06

Do as I did: virtualization best practices

`FastAPI` is a modern and fast framework for building web APIs with Python. With it, you create a variety of applications and services, especially focused on APIs. When developing applications using the framework `FastAPI`, it is common to interact with external services through HTTP requests.

Let's define a simple endpoint that returns a greeting when accessed?

```
from fastapi import FastAPI, Query
import requests

app = FastAPI()

@app.get('/api/hello')
def hello_world():
    '''
    Endpoint que exibe uma mensagem incrível do mundo da program

    '''
    return {'Hello':'World'}
```

COP

Here, we define a simple endpoint that returns a greeting when accessed.

```
@app.get('/api/restaurantes/')
def get_restaurantes(restaurante: str = Query(None)):
    '''
    Endpoint para ver os cardápios dos restaurantes

    '''
    url = 'https://guilhermeonrails.github.io/api-restaurantes/r
    response = requests.get(url)
```

COP

In the next endpoint, we make a request to the Restaurant API. If the response is successf 200), the code loops through each item in this object, representing a restaurant, and extra specific data, such as the restaurant name (Company), the item offered (Item), the price (the description (description).

```
    if response.status_code == 200:
        dados_json = response.json()
        if restaurante is None:
            return {'Dados':dados_json}

        dados_restaurante = []
        for item in dados_json:
            if item['Company'] == restaurante:
                dados_restaurante.append({
                    "item": item['Item'],
                    "price": item['price'],
                    "description": item['description']
                })
        return {'Restaurante':restaurante,'Cardapio':dados_resta
    else:
        return {'Erro':f'{response.status_code} - {response.text
```

COP

If the response is 200, indicating that the request was successful, the code loops through in the object `dados_json` and extracts the desired data for the specific restaurant provide parameter. The result is then returned in the form of a dictionary containing the name of restaurant (Restaurant) and its menu (Menu). If the request is not successful, the code ret dictionary indicating the error that occurred, including the status code and the response

```
else:
    return {'Erro':f'{response.status_code} - {response.text}'}
```

COP

Remember: it's important to check the `status code` , process the data, and return spe information based on the parameters provided. FastAPI's versatility makes it a solid ch a variety of use cases, providing high performance and an enjoyable Python developm experience.