Building WhatsApp bot on Python

Difficulty Level : Expert • Last Updated : 31 Aug, 2021

A WhatsApp bot is application software that is able to carry on communication with humans in a spoken or written manner. And today we are going to learn how we can create a WhatsApp bot using python. First, let's see the requirements for building the WhatsApp bot using python language.

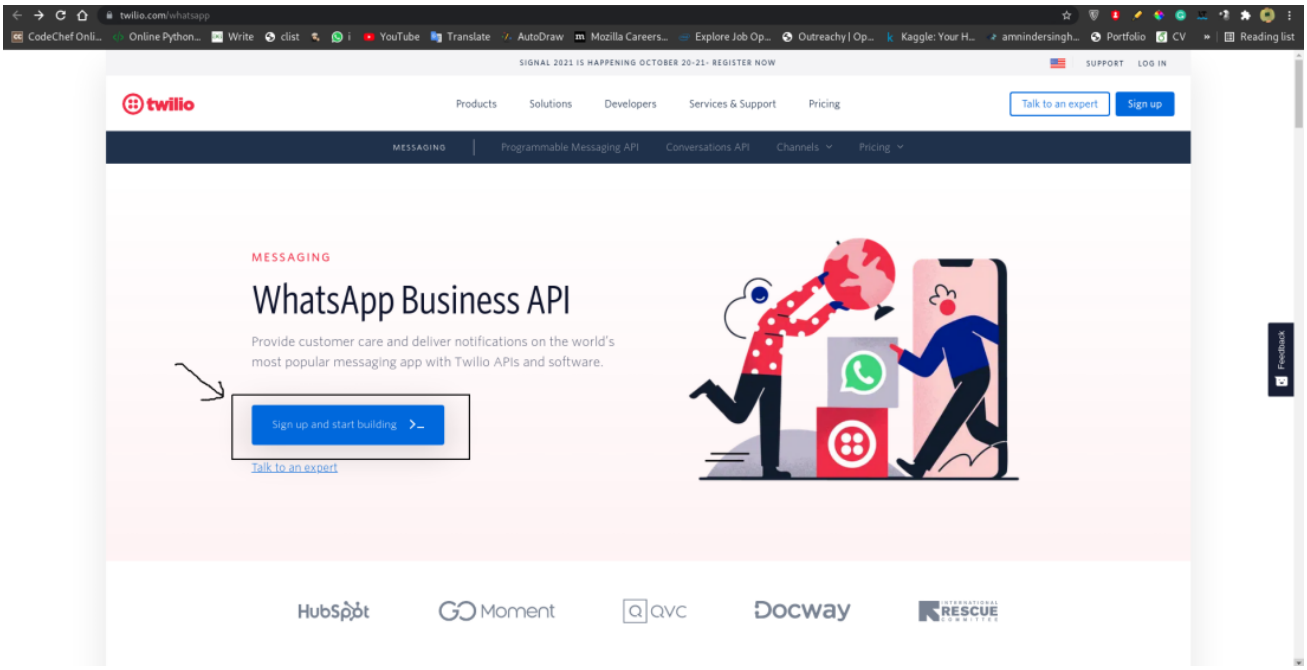
System Requirements:

- A Twilio account and a smartphone with an active phone number and WhatsApp installed.
- Must have **Python 3.9 or newer** installed in the system.
- **Flask:** We will be using a flask to create a web application that responds to incoming WhatsApp messages with it.
- **ngrok:** Ngrok will help us to connect the **Flask application running on your system to a public URL that Twilio can connect to**. This is necessary for the development version of the chatbot because your computer is likely behind a router or firewall, so it isn't directly reachable on the Internet.

Getting Started

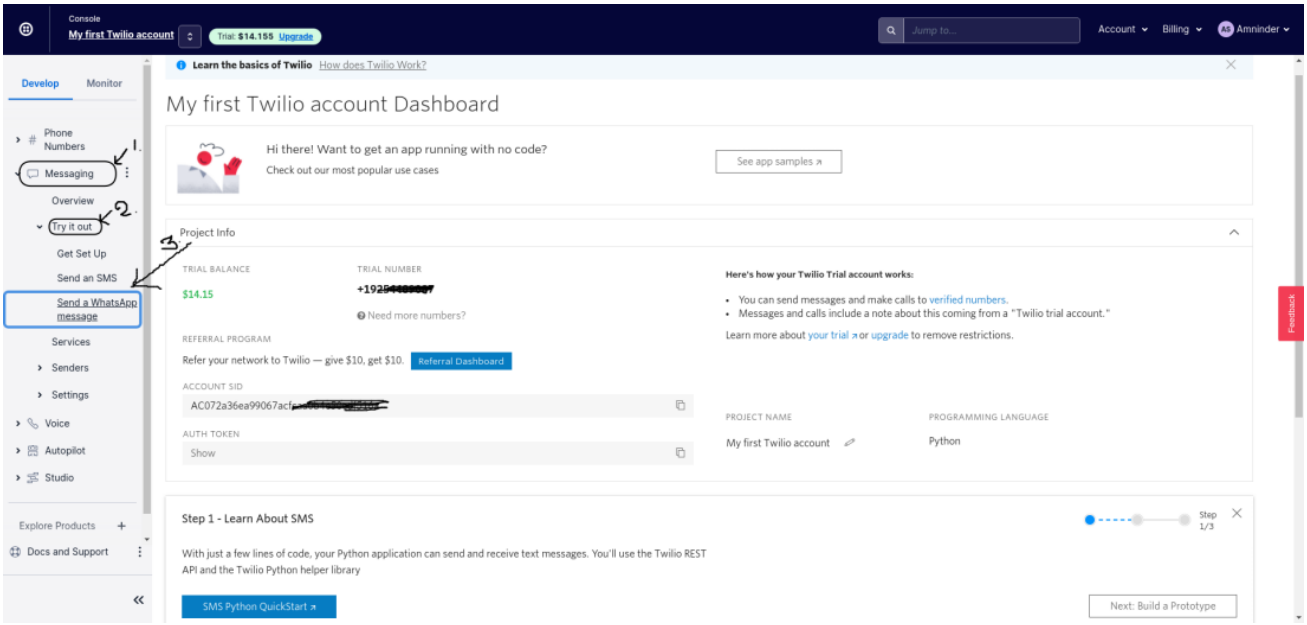
Step 1: Set up the Twilio account using the **Twilio WhatsApp API.**

Go to this link and click on **signup and start building** button and fill in your details and verify your email ID and mobile number.



Sign up

After login, select the **Develop** option from the left menu and then further select the **Messaging** subject then select the **Try it out** option, and in the last click on **Send a WhatsApp message.** This will open up a new webpage for setting up the WhatsApp Sandbox.



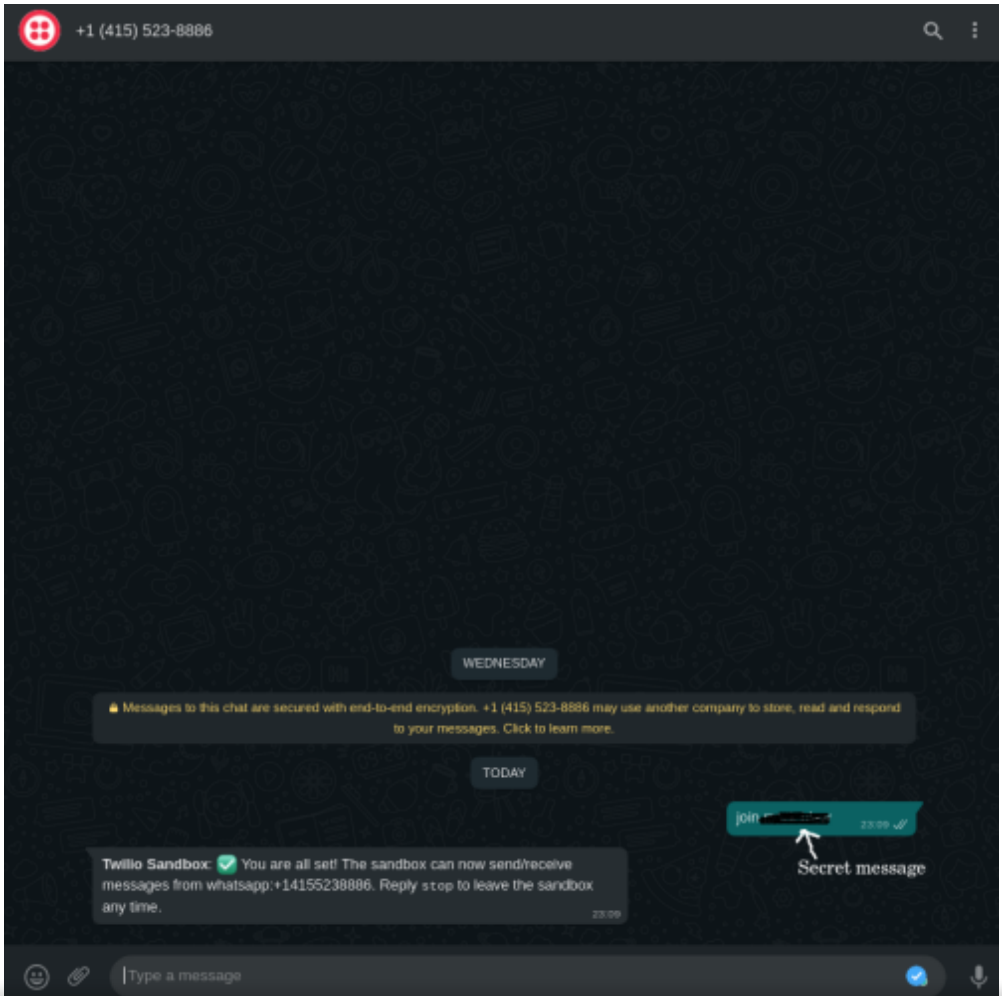
Setup Whatsapp messaging

Step 2: Configure the Twilio WhatsApp Sandbox by sending a message to this WhatsApp number with the secrete unique security code as shown in the below

secret code : *join <secret-code>*

Setup Sandbox

Now, send the secret code to the above WhatsApp message and you will receive a confirmation message as below:



packages.

- **To create the directory and navigate to that directory**:

```
mkdir geeks-bot && cd geeks-bot
```

- **To create and activate the python virtual environment:**

```
python3 -m venv geek-bot-env && source geek-bot-env/bin/activate
```

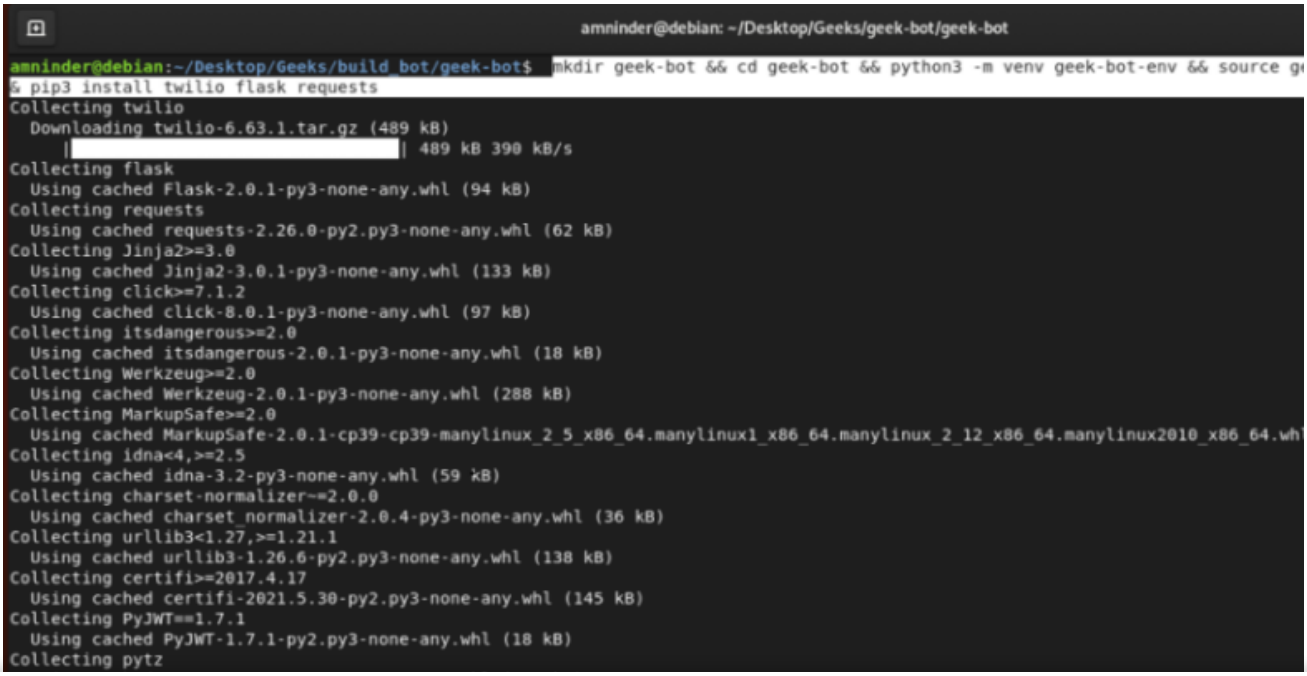
- **To install Twilio, flask and requests:**

```
pip3 install twilio flask requests
```

Here are the above commands in just one line :

mkdir geek-bot && cd geek-bot && python3 -m venv geek-bot-env && source geek-bot-env/bin/activate && pip3 install twilio flask requests

Output:



Creating a Flask Chatbot Service for running the bot locally:

Step 1: Import the necessary files needed to run this flask app.

Python3

```
from flask import Flask, request
import requests
from twilio.twiml.messaging_response import MessagingResponse
```

Step 2: Receiving message entered by the user and sending the response. We can access the user response that is coming in the payload of the POST request with a key of '**Body**'.

Python3

```
from flask import request

incoming_msg = request.values.get('Body', '').lower()
```

Python3

```
from twilio.twiml.messaging_response import MessagingResponse

response = MessagingResponse()
msg = response.message()
msg.body('this is the response/reply  from the bot.)
```

Step 3: So now we will build the chatbot logic, we are going to ask the user to enter a topic that he/she want to learn then we send the message to the bot, and the bot will search the query and respond with the most relevant article from **geeksfogeeks** to

```
# chatbot logic
def bot():

    # user input
    user_msg = request.values.get('Body', '').lower()

    # creating object of MessagingResponse
    response = MessagingResponse()

    # User Query
    q = user_msg + "geeksforgeeks.org"

    # list to store urls
    result = []

    # searching and storing urls
    for i in search(q, tld='co.in', num=6, stop=6, pause=2):
        result.append(i)

    # displaying result
    msg = response.message(f"--- Result for '{user_msg}' are  ---")

    msg = response.message(result[0])
    msg = response.message(result[1])
    msg = response.message(result[2])
    msg = response.message(result[3])
    msg = response.message(result[4])

    return str(response)
```

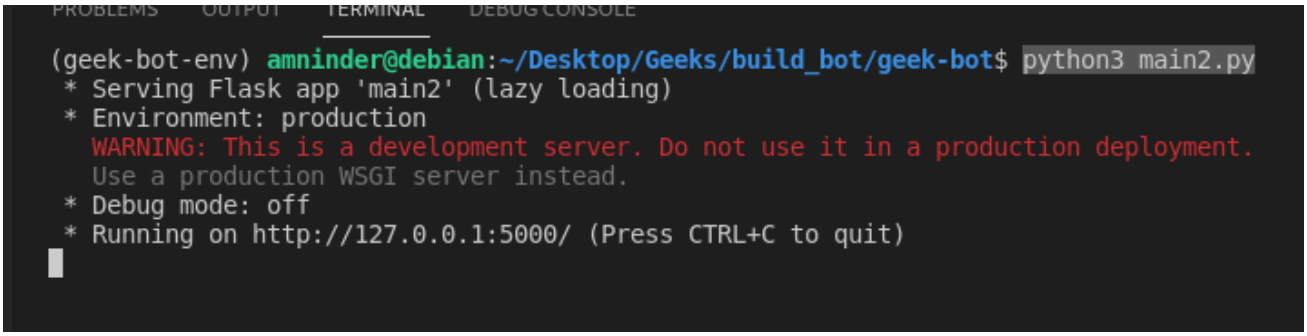
Here, in this function, using **user_msg** will receive the user response/query. Then we are using **google search** to fetch the first 5 links from the Google search with the user query and storing them into a list called **result**. After that, are simply sending the first 5 links to the user using the Twilio messaging service.

To run the bot will follow these steps:

Firstly, run the above script using the following command:

```
python3 main2.py
```

Output:

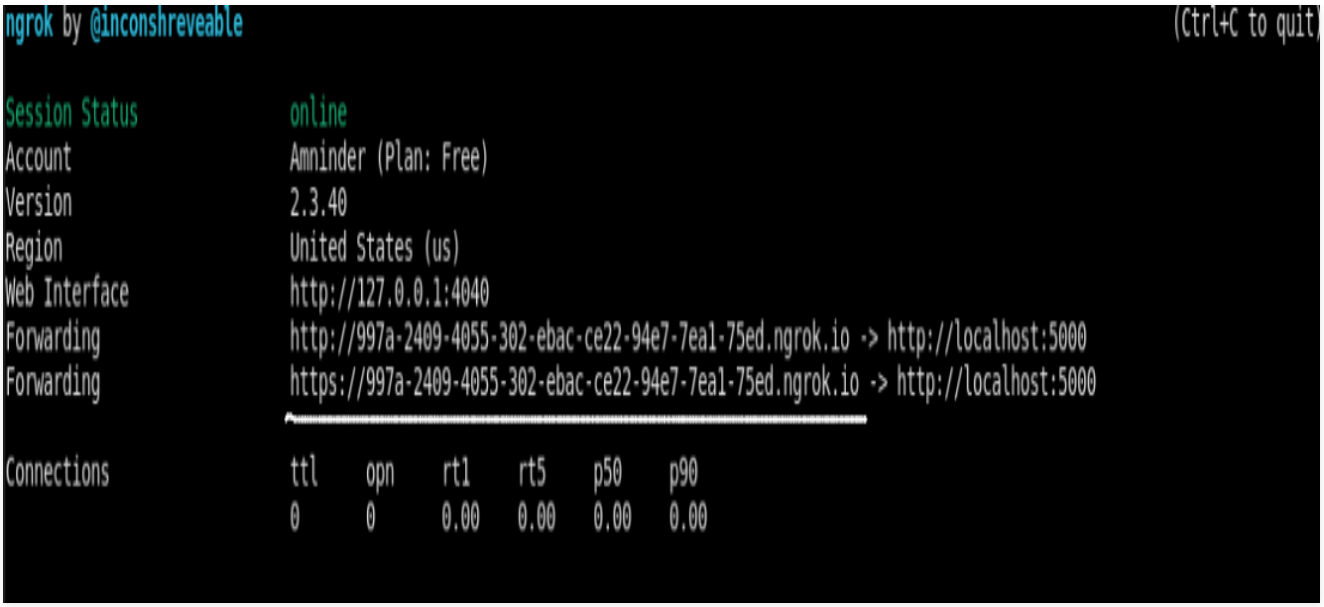


Running the bot script

Secondly, open up another terminal window and run the following command to start the ngrok server.

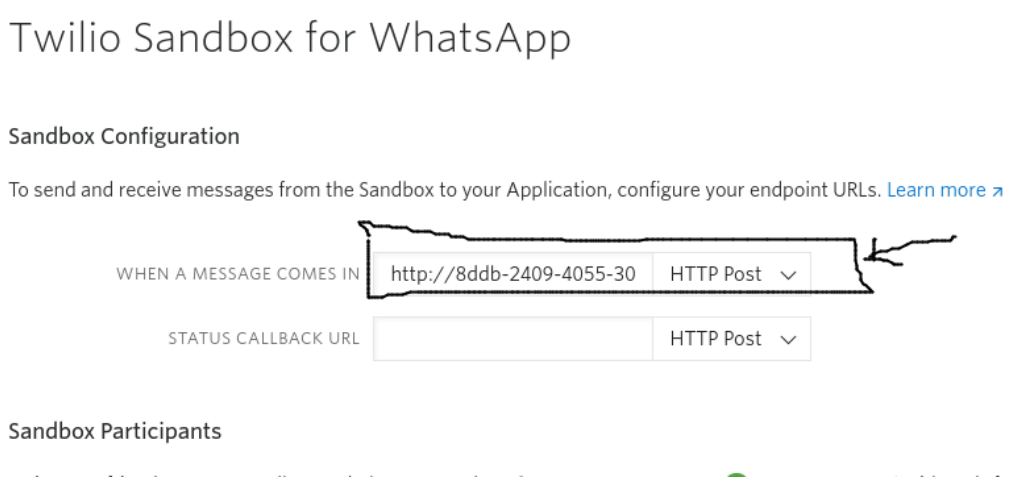
```
ngrok http 5000
```

Output:



And third and last step we have to do is to set up the forwarding URL in the **WhatsApp Sandbox Settings.** Navigate to the following link and paste the forwarding URL in the selected location and click on save.

Link: https://www.twilio.com/console/sms/whatsapp/sandbox



Setup URL in Twilio

Below is the full implementation:

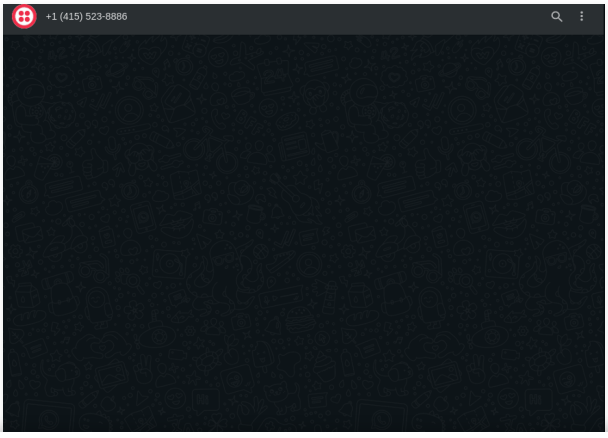
Here, we have imported all the necessary libraries that we're going to use during the execution of the chatbot then we are creating a function called a bot, where we are going to implement our chatbot logic. In the bot function, firstly, we are fetching the response made by the user using WhatsApp and saving it into a variable called user_msg. After that we have created an object of MessagingResponse(), we need that for sending the reply to the user using WhatsApp. We are appending user query with the word "geeksforgeeks.org" because we have made this bot with respect to a user who might have the study-related queries and he/she can ask any doubt related to studies. After that, we have created a list called result where we are going to save the URLs that we have to send to the user. We are using the google search library for googling purposes. Using for loop, we are fetching the first 5 article links and saving them into the result. Using response.message() function we are simply sending the result back to the user through WhatsApp.

Python3

```
from flask import Flask
from googlesearch import search
import requests
from twilio.twiml.messaging_response import MessagingResponse
```

```
def bot():

    # user input
    user_msg = request.values.get('Body', '').lower()

    # creating object of MessagingResponse
    response = MessagingResponse()

    # User Query
    q = user_msg + "geeksforgeeks.org"

    # list to store urls
    result = []

    # searching and storing urls
    for i in search(q, tld='co.in', num=6, stop=6, pause=2):
        result.append(i)

    # displaying result
    msg = response.message(f"--- Result for '{user_msg}' are  ---")

    msg = response.message(result[0])
    msg = response.message(result[1])
    msg = response.message(result[2])
    msg = response.message(result[3])
    msg = response.message(result[4])

    return str(response)


if __name__ == "__main__":
    app.run()
```

Output:



Like 8

Previous Next

RECOMMENDED ARTICLES

Page : 1 2 3 4 5 6

01 **Python | Whatsapp birthday bot**
21, Oct 19

02 **Building a Discord Bot in Python**
16, Mar 22

03 **Google Chrome Dino Bot using Image Recognition | Python**
27, Sep 19

04 **How to make a Twitter Bot in Python?**
02, Jun 20

Vote for difficulty

Current difficulty : Expert

Easy | Normal | Medium | Hard | Expert

Article Tags : Picked, Python-projects, python-utility, Python

Improve Article | Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

News

Top News

Technology

Work & Career

Business

Finance

Lifestyle

Languages

Python

Java

CPP

Golang

C#

SQL

Web Development

Web Tutorials

Django Tutorial

HTML

CSS

JavaScript

Bootstrap

Contribute

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship