorizar a declaração de endpoints cujos caminhos possam ser confundidos com *path parameters*

```
app.get('/test/hello', (req: Request, res: Response) => {  
    res.send(`Olá, mundo!`)  
})  
  
app.get('/test/:number', (req: Request, res: Response) => {  
    res.send(  
        `Seu número da sorte é ${Number(req.params.number) + 3}!`  
    )  
})
```





- O *express*, ao receber uma requisição, percorre todos os endpoints, na ordem em que foram declarados, procurando por uma correspondência
- Devemos priorizar a declaração de endpoints cujos caminhos possam ser confundidos com *path parameters*



# Exercício 1

Na aula de hoje, vamos completar nosso clone do Labefy com os seguintes endpoints:

- Search Playlist
- Create Playlist
- Add Track to Playlist

# Documentação

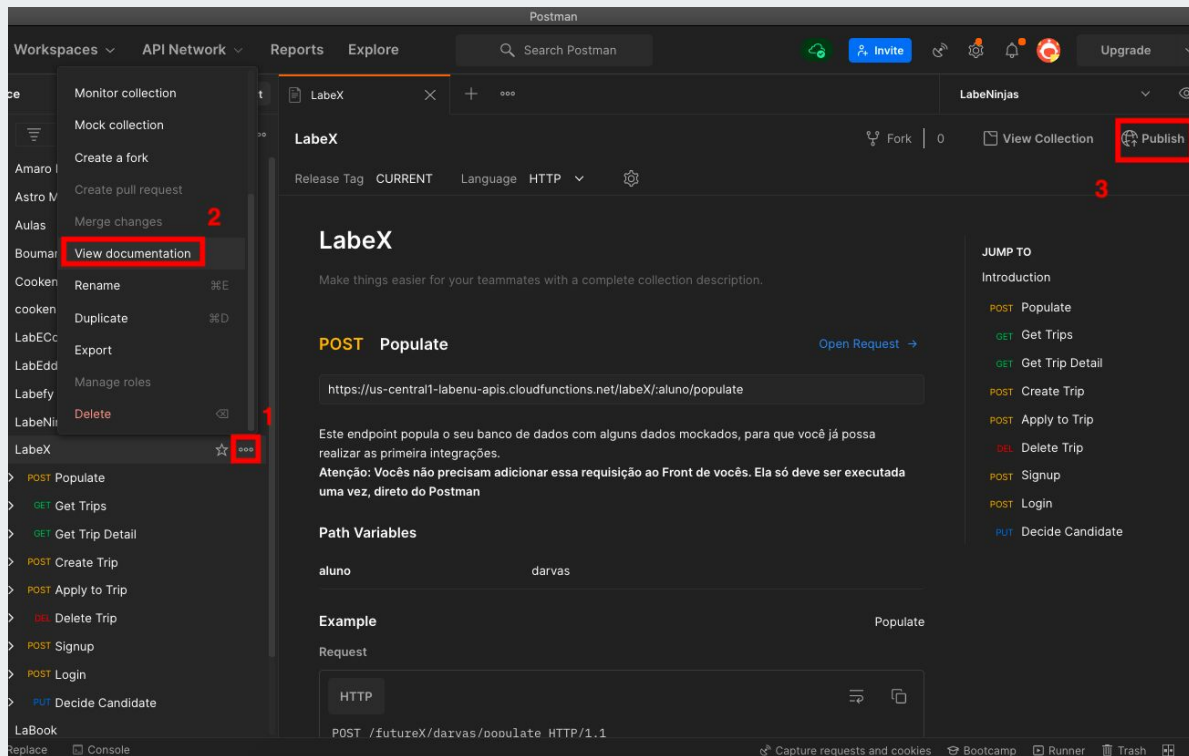
Labenu\_



# Documentação 📜

- Etapa crucial no desenvolvimento de qualquer aplicação
- Pode ser feita com as mais variadas ferramentas, desde comentários no código até interfaces gráficas completas
- No Postman, uma coleção de requisições pode ser publicada com poucos cliques

# Documentação



The screenshot displays the Postman application interface. On the left sidebar, a context menu is open for the 'LabeX' collection, with the 'View documentation' option highlighted by a red box and labeled with a red '2'. The 'LabeX' collection is selected in the top bar, and its 'Publish' button is also highlighted with a red box and labeled with a red '3'. The main panel shows the 'LabeX' collection details, including a description, a 'POST Populate' endpoint, and a 'Path Variables' section. The bottom status bar indicates the current request is a 'POST /futureX/darvas/populate HTTP/1.1'.

Labenu\_

# Documentação de APIs

- A documentação da API funciona como um **manual de uso**, ou seja, ela é um **instrumento crucial** para devs e outros profissionais da tecnologia
- Esse documento deve descrever **como utilizar a API**, com exemplos, instruções de autenticação e descrição de cada funcionalidade (e suas exigências), entre outros
- Devemos tomar cuidado com ambiguidades, inconsistências, documentação desatualizada e/ou incompleta
- Documentação de má qualidade é um **obstáculo** para o consumo da API
- Lembre-se: a documentação é um **acordo** entre quem disponibiliza a API e quem a consome

# Documentação de APIs 📜

Alguns exemplos de boas documentações:

- [PayPal](#)
- [Spotify](#)
- [Twitter](#)
- [Notion](#)
- [Gmail](#)
- [GitHub](#)



## Exercício 2

Bóra praticar? Vamos ver como publicar a documentação da nossa API com o **Postman**!

Obs: o próprio Postman fornece um [guia](#) para documentar nossas APIs (em inglês)

# Resumo

Labenu\_



# Resumo



- Devemos **priorizar** a declaração de endpoints que possam ser confundidos com *path parameters* (primeiro o específico, depois o geral)
- A documentação das APIs é essencial para que elas sejam consumidas e testadas sem maiores dificuldades



# Dúvidas?





Obrigada(o)!