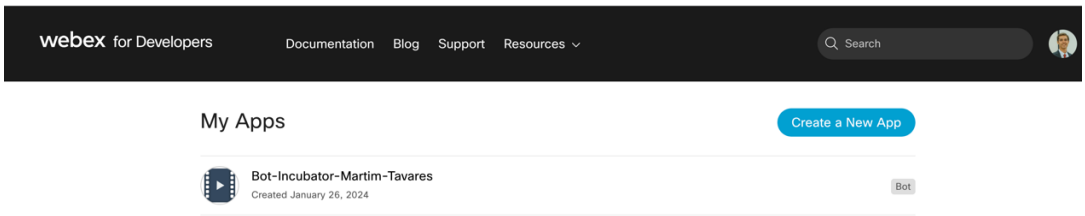
Webex bot that gives the location of IP address

1. The first step I took was the creation of a Webex bot via the Webex site. I created the bot, which user name is incubator-bot-martim-tavares@webex.bot and saved the bot ID, as shown in the image below.



2. After finding the bot on the Webex application, I interacted with it and noticed, as expected, the lack of response from the bot.

3. The next step was to create the bot. My bot uses the API from ipgeolocation.abstractapi.com which gathers information related with an IP address.

4. When the code was being developed, I noticed I wasn't able to make the bot work just by reproducing the steps shown in the DevNet#5 class, probably because my Webex email is not from the cisco domain (i.e. @cisco.com). As such, I included in my git repo the code I was originally going to use (based on the class's code) and the code I ended up using.

    4.1. The code I intended to use is shown bellow. It uses the address.py file to configure the command used by the bot to interact with the API that gathers the information related to any IP address. This API requires a key which is provided in the url. The information I retrieve to the user is the location (city, region and country), the region's current time and the ISP that provided the IP to the user.

    address.py

```python
from webex_bot.models.command import Command
import requests
import json

class Address(Command):
    def __init__(self):
        super().__init__(
            command_keyword="Locate",
            help_message="Get current location and time of an IP address.",
            card=None
        )
    def execute(self, message, attachment_actions, activity):
        IPADDRESS_KEY = "99dba247a0cf4efc9479304cc2d9c696"
        url = "https://ipgeolocation.abstractapi.com/v1/?api_key="
        url += f"{IPADDRESS_KEY}&ip_address="
        url += f"{message}"

        json_string_data = response.content.decode('utf-8')
        response = json.loads(json_string_data)
        address = response['city'] + ", " + response['region'] + ", " + response['country']
        time = response['timezone']['current_time'] + ", " + response['timezone']['abbreviation']
        isp = response['connection']['isp_name']
        response_msg = f"IP address {ip} found!\n ---------------------------------------------- \n"
        response_msg += f"This IP address is located in {address}.\n"
        response_msg += f"In this location, the current time is: {time}.\n"
        response_msg += f"This data is provided by ISP {isp}."
        return response_msg
```

bot.py

```python
from webex_bot.webex_bot import WebexBot
import os
from address import Address
webex_token = os.environ["WEBEX_TOKEN"]
bot = WebexBot(webex_token)
bot.add_command(Address())
bot.run()
```

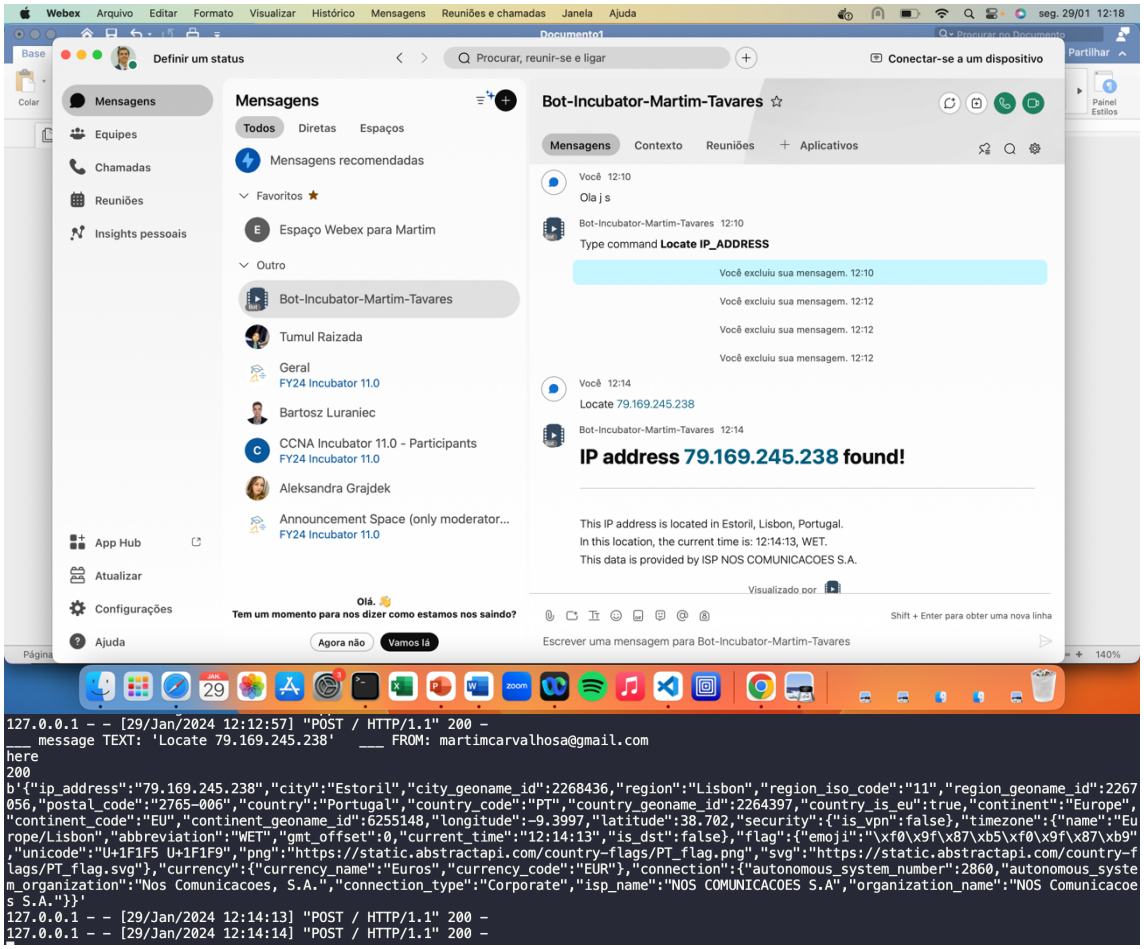
4.2. As the bot didn't receive any messages, I ended up using ngrok where I created a webhook at port 3000 and the files are templated by a git repo, where I only had to indicate the Webex token, the ngrok port and changed the message processing task.

The code in the webex-bot-ngrok.py file demanded a few more configurations than the first option, as I first need to check the string received in the message. Only a message that starts with the word "Locate" and contains a valid number formatted as an IP address could be processed, otherwise the bot would indicate the user how to properly write the request. After that the code interacts with the API just like in the option first presented in 4.1. Another difference with this code is that the response message is sent in a markdown format which makes it more readable.

```python
def process_message(message_obj):
    # Process messages that the bot receives.
    # Access incoming message content with: message_obj.personEmail, message_obj.text, etc. Example API msg at the end of
    import ipaddress
    ip = "0.0.0.0"
    parse_error = False
    print("here")
    for str in message_obj.text.lower().split():
        if message_obj.text.lower().split()[0] == "locate" and str != "locate":
            if ipaddress.ip_address(str):
                ip = str
                parse_error = False
            else:
                parse_error == True
        elif message_obj.text.lower().split()[0] != "locate":
            parse_error = True
    if parse_error:
        msg_result = api.messages.create(toPersonEmail=message_obj.personEmail, markdown="Type command **Locate IP_ADDRES
    else:
        response = requests.get("https://ipgeolocation.abstractapi.com/v1/?api_key=99dba247a0cf4efc9479304cc2d9c696&ip_ad
        print(response.status_code)
        print(response.content)
        json_string_data = response.content.decode('utf-8')
        response = json.loads(json_string_data)
        address = response['city'] + ", " + response['region'] + ", " + response['country']
        time = response['timezone']['current_time'] + ", " + response['timezone']['abbreviation']
        isp = response['connection']['isp_name']
        response_msg = f"IP address {ip} found!\n ---------------------------------------------- \n"
        response_msg += f"This IP address is located in {address}.\n"
        response_msg += f"In this location, the current time is: {time}.\n"
        response_msg += f"This data is provided by ISP {isp}."
        msg_result = api.messages.create(toPersonEmail=message_obj.personEmail, markdown="# "+ response_msg)

    return msg_result
```

5. The end result is shown in the following image which shows two requests (one poorly made and the other correctly made) and the respective responses from the bot.




```
127.0.0.1 - - [29/Jan/2024 12:12:57] "POST / HTTP/1.1" 200 -
___ message TEXT: 'Locate 79.169.245.238'   ___ FROM: martimcarvalhosa@gmail.com
here
200
b'{"ip_address":"79.169.245.238","city":"Estoril","city_geoname_id":2268436,"region":"Lisbon","region_iso_code":"11","region_geoname_id":2267
056,"postal_code":"2765-006","country":"Portugal","country_code":"PT","country_geoname_id":2264397,"country_is_eu":true,"continent":"Europe",
"continent_code":"EU","continent_geoname_id":6255148,"longitude":-9.3997,"latitude":38.702,"security":{"is_vpn":false},"timezone":{"name":"Eu
rope/Lisbon","abbreviation":"WET","gmt_offset":0,"current_time":"12:14:13","is_dst":false},"flag":{"emoji":"\xf0\x9f\x87\xb5\xf0\x9f\x87\xb9"
,"unicode":"U+1F1F5 U+1F1F9","png":"https://static.abstractapi.com/country-flags/PT_flag.png","svg":"https://static.abstractapi.com/country-f
lags/PT_flag.svg"},"currency":{"currency_name":"Euros","currency_code":"EUR"},"connection":{"autonomous_system_number":2860,"autonomous_syste
m_organization":"Nos Comunicacoes, S.A.","connection_type":"Corporate","isp_name":"NOS COMUNICACOES S.A","organization_name":"NOS Comunicacoe
s S.A."}}'
127.0.0.1 - - [29/Jan/2024 12:14:13] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [29/Jan/2024 12:14:14] "POST / HTTP/1.1" 200 -
```