Olá Dennis,

Tudo bem?

Primeiramente **PARABÉNS!** Você foi selecionado para participar do nosso processo seletivo para a vaga de Desenvolvedor na ADTsys - Campinas/SP.

A primeira etapa do nosso processo seletivo consiste em um teste, segue abaixo o link para a realização do mesmo.

[https://github.com/adtsys-cloud/weather-pokemon-test/tree/master/developer](https://www.google.com/url?q=https://github.com/adtsys-cloud/weather-pokemon-test/tree/master/developer&sa=D&source=hangouts&ust=1559155449737000&usg=AFQjCNGkno4mn4mLjfenyDUrvoHuD9knWA)

**Instruções:**

- Referente ao prazo preferencialmente até**23/03/2020 - Segunda-feira;**

- Enviar o teste para [caroline.santos@adtsys.com.br](mailto:caroline.santos@adtsys.com.br) (respondendo a este e-mail).

**FAVOR CONFIRMAR O RECEBIMENTO DO E-MAIL!!**

Dúvidas favor entrar em contato.

Agradecemos desde já pela participação, e BOA SORTE!!!!

Atenciosamente,

**Desafio Pokémon para Desenvolvedores**

**Definições**

* O desafio a seguir é o dia-a-dia que qualquer desenvolvedor irá enfrentar no time de produtos da ADTSys;
* Leia todo o conteúdo antes de iniciar e busque entender de fato o desafio proposto;
* Utilize as tecnologias de sua preferência: linguagem de programação, frameworks, etc;
* Não se trata de um desafio complexo, apesar da leitura ser extensa;
* O desafio baseia-se em consumir 2 APIs de diferentes fontes de dados com intuito de agregar as informações e solucionar o problema de acordo com o que foi proposto;
* Não completar o desafio não será critério para desclassificação, iremos analisar como a proposta do desafio foi elaborada e não somente o resultado;
* Quaisquer dúvidas sobre o desafio ou dúvidas técnicas fique a vontade para nos perguntar pelo e-mail [team@cloudhit.io](mailto:team@cloudhit.io).

**O Desafio**

Nós do time de produtos das ADTSys, como bons nerds, somos fãs de Pokémon. Durante um bate-papo de descontração, alguém fez a seguinte pergunta: - “E se os pokémons fossem reais, em que lugares de mundo poderíamos encontrar cada tipo?” Então, outra pessoa respondeu: - “Eles se dividiram de acordo com o clima do nosso planeta, por exemplo, em lugares quentes poderíamos encontrar pokémons de fogo.”

O objetivo é criar uma aplicação web seguindo os seguintes critérios:

* Em uma página HTML deve ser possível informar uma cidade de qualquer lugar do mundo;
* De acordo com as condições climáticas desta cidade deve-se exibir um Pokémon baseado em seu tipo (fogo, água, elétrico, etc) seguindo as seguintes regras:
  + Lugares onde a temperatura for menor **(<) que 5ºC**, deve-se retornar um pokémon de **gelo (ice)**.
  + Lugares onde a temperatura estiver entre **(>=) 5ºC e (<) 10ºC**, deve-se retornar um pokémon do tipo **água (water)**.
  + Lugares onde a temperatura estiver entre **(>=) 12ºC e (<) 15ºC**, deve-se retornar um pokémon do tipo **grama (grass)**.
  + Lugares onde a temperatura estiver entre (>=) 15ºC e (<) 21ºC, deve-se retornar um pokémon do tipo **terra (ground)**.
  + Lugares onde a temperatura estiver entre **(>=) 23ºC e (<) 27ºC**, deve-se retornar um pokémon do tipo **inseto (bug)**.
  + Lugares onde a temperatura estiver entre **(>=) 27ºC e 33ºC inclusive**, deve-se retornar um pokémon do tipo **pedra (rock)**.
  + Lugares onde a temperatura for **maior que 33ºC**, deve-se retornar um pokémon do tipo **fogo (fire)**.
  + **Para qualquer outra temperatura**, deve-se retornar um pokémon do tipo **normal**.
  + E, no caso em que **esteja chovendo na região** um pokémon **elétrico (electric)** deve ser retornado, independente da temperatura.
* O pokémon mostrado deve ser aleatório e não deve aparecer duas vezes consecutivas;
* Após a consulta deve-se exibir na tela:
  + Temperatura atual da cidade em graus Celcius;
  + Se está chovendo ou não;
  + Nome do Pokémon seguindo as regras acima.

**Orientações**

* Para consultas de condições climáticas utilize a API [OpenWeatherMap](https://openweathermap.org/api);
* Para consultas de Pokémons utilize a API [PokéAPI](https://pokeapi.co/docs/v2.html).

**Passo 1 - Configurando o OpenweatherMap**

* Acesse o link da plataforma em <https://openweathermap.org/>;
* No topo da página procure e clique no botão de “Sign UP”;
* Entre com as suas credenciais e crie um novo acesso, para que possa gerar um APPID, na plataforma;
* Quando estiver logado, procure e clique no botão “API Keys”;
* Ao ser direcionado para a próxima página visualize um pequeno formulário chamado “Create Key”;
* No input “Name”, coloque o nome que achar mais conveniente, por exemplo: “Default”;
* Em seguida clique no botão “Generate”;
* Ao lado do formulário uma “Key” (chave), será gerada com o nome que você informou no passo anterior, essa chave é o que a plataforma chama de APPID e será utilizada ao realizarmos as requisições Rest para as API(s) da plataforma.

**Passo 2 - API do OpenWeatherMap**

* Estando logado na plataforma procure pelo botão “API” no menu principal da aplicação, ou simplesmente acesse <https://openweathermap.org/api>;
* Na página de API procure pelo título **“Current weather data”**, e clique no botão **“API Doc”**, logo abaixo do título;
* Após o redirecionamento, você terá diversas informações sobre o que a API pode fazer, resumidamente iremos utilizá-la para informar o nome da cidade que deseja ver as condições climáticas da região;
* Para isso deve ser utilizada a API chamada **“By city name”**.
  + Ex.: <https://api.openweathermap.org/data/2.5/weather?q=NOME_DA_CIDADE&appid=APPID>
  + ou <https://api.openweathermap.org/data/2.5/weather?q=Campinas&appid=5445a9ae08df1a3117ae57894fa7cdadah>
  + Substituindo o conteúdo **NOME\_DA\_CIDADE** e **APP\_ID**, com o nome da cidade que deseja saber o clima e o token gerado no passo 1;
* O resultado dessa requisição será um objeto do tipo JSON, atualmente o mais utilizado para aplicações RESTful;
  + Apenas dois atributos são de importância para o desafio o primeiro é o atributo chamado **“weather”**, que irá dar informação caso esteja chovendo na região, e o atributo **“main”**, que contém a temperatura da cidade em questão.

{

"coord": {

"lon": -0.13,

"lat": 51.51

},

"weather": [

{

"id": 800,

"main": "Rain",

"description": "light rain",

"icon": "10n"

}

],

"base": "stations",

"main": {

"temp": 7.5,

"pressure": 1020,

"humidity": 70,

"temp\_min": 5.56,

"temp\_max": 9.44

},

"visibility": 10000,

"wind": {

"speed": 2.1,

"deg": 230

},

"clouds": {

"all": 0

},

"dt": 1557184622,

"sys": {

"type": 1,

"id": 1413,

"message": 0.0157,

"country": "GB",

"sunrise": 1557202928,

"sunset": 1557257521

},

"id": 2643743,

"name": "London",

"cod": 200

}

**Passo 3 - API do PokeAPI**

A plataforma PokéAPI não exige token de autenticação o que torna sua utilização bem mais simples, então acesse o link <https://pokeapi.co/>

* Logo na página inicial é possível testar a API, basta seguir os exemplos clicando nos links abaixo do input de submit, por exemplo **“/type/3/”**;
* A API que nos interessa nesse caso é exatamente a de tipo, porém ao invés de passar um valor numérico iremos passar o nome do tipo como parâmetro, como pode ser visto na documentação: <https://pokeapi.co/docs/v2.html/#types>
* Para isso pode-se fazer a seguinte requisição como no exemplo:
  + <https://pokeapi.co/api/v2/type/><NOME\_DO\_TIPO>
  + ou
  + <https://pokeapi.co/api/v2/type/electric>
  + No qual <NOME\_DO\_TIPO> deve ser substituído pelo tipo de elemento desejado (ice, water, grass, ground, etc)
  + O retorno dessa requisição também será um objeto do tipo JSON, sendo que um único atributo é de interesse para esse desafio, que se trata do array de elementos chamado “pokemon”;

{

"damage\_relations": {...},

"game\_indices": [{...}],

"generation": {...},

"id": 13,

"move\_damage\_class": {...},

"moves": [{...}],

"name": "electric",

"names": [{...}],

"pokemon": [

{

"pokemon": {

"name": "pikachu",

"url": "https://pokeapi.co/api/v2/pokemon/25/"

},

"slot": 1

},

{

"pokemon": {

"name": "raichu",

"url": "https://pokeapi.co/api/v2/pokemon/26/"

},

"slot": 1

},

{

"pokemon": {

"name": "magnemite",

"url": "https://pokeapi.co/api/v2/pokemon/81/"

},

"slot": 1

},

{

"pokemon": {

"name": "magneton",

"url": "https://pokeapi.co/api/v2/pokemon/82/"

},

"slot": 1

},

{...},

]

}

**Informações Adicionais**

* Utilizar o Docker durante o desafio será considerado como ponto adicional; **(não é um requisito obrigatório)**
* Testes unitários também serão considerados como ponto adicional; **(não é um requisito obrigatório)**
* Uma breve documentação com os passos para executar a aplicação deve ser elaborada; **(requisito obrigatório)**
* Na documentação deve constar as tecnologias utilizadas; **(requisito obrigatório)**
* Qualquer coisa adicionada como extra por parte do desenvolvedor, por exemplo exibir a imagem do pokémon na tela, será contabilizada como ponto extra, desde que os outros requisitos estejam 100% funcionais; **(não é um requisito obrigatório)**
* Todo o código deve ser adicionado no em seu Github pessoal, além da documentação no formato markdown facilmente aceito pelo github. **(requisito obrigatório)**

**O que será avaliado**

* Seu domínio com as linguagens utilizadas
* Organização
* Coesão e design de código
* Lógica
* Preocupação com o resultado final
* Facilidade no setup da aplicação
* Documentação
* Tratamentos de erro

**Dicas**

Recomendamos o uso de uma ferramenta chamada postman para realizar o testes com as API(s) (<https://www.getpostman.com/>);

Lembre-se que um dos critérios da aplicação é o uso da temperatura em graus Celsius, caso faça a request de clima no formato original, o resultado dos atributos de temperatura estarão em Fahrenheit, então verifique como alterar para Celsius utilizando o parâmetro units na requisição.

Fique atento as pegadinhas. 8ˆP

Para qualquer desenvolvedor independente de seu conhecimento técnico saber entender a documentação das ferramentas, plataformas, frameworks, etc, é um requisito básico, então sempre dê uma passadinha na área de documentação, mesmo que já tenha utilizado determinada ferramenta, as coisas sempre podem mudar e novas podem surgir.