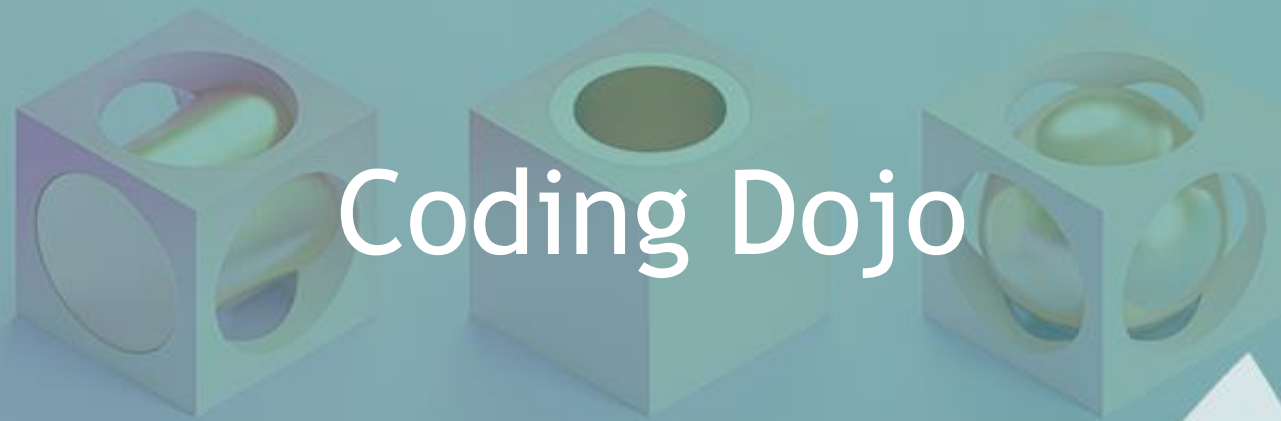


CIÊNCIA DA
COMPUTAÇÃO



Coding Dojo

ALEXANDRA RAIBOLT

Dojo

- A palavra *Dojo* possui origem japonesa e seu significado pode ser entendido como “*local de treinamento*”.



Dojo



CODING DOJO



Coding Dojo

- Quando nos referimos ao *Coding Dojo*, significa que estamos nos referindo a um “*local de treinamento de código*”.

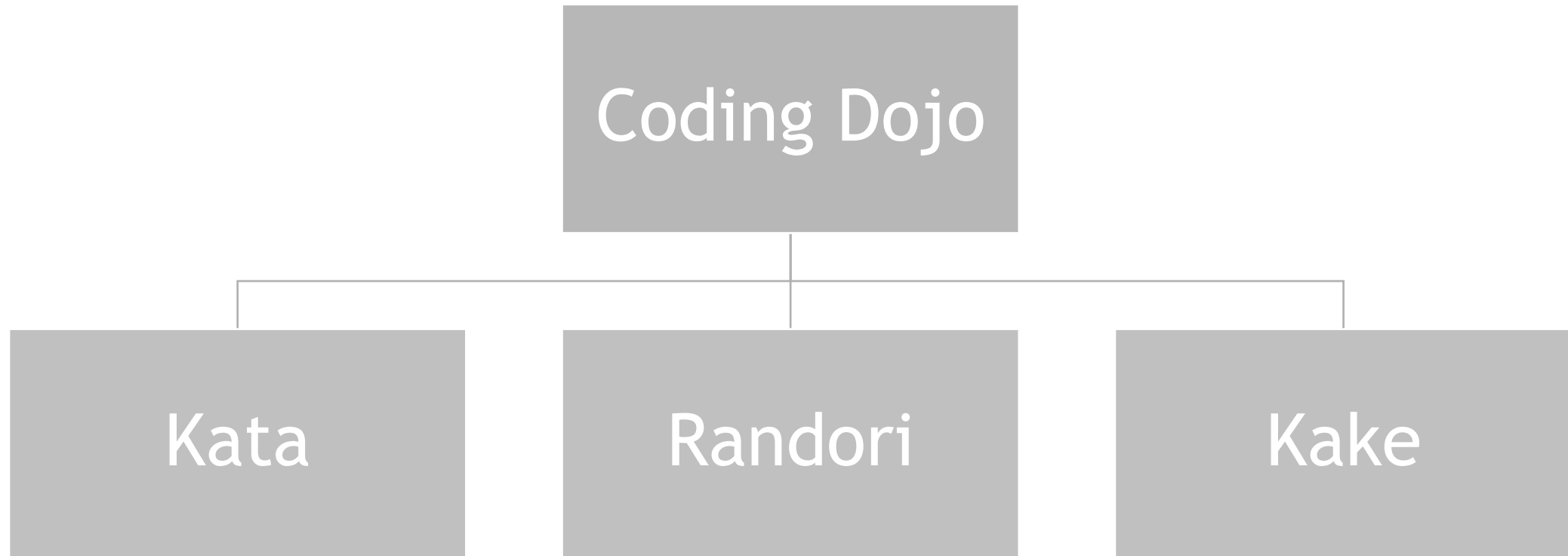


Coding Dojo

- O **Coding Dojo** (BACHE, 2013) é uma metodologia para o desenvolvimento de desafios e projeto em programação.
- **Pilares:**
 - **Coletividade** para **atingir** a **resolução** de um dado problema;
 - **Integração** entre **participantes**, e;
 - **Compartilhamento** de **ideias** e **conhecimentos**.

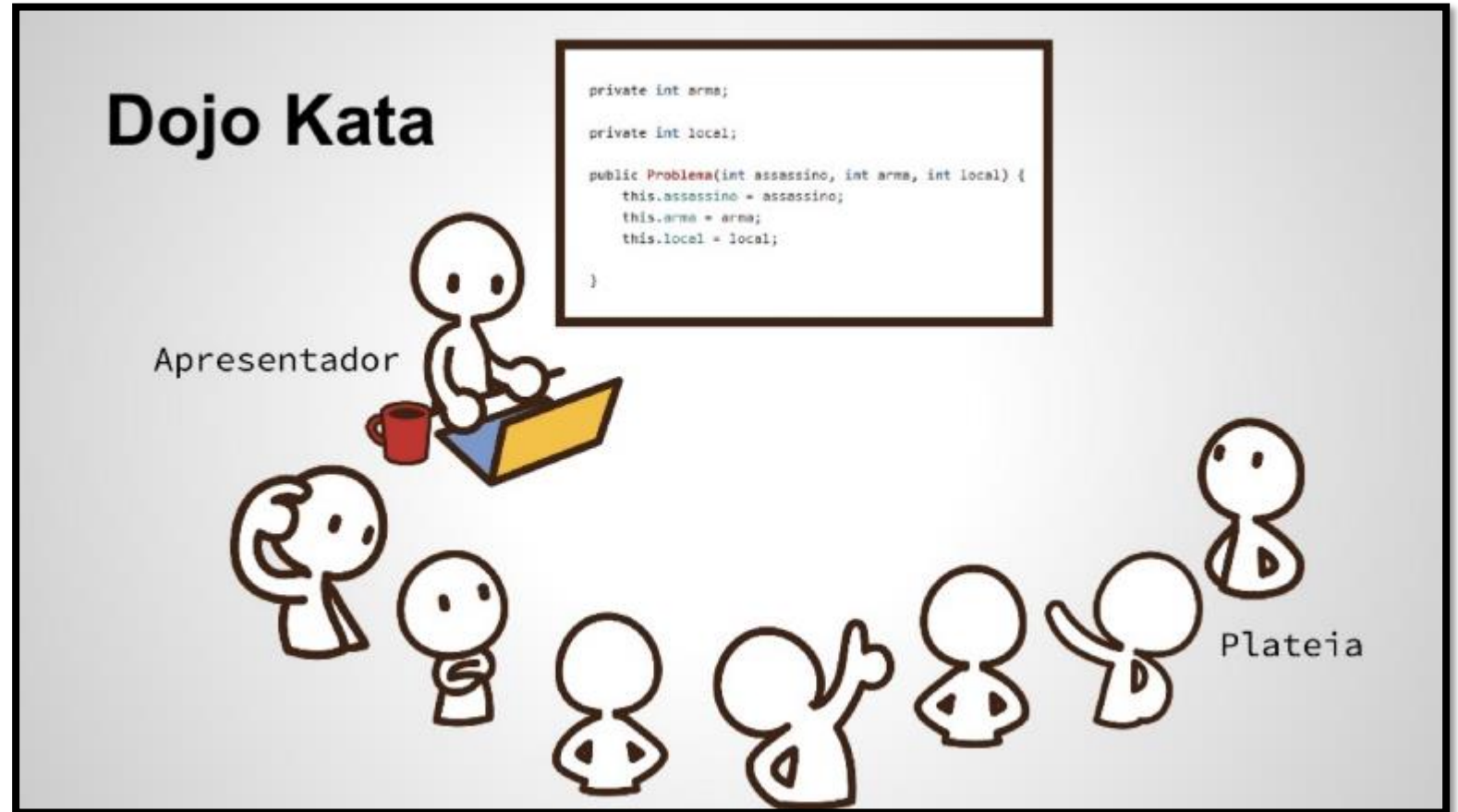


Coding Dojo

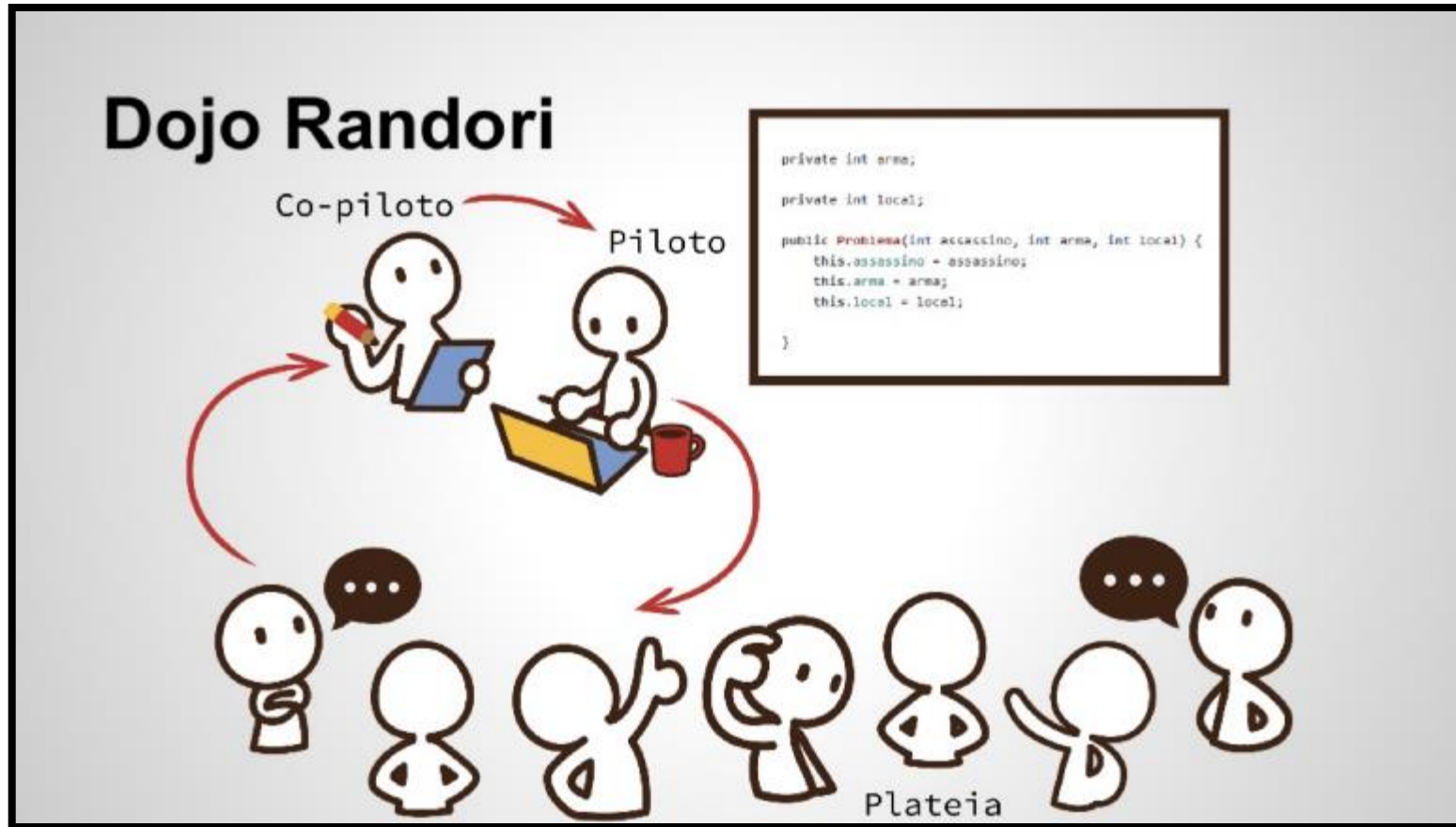


Coding Dojo: modalidade (*Prepared*) Kata

- **Indicado** para **introduzir** novos **conhecimento**.
- A **plateia** tem uma **participação** mais **passiva**.



Coding Dojo: modalidade Randori

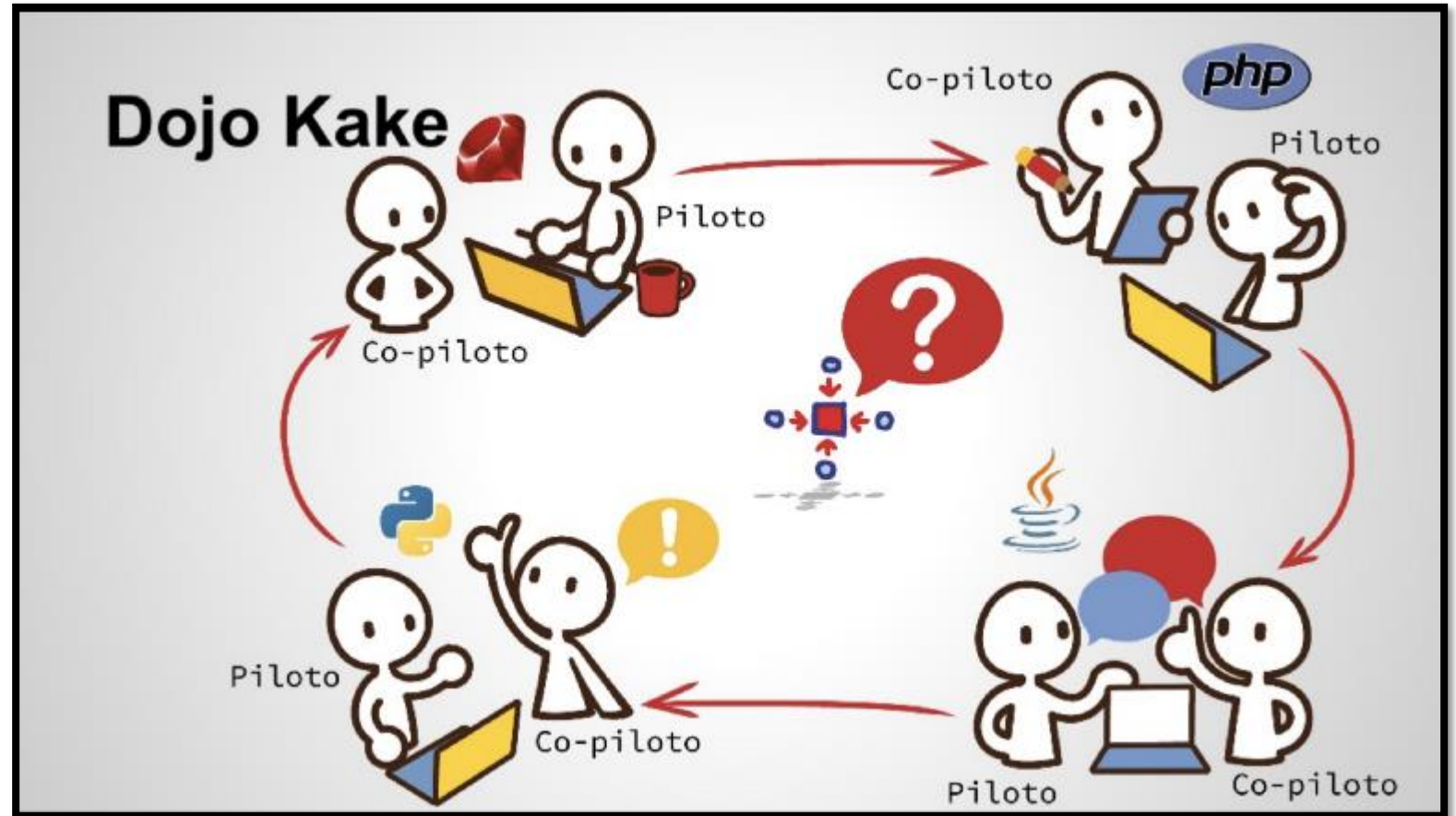


- Praticar conhecimentos já aprendidos.
- Todos participam ativamente na programação.



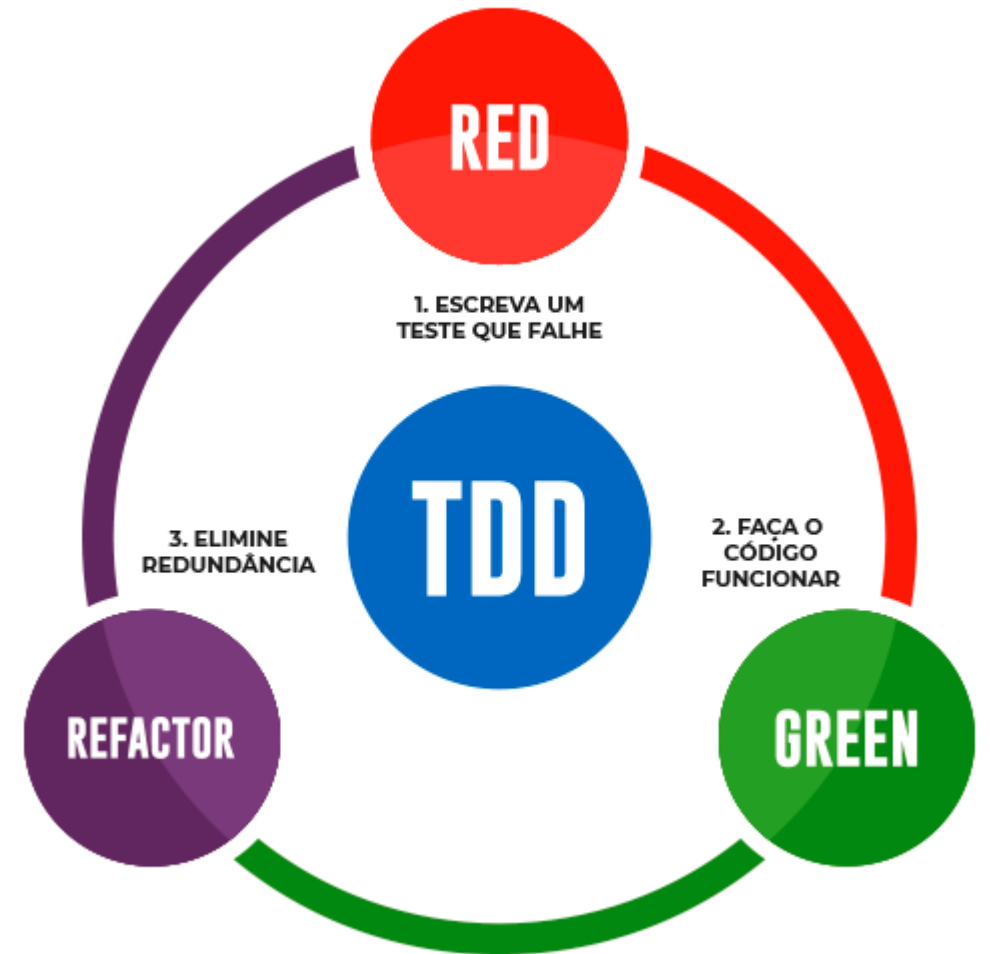
Coding Dojo: modalidade Kake

- Muita troca de **experiência** entre os **participantes**.
- Diversas **linguagens diferentes** para a mesma **solução**.



Desenvolvimento Orientado por Testes

- **TDD** (em inglês, *Test Driven Development*).
- **Desenvolvimento** baseado em **testes**.



Desafio



```
list = [11, 3, 7, 5, 2]

list.sort()

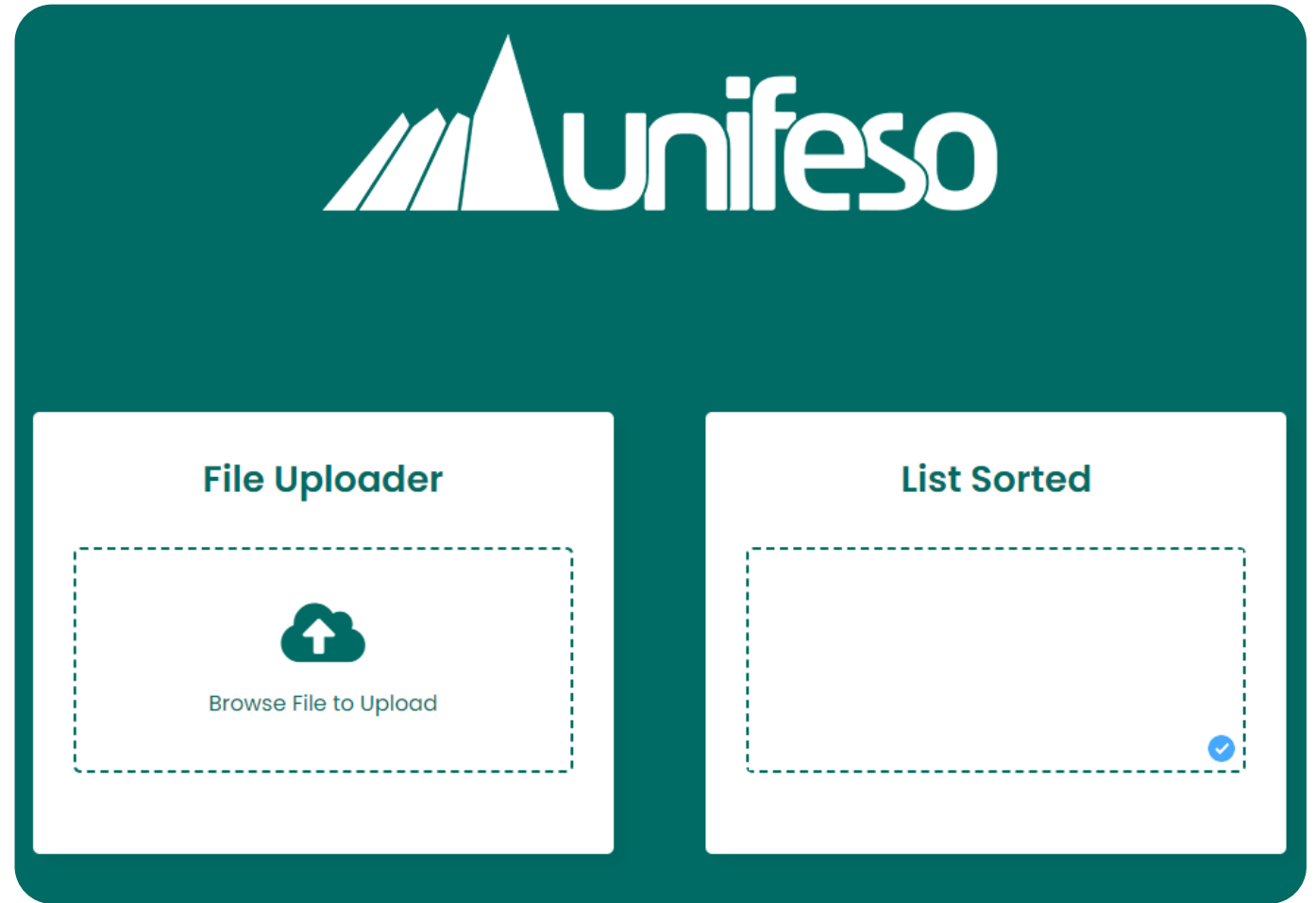
print(list)
# Saída: [2, 3, 5, 7, 11]
```

- Você **criou** em **Python** um **algoritmo** para classificar em ordem **crescente** **listas**.



Desafio

- Mas seu **colega desenvolveu** uma interface em **HTML, CSS e JavaScript** para receber as **listas** a serem **ordenadas**.



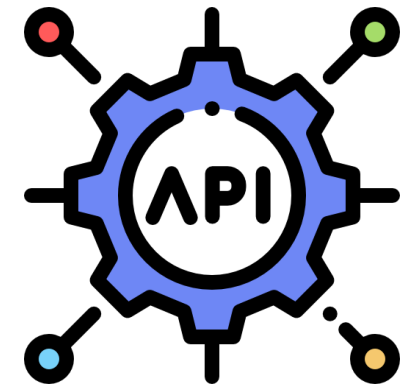
Desafio

- Como **integrar** o seu **algoritmo Python** com o aplicativo **Web desenvolvida** pelo seu **colega**?



Desafio

- Como **integrar** o seu **algoritmo Python** com o aplicativo **Web desenvolvida** pelo seu **colega**?
- **Felizmente**, temos o **poder** das **APIs**.



Application Programming Interface

- **Interface de Programação de Aplicações** (em inglês, *Application Programming Interface* – **API**).
- É um **conjunto de normas** que **possibilita** a **comunicação** entre **plataformas** através de uma série de **padrões** e **protocolos**.



Application Programming Interface

- Permite que **dois** ou **mais componentes** de **software** se **comuniquem** através destes **padrões** e **protocolos**.
- **Destina-se** a **integrar sistemas** e **facilitar** a **comunicação** entre **aplicações** de **software**.
- Permitindo a **reutilização** das suas **funcionalidades** por outras **aplicações** ou **software**.



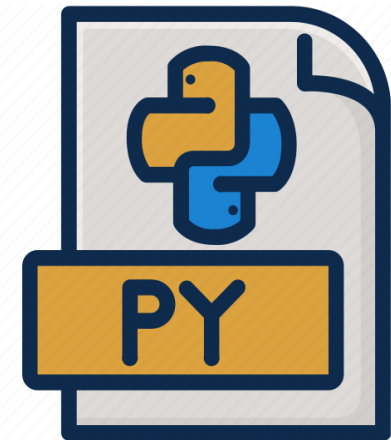
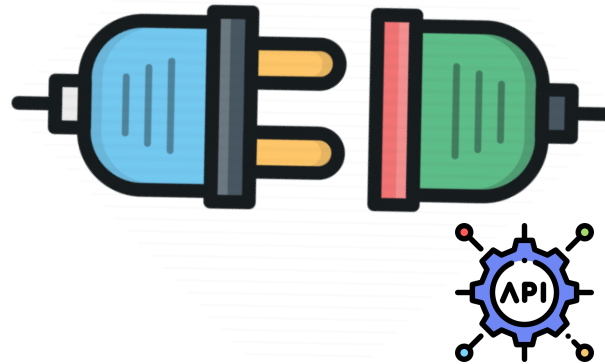
Application Programming Interface

- Muito utilizada para trocar dados entre diferentes tipos de softwares a fim de automatizar procedimentos e desenvolver novas funcionalidades.



Application Programming Interface

- Uma **API** é uma espécie de tomada que liga diferentes tipos de softwares ou aplicações e pode ser criada em várias linguagens de programação.



Application Programming Interface

- É frequentemente utilizada um formato de dados predefinido para compartilhar informação entre sistemas, com o objetivo de obter a integração entre eles.
- Os mais usados são:
 - **YAML** (*Yet Another Markup Language*).
 - **XML** (*Extensible Markup Language*).
 - **JSON** (*JavaScript Object Notation*).



YAML vs. XML vs. JSON

YAML

```
apis:  
- name: login  
  port: 8080  
- name: profile  
  port: 8090
```

XML

```
<apis>  
  <api>  
    <name>login</name>  
    <port>8080</port>  
  </api>  
  <api>  
    <name>profile</name>  
    <port>8090</port>  
  </api>  
</apis>
```

JSON

```
{  
  "apis": [  
    {  
      "name": "login",  
      "port": 8080  
    },  
    {  
      "name": "profile",  
      "port": 8090  
    }  
  ]  
}
```



Arquiteturas de APIs

- A arquitetura da API geralmente é explicada em termos de cliente e servidor.
- A aplicação que envia a solicitação é chamada de cliente.
- E a aplicação que envia a resposta é chamada de servidor.



Arquiteturas de APIs

- APIs SOAP:

- **Cliente** e **servidor** trocam mensagens usando **XML**.

- Essas **APIs** usam o ***Simple Object Access Protocol*** (Protocolo de Acesso a Objetos Simples).



Arquiteturas de APIs

- APIs RPC:

- O **cliente** conclui uma **função** (ou um **procedimento**) no **servidor** e o **servidor** envia a **saída** de volta ao **cliente**.
- Essas **APIs** são **conhecidas** como ***Remote Procedure Calls*** (Chamadas de Procedimento Remoto).



Arquiteturas de APIs

- **APIs REST:**

- O **cliente** envia **solicitações** ao **servidor** como **dados**.
- O **servidor** usa essa **entrada** do **cliente** para **iniciar funções internas** e **retorna** os **dados** de **saída** ao **cliente**.
- Essas são as **APIs** mais **populares** e **flexíveis** atualmente.
- **REST** significa **Transferência Representacional de Estado**.



Escopo de uso de APIs

- APIs privadas:

- Elas são **internas** a uma **empresa** e são usadas **apenas** para **conectar sistemas** e **dados** dentro da empresa.

- APIs públicas:

- Estas são **abertas** ao **público** e **podem** ser **usadas** por **qualquer** pessoa. Pode ou não haver alguma **autorização** e **custo associado** a esses tipos de **APIs**.



Escopo de uso de APIs

- APIs de parceiros:

- Estas são **acessíveis** apenas por **desenvolvedores externos autorizados** para **auxiliar** as **parcerias** entre **empresas**.

- APIs compostas:

- Estas **combinam** duas ou mais **APIs distintas** para **atender** a **requisitos** ou **comportamentos** complexos do **sistema**.



Vantagens das APIs

- **Integração:**

- As **APIs** são usadas para **integrar** novas **aplicações** com **sistemas de softwares existentes**.

- **Inovação:**

- **Setores** inteiros podem **mudar** com a **chegada** de uma **nova aplicação**.



Vantagens das APIs

- **Expansão:**

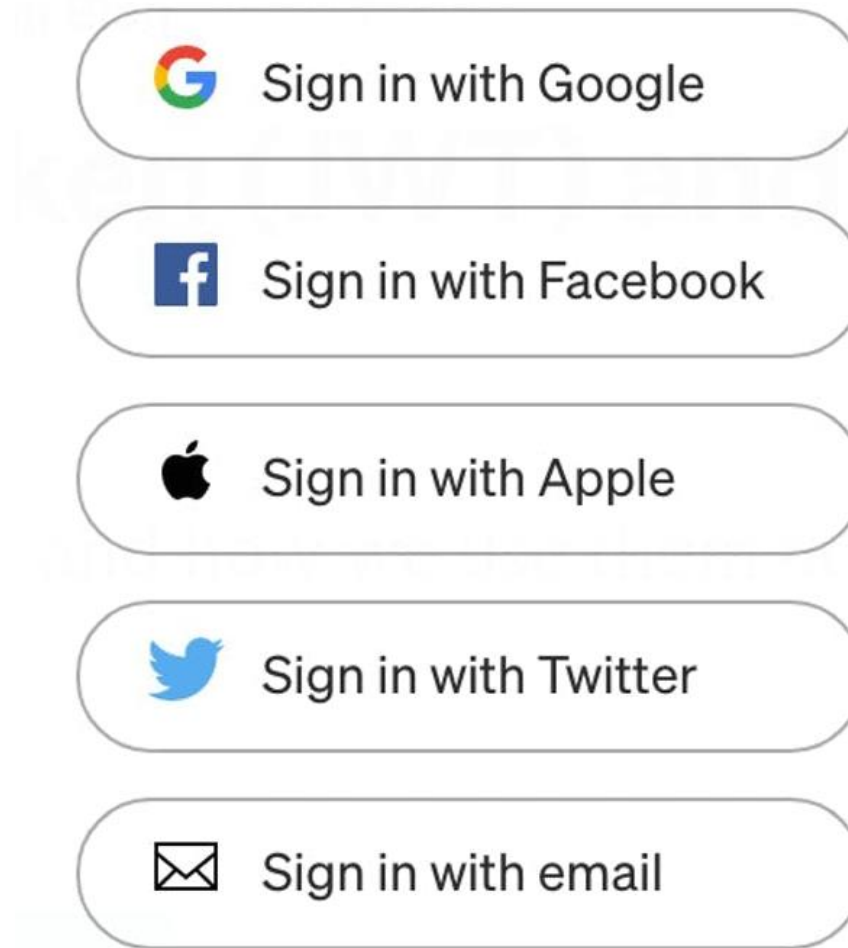
- As **APIs** apresentam uma **oportunidade única** para as **empresas atenderem às necessidades** de seus **clientes em diferentes plataformas.**

- **Facilidade de manutenção:**

- As **APIs** atua como uma **ponte** entre dois **sistemas.**

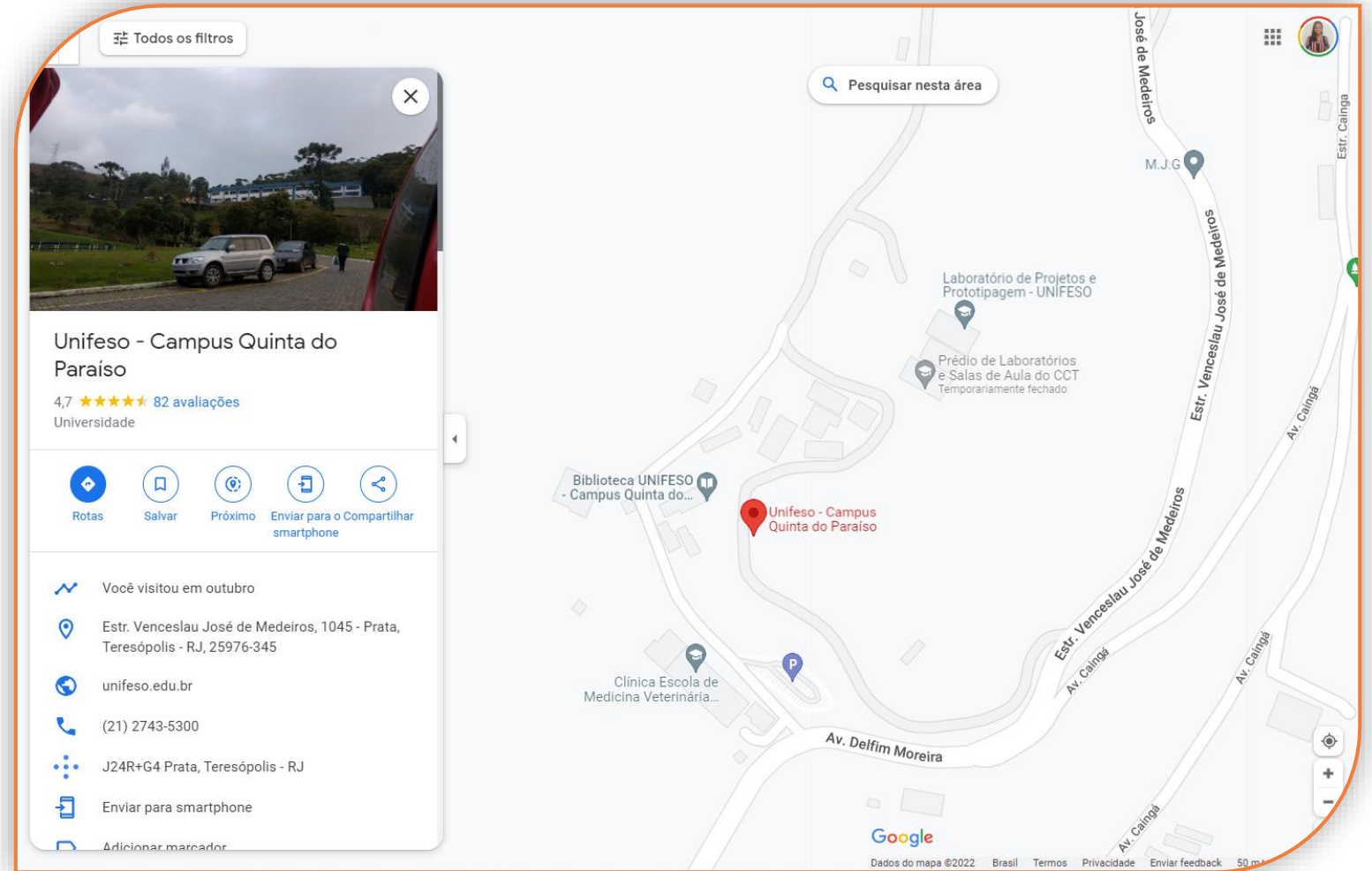


Exemplos de APIs: Redes Sociais



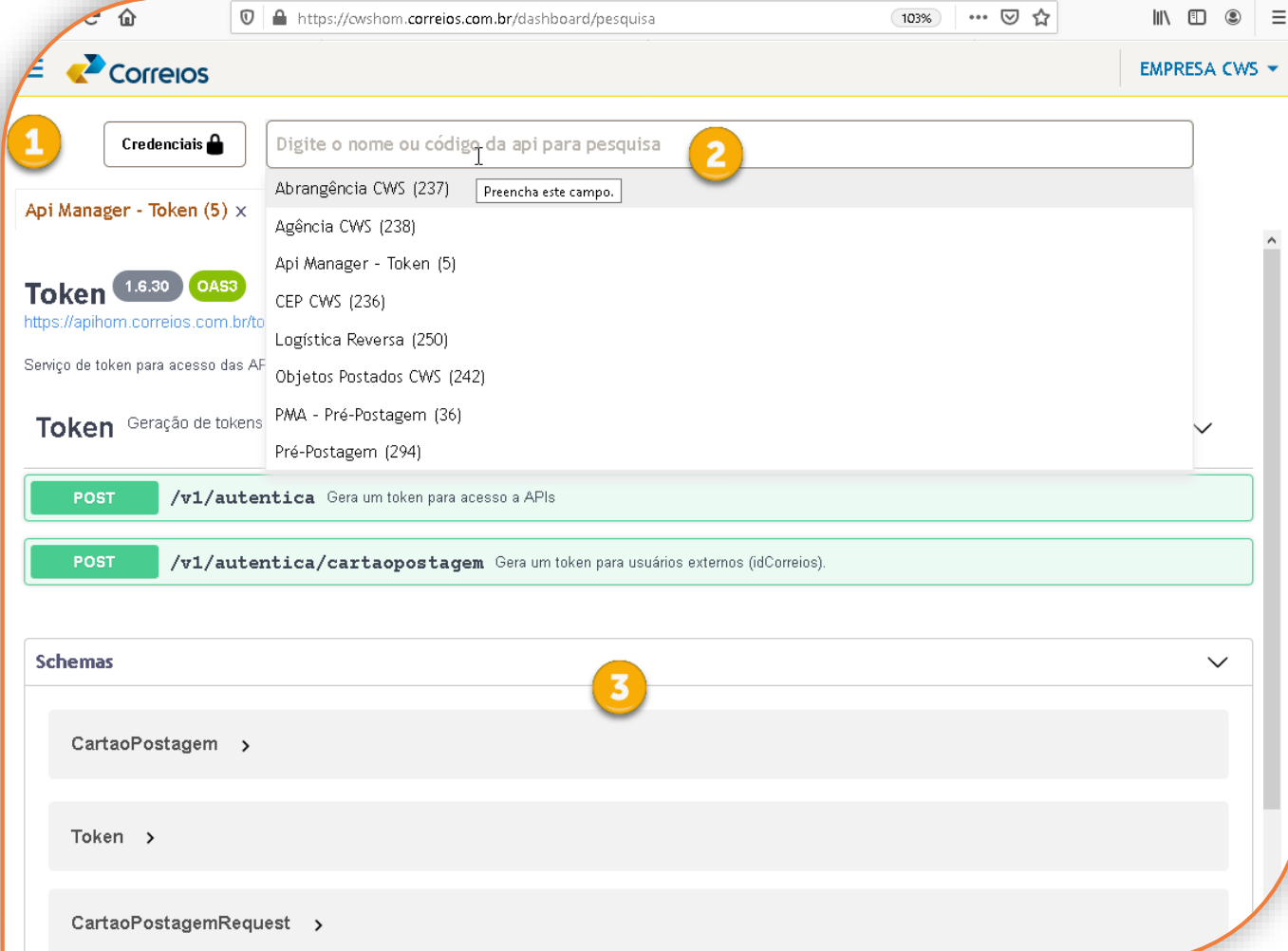
Exemplos de APIs: Mapas

- Bing Maps.
- Here Maps.
- Apple Maps .
- OpenStreetMaps.



Exemplos de APIs: CEP

- ViaCEP.
- Buscar CEP.
- WebmaniaBR.

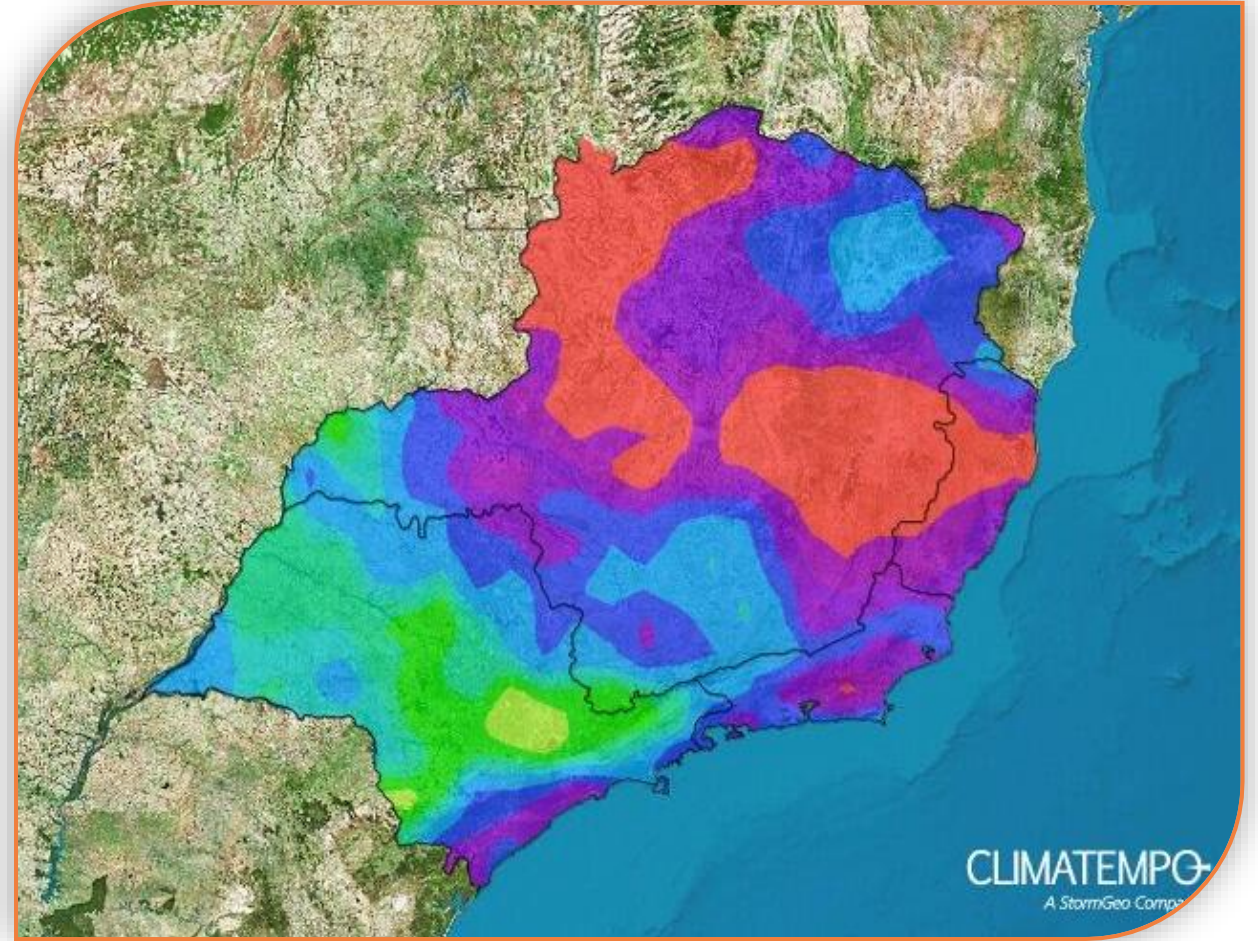


The screenshot shows the 'Correios' API Manager dashboard. At the top, there's a search bar with the placeholder text 'Digite o nome ou código da api para pesquisa'. A dropdown menu is open, displaying a list of APIs including 'Abrangência CWS (237)', 'Agência CWS (238)', 'Api Manager - Token (5)', 'CEP CWS (236)', 'Logística Reversa (250)', 'Objetos Postados CWS (242)', 'PMA - Pré-Postagem (36)', and 'Pré-Postagem (294)'. Below the search bar, there are two API endpoints listed: 'POST /v1/autentica' and 'POST /v1/autentica/cartaopostagem'. At the bottom, there's a 'Schemas' section with three items: 'CartaoPostagem', 'Token', and 'CartaoPostagemRequest'. Three yellow circular callouts with numbers 1, 2, and 3 are overlaid on the image. Callout 1 points to the 'Credenciais' button, callout 2 points to the search input field, and callout 3 points to the 'Schemas' section.



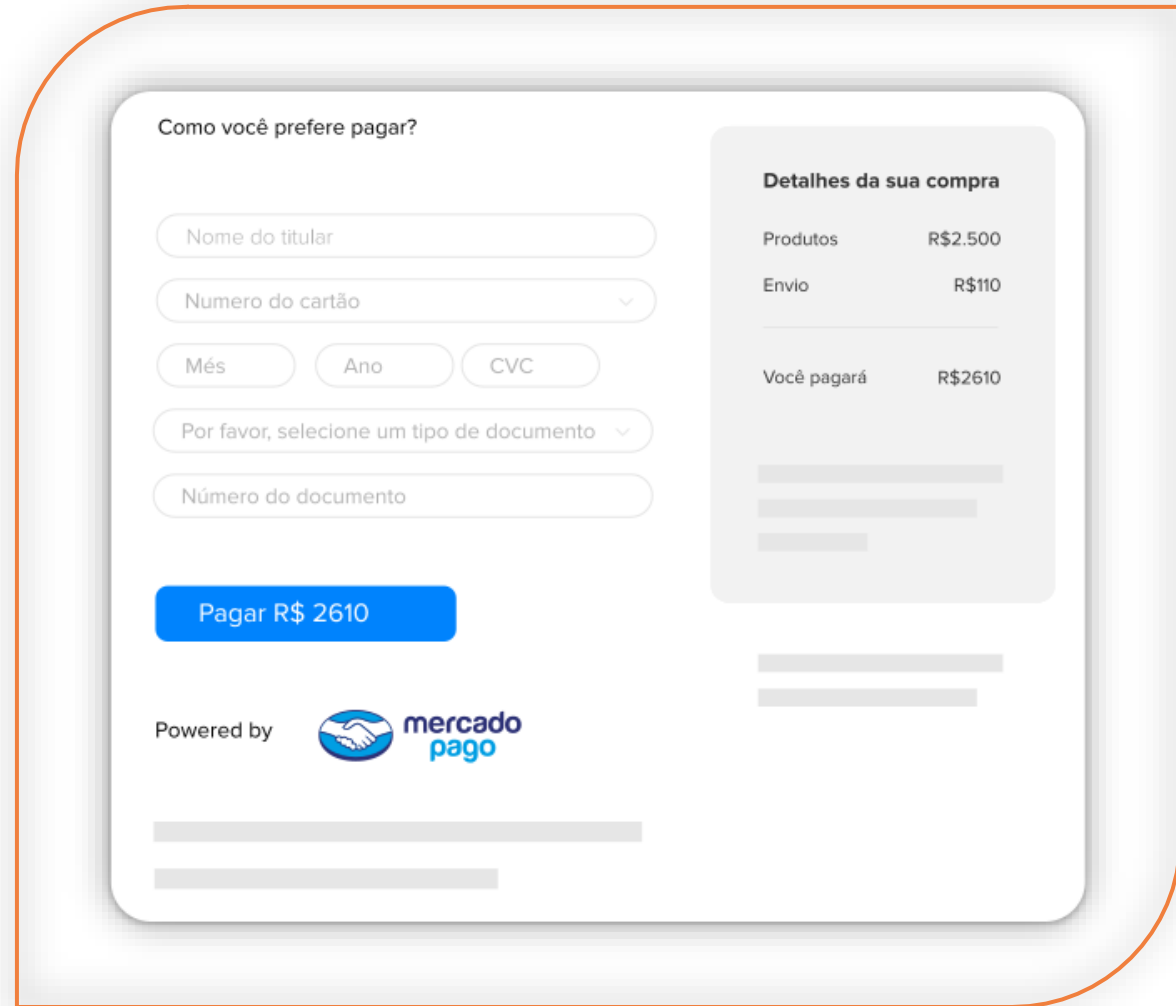
Exemplos de APIs: Clima

- OpenWeatherMap API.
- Climatempo.
- HG Weather.
- CPTEC/INPE.



Exemplos de APIs: Pagamentos

- Mercado Pago.
- PagSeguro.
- Paypal.
- Ebanx.



Como você prefere pagar?

Nome do titular


Numero do cartão

Mês Ano CVC

Por favor, selecione um tipo de documento

Número do documento

Pagar R\$ 2610

Powered by  mercado pago

Detalhes da sua compra

Produtos	R\$2.500
Envio	R\$110
<hr/>	
Você pagará	R\$2610



Métodos de uma Requisição REST

- **GET:**

- Geralmente é **usado** para **solicitar** que um **servidor** **envie** um **recurso**.

- **POST:**

- Foi **projetado** para **enviar dados** de **entrada** para o **servidor**. Na prática, é **frequentemente** usado para suportar **formulários HTML**.



Métodos de uma Requisição REST

- **PUT:**

- **Edita e atualiza documentos em um servidor.**

- **DELETE:**

- **Deleta certo dado ou coleção do servidor.**



Dicas de Ferramentas

- **Python** (Linguagem de Programação).
- **Flask** (Framework API).
- **Insomnia** (Cliente API).



Flask

- É um **Framework** (ou Micro Framework) **Python** para **criar** aplicativos **Web**.
- Aplicativos Web Server Gateway Interface (**WSGI**).
- **Aplicações que utilizam Flask:**
 - Pinterest.
 - LinkedIn.



Insomnia

- Ferramenta para testar e debugar APIs.
- Suporta métodos HTTP como GET, POST, PUT, DELETE.
- Permite visualizar respostas em formatos como JSON e XML.
- Usado por desenvolvedores para testar endpoints de API.



Porque JSON com Python?

- Se você examinar o dicionário Python e o formato JSON, verá que o formato JSON é bastante semelhante ao dicionário Python.
- Os dados são armazenados por pares de chave-valor.



CIÊNCIA DA
COMPUTAÇÃO



Coding Dojo

Até a próxima!

ALEXANDRA RAIBOLT