

# Cloud

## 👤 Nome do Aluno

Breno Schneider Salles de Oliveira

## 📝 Descrição do Projeto

Este projeto é uma **API dockerizada** integrada com um banco de dados **PostgreSQL**, desenvolvida para **gerenciar usuários** com funcionalidades de registro, login e autenticação utilizando **JWT**. Utilizando tecnologias como **Python**, **FastAPI**, **Docker** e **SQLAlchemy**, a aplicação realiza **web scraping** no serviço **OpenWeatherMap** para obter a previsão do clima da cidade de **São Paulo** nos próximos **5 dias**. A API permite que os usuários autenticados consultem informações detalhadas como temperatura prevista, condições climáticas, umidade e velocidade do vento. A integração com o banco de dados assegura o armazenamento seguro das credenciais dos usuários e a gestão eficiente das sessões autenticadas, proporcionando uma experiência segura e confiável para acessar dados meteorológicos atualizados.

## Funcionalidades Implementadas:

- **Scraping de Dados:** Coleta de informações específicas de **[OpenWeatherMap]**.
- **API RESTful:** Fornece endpoints para acessar os dados coletados.
- **Autenticação JWT:** Segurança para proteger os endpoints da API.
- **Containerização com Docker:** Facilita a implantação e execução da aplicação em qualquer ambiente.

## 🔧 Como Executar a Aplicação

### Pré-requisitos

- [Docker](#) instalado na sua máquina.
- [Git](#) instalado.

### Passo a Passo

#### 1. Clone o Repositório:

```
git clone https://github.com/brnoschsaloli/nuvemProjeto.git  
cd nuvemProjeto
```

#### 2. Certifique-se de que o Docker está rodando no seu computador

#### 3. Entre na pasta correta para inicializar a aplicação:

```
cd compose
```

#### 4. Iniciar os Contêineres com Docker Compose:

```
docker compose up -d
```

## 5. Acessar a Aplicação:

- A API estará disponível em <http://localhost:8000/docs>.

# .Documentação dos Endpoints da API

## Autenticação

- **Registrar Usuário**

- **URL:** /register/
- **Método:** POST
- **Body:**

```
{  
    "nome": "seu nome",  
    "email": "seu email",  
    "senha": "sua senha"  
}
```

- **Descrição:** Registra um novo usuário e retorna um token JWT.

- **Login**

- **URL:** /login/
- **Método:** POST
- **Body:**

```
{  
    "email": "seu email",  
    "senha": "sua senha"  
}
```

- **Descrição:** Autentica um usuário e retorna um token JWT.

## Previsão do Tempo

- **Consultar Clima**

- **URL:** /consultar/
- **Método:** GET
- **Headers:**

```
Authorization: Bearer <seu_token_jwt>
```

- **Descrição:** Retorna a previsão do tempo baseada nos dados coletados pelo web scraping.
- **Resposta:**

```
{  
    "previsao": [  
        {  
            "data": "2024-10-15",  
            "temperatura_minima": 16.55,  
            "temperatura_maxima": 26.89,  
            "descricao": "nublado"  
        },  
        {  
            "data": "2024-10-16",  
            "temperatura_minima": 19.94,  
            "temperatura_maxima": 30.03,  
            "descricao": "nublado"  
        }  
        // Mais previsões...  
    ]  
}
```

## 📸 Screenshots dos Endpoints Testados

The screenshot shows the Postman interface with a successful API call. The URL is `localhost:8000/registrar/`. The request method is `POST`. The body is a JSON object with fields: `nome: "bruno"`, `email: "exemplo@email.com"`, and `senha: "senha"`. The response status is `200 OK` with a response time of 16 ms and a body size of 253 B. The response JSON includes a `jwt` token.

```
POST localhost:8000/registrar/  
POST localhost:8000/registrar/  
Params Authorization Headers (10) Body Scripts Tests Settings Cookies Beautify  
none form-data x-www-form-urlencoded raw binary GraphQL JSON  
1 {  
2     "nome": "bruno",  
3     "email": "exemplo@email.com",  
4     "senha": "senha"  
5 }  
Body Cookies Headers (4) Test Results 200 OK • 16 ms • 253 B Save Response ...  
Pretty Raw Preview Visualize JSON  
1 {  
2     "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJleGVtcGxvQGVtYwlsLmNvbSJ9.S0p2q-XP-Eamt_wQzAJTcJVz98wzjejJ36aIxJg4dFw"  
3 }  
Online Console Postbot Runner Capture requests Auto-select agent Cookies Vault Trash
```

Descrição da screenshot: Mostrando a resposta do endpoint `/registrar/`.

The screenshot shows the Postman interface with a successful API call. The request URL is `localhost:8000/login`. The request method is `POST`. The response status is `200 OK` with a response time of `35 ms`, a size of `255 B`, and a token header. The response body is a JSON object containing a JWT token.

```

1 {
2   "email": "exemplo3@email.com",
3   "senha": "senha"
4 }
    
```

```

1 {
2   "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJleGVtcGxvM0B1bWFpbC5jb20ifQ.tSwhnW7wIXKDabkn6nAZBqiiqx1iKFEPoFmMs6DBQXs"
3 }
    
```

Descrição da screenshot: Mostrando a resposta do endpoint `/Login/`.

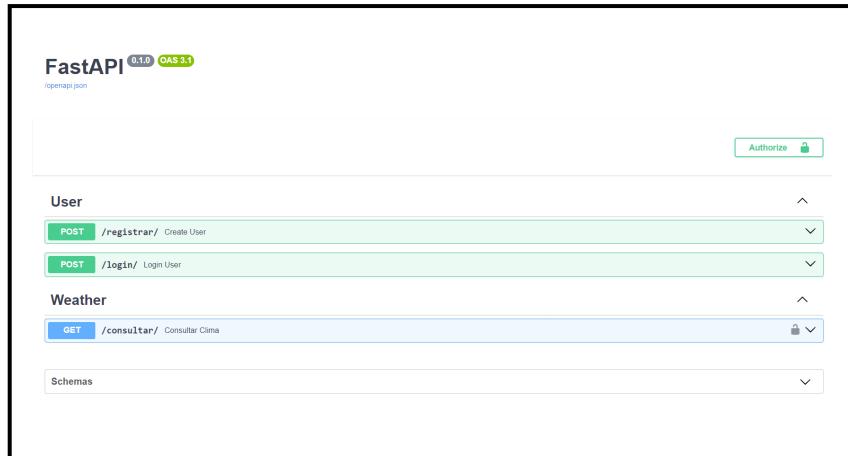
The screenshot shows the Postman interface with a successful API call. The request URL is `localhost:8000/consultar/`. The request method is `GET`. The response status is `200 OK` with a response time of `431 ms`, a size of `737 B`, and a token header. The response body is a JSON array of weather forecasts for specific dates.

```

1 [
2   {
3     "previsao": [
4       {
5         "data": "2024-10-17",
6         "temperatura_minima": 26.34,
7         "temperatura_maxima": 26.34,
8         "descricao": "céu limpo"
9       },
10      {
11        "data": "2024-10-18",
12        "temperatura_minima": 18.11,
13        "temperatura_maxima": 38.56,
14        "descricao": "nublado"
15      },
16      {
17        "data": "2024-10-19",
18        "temperatura_minima": 15.36,
19        "temperatura_maxima": 19.51,
20        "descricao": "chuva leve"
21      },
22      {
23        "data": "2024-10-28",
24        "temperatura_minima": 14.43,
25        "temperatura_maxima": 17.09,
26        "descricao": "chuva leve"
27      },
28      {
29        "data": "2024-10-21",
30        ...
31      }
32    ]
33  }
34 ]
    
```

Descrição da screenshot: Mostrando a resposta do endpoint `/consultar/` com a previsão do tempo.

## 🎥 Vídeo de Execução da Aplicação



Descrição: Vídeo demonstrando a execução da aplicação e a interação com os endpoints da API.

## 📦 Link para o Docker Hub do Projeto

[Docker Hub Repository](#)

Descrição: Link para a imagem Docker do projeto no Docker Hub.

---