<u>Documentation for Job Searching Chatbot Using Facebook's Wit.AI</u>

Submitted by:V Shiva Kumar

This Project was done by me at Curvelogics Advanced Technology Solutions Private Limited, (http://www.curvelogics.com/) as a Intern.Thanks to everyone there for their Continuous Support and Guidance

Overview

This Project is Basically a Chatbot which is Integrated into Facebook Messenger the Bot will Find Jobs Based on the Users Requirements from the popular Job Site Naukri.com(https://www.naukri.com/)

<u>Working</u>

- The Front end of the Bot is Basically a Facebook Messenger while the Back end of the Program is a Python Code which is Connected using a Web hook. All the Back end and front end Activities where done using Facebook's Developer Account
- The Back end Consists of 3 Programs namely main.py, data.py, scrapper.py
- All Communication between the Front end and Back end was done using JSON(JavaScript Object Notation)
- For Development a Tool Called Ngrok(https://ngrok.com/) was used. Ngrok is a limited Service which could only run at 7 hours at a time. The Same product can be deployed Continuously by using Cloud Services like Heroku(https://heroku.com/) and Google App Engine(https://cloud.google.com/appengine/)

<u>Detailed Working</u>

- **<u>The User Communicates through a Facebook Messenger which is Integrated into a Facebook Page(</u>**https://www.facebook.com/Test_page-113722096692640/)
- The Messages sent by the User reaches the Web hook(main.py) in JSON Format.Which is linked to the Respective Page using Facebook's Developer Page. A Sample Json Sent by the Messenger to the Web hook for the Message "Yes" is given Below
  "{'object': 'page', 'entry': [{'id': '113722096692640', 'time': 1571719878674, 'messaging': [{'sender': {'id': '2701441109880031'}, 'recipient': {'id': '113722096692640'}, 'timestamp': 1571719878281, 'message': {'mid': 'bJ0_BrBAqoD4QISM6MBEGvcZksxQjJPrczPjEPXqTTdz-2Os2oYdTdbgE13cKbLv3PoR3TRZBVJI3w8JHxS-uA', 'text': 'Yes'}}]}]}
- Where Sender id is Unique for each user
- From the Above JSON main.py retrieves various Parameters like Sender ID,Recipient id and Text
- Later the Text is passed onto a Chatbot Engine Which is Wit.AI(https://wit.ai/) Wit.AI Contains a Pre trained Model which can identify various Parameters of Job
- After Analysis of the Text Wit.AI returns the
  - Intent: intention or purpose of Statement
  - Entity: A unique Keyword or object in the Statement
  - Value: the respective Word that made Wit identify the Entity
- After the Analysis Wit. Ai returns A JSON with the necessary details from the Analysis
- Based on the Analysis the Web hook Instructs the Bot to Communicate with the User and using the Communication the Bot gets further details to Search for Jobs
- After getting Sufficent Details(Post and Place) the bot Stores the Details in 2 text files which is Unique for each User as it is named after the Sender_id after storing the bot Initiates Another Script Called m.py which takes the value from the Text files and passes it on to a Scrapping Engine which Scrapes naukri for nessecary Data after the Scrapping the data is Stored as a pandas Dataframe

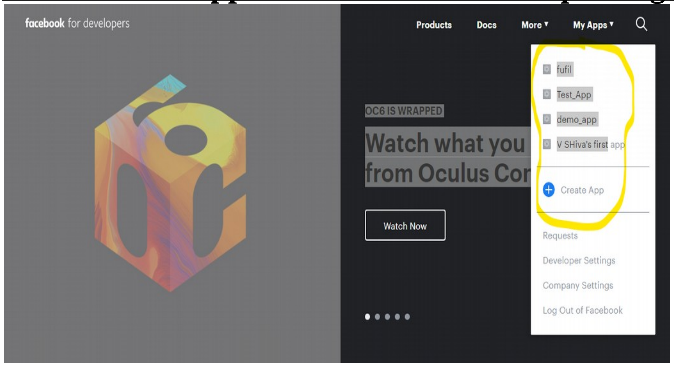
- Later the Dataframe is passed onto a Script called scrapper.py which converts the Dataframe into a String to be passed back to the user
- All Responses from the Bot back to the user is done using a python package called pymessenger
- The Web hook is done using Flask

Procedure

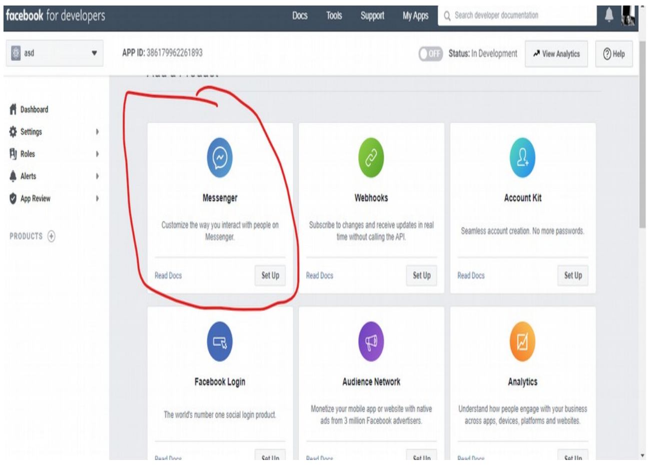
For Developer

Install all necessary packages Mentioned in requirments.txt

1. **Create a new App from Facebook Developer Page(https://developers.facebook.com/)**



2. **It will open a new window Under that in the Messenger Window Select Setup**



Customize the way you interact with people on Messenger.

Subscribe to changes and receive updates in real time without calling the API.

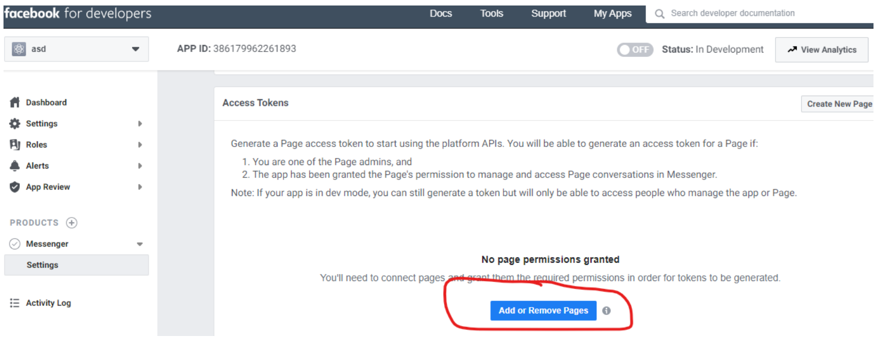
Seamless account creation. No more passwords.

The world's number one social login product.

Monetize your mobile app or website with native ads from 3 million Facebook advertisers.

Understand how people engage with your business across apps, devices, platforms and websites.

3. **A New Window Will Open Under that Click Acess token and select Add or Remove page if you have a Facebook page or create a page using the option on the top right corner**



Access Tokens

Generate a Page access token to start using the platform APIs. You will be able to generate an access token for a Page if:

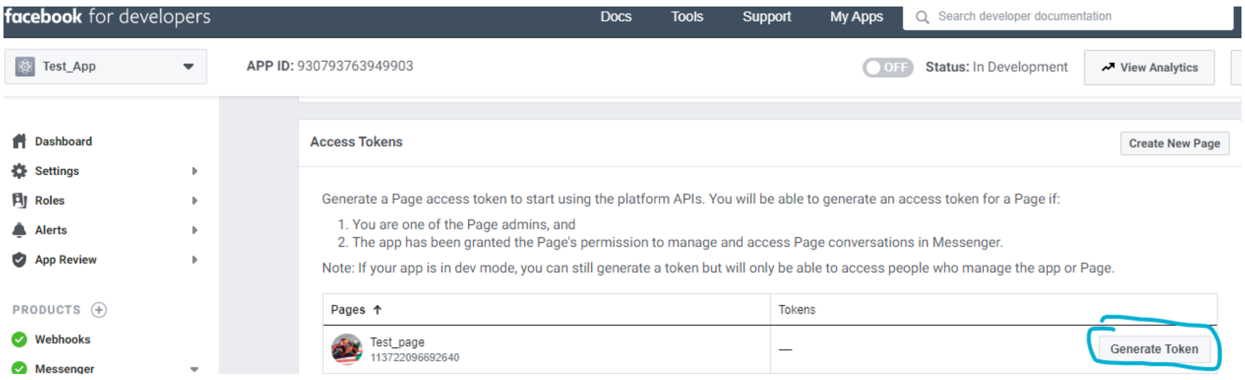
1. You are one of the Page admins, and
2. The app has been granted the Page's permission to manage and access Page conversations in Messenger.

Note: If your app is in dev mode, you can still generate a token but will only be able to access people who manage the app or Page.

No page permissions granted

You'll need to connect pages and grant them the required permissions in order for tokens to be generated.

4. **<u>Once you have Linked your page generate a Acess Token</u>**



facebook for developers — Docs — Tools — Support — My Apps — Search developer documentation

Test_App — APP ID: 930793763949903 — OFF — Status: In Development — View Analytics

Dashboard
Settings
Roles
Alerts
App Review
PRODUCTS
Webhooks
Messenger

Access Tokens — Create New Page

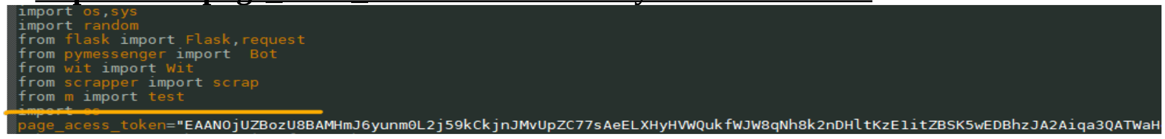
Generate a Page access token to start using the platform APIs. You will be able to generate an access token for a Page if:

1. You are one of the Page admins, and
2. The app has been granted the Page's permission to manage and access Page conversations in Messenger.

Note: If your app is in dev mode, you can still generate a token but will only be able to access people who manage the app or Page.

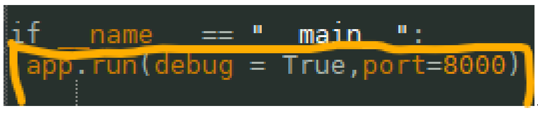
Pages ↑	Tokens
Test_page 113722096692640	— Generate Token

5. **<u>Replace the page_acess_token in Code with your Acess Token</u>**



```
import os,sys
import random
from flask import Flask,request
from pymessenger import  Bot
from wit import Wit
from scrapper import scrap
from m import test
import os
page_acess_token="EAAN0jUZBozU8BAMHmJ6yunm0L2j59kCkjnJMvUpZC77sAeELXHyHVWQukfWJW8qNh8k2nDHltKzE1itZBSK5wEDBhzJA2Aiqa3QATWaH
```

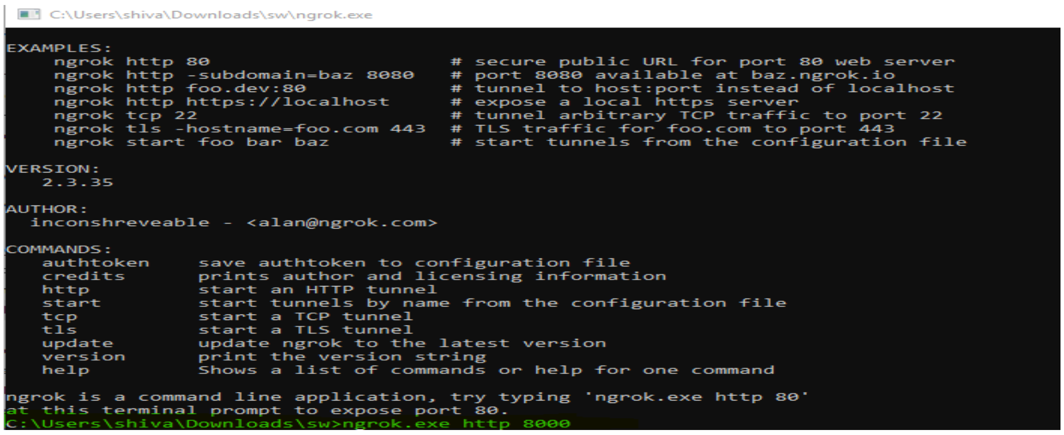
6. **<u>Now run main.py you can specify the Port By default it runs in port 5000</u>**



```
if __name__ == "__main__":
    app.run(debug = True,port=8000)
```

in the picture port is 8000

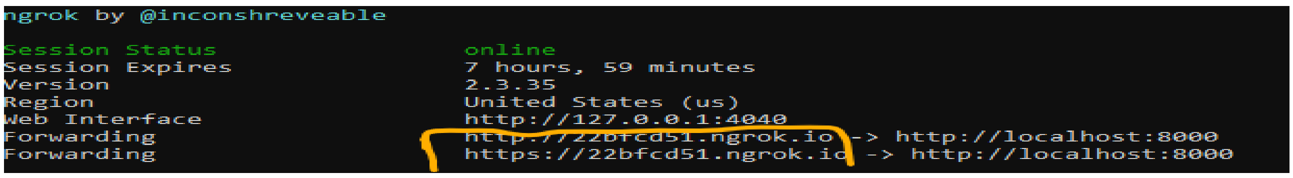
7.now open ngrok and deploy the python using “ngrok.exe http ‘portnumber’”



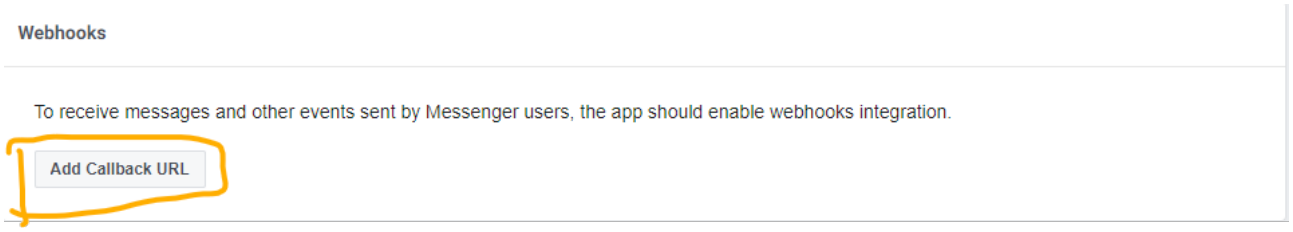
C:\Users\shiva\Downloads\sw\ngrok.exe

```
EXAMPLES:
    ngrok http 80                          # secure public URL for port 80 web server
    ngrok http -subdomain=baz 8080         # port 8080 available at baz.ngrok.io
    ngrok http foo.dev:80                  # tunnel to host:port instead of localhost
    ngrok http https://localhost           # expose a local https server
    ngrok tcp 22                           # tunnel arbitrary TCP traffic to port 22
    ngrok tls -hostname=foo.com 443        # TLS traffic for foo.com to port 443
    ngrok start foo bar baz                # start tunnels from the configuration file

VERSION:
   2.3.35

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
   authtoken     save authtoken to configuration file
   credits       prints author and licensing information
   http          start an HTTP tunnel
   start         start tunnels by name from the configuration file
   tcp           start a TCP tunnel
   tls           start a TLS tunnel
   update        update ngrok to the latest version
   version       print the version string
   help          Shows a list of commands or help for one command

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Users\shiva\Downloads\sw>ngrok.exe http 8000
```

8.Now Ngrok will provide a link copy it and and click Add Callback URL under Webhooks



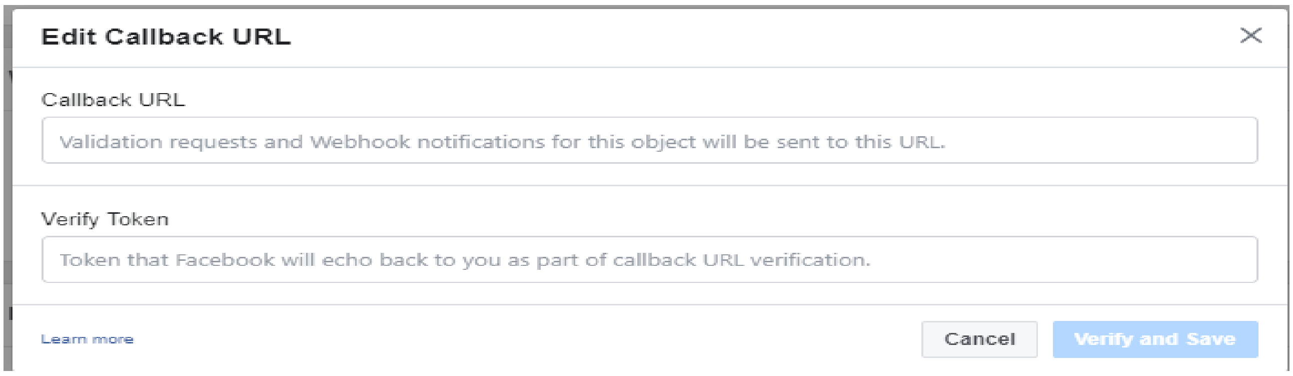
```
ngrok by @inconshreveable

Session Status                online
Session Expires               7 hours, 59 minutes
Version                       2.3.35
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://22bfcd51.ngrok.io -> http://localhost:8000
Forwarding                    https://22bfcd51.ngrok.io -> http://localhost:8000
```



Webhooks

To receive messages and other events sent by Messenger users, the app should enable webhooks integration.

This will open a new window



Edit Callback URL

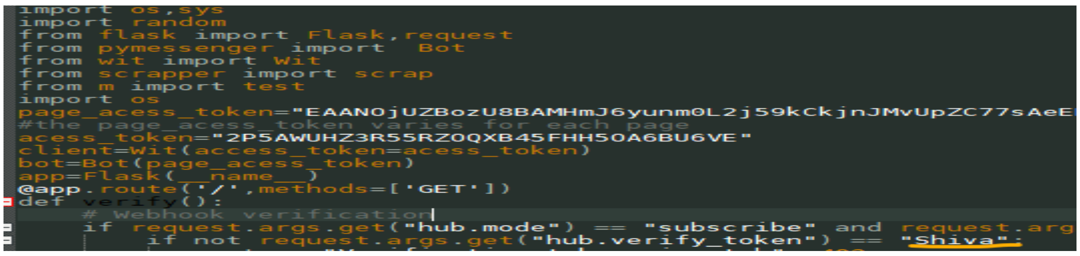
Callback URL

Validation requests and Webhook notifications for this object will be sent to this URL.

Verify Token

Token that Facebook will echo back to you as part of callback URL verification.

In the Callback URL tab paste the ngrok link
and in the verify token paste your Verify token you can change the token by referring the picture below



```
import os,sys
import random
from flask import Flask,request
from pymessenger import  Bot
from wit import Wit
from scrapper import scrap
from m import test
import os
page_acess_token="EAAN0jUZBozU8BAMHmJ6yunm0L2j59kCkjnJMvUpZC77sAeE
#the page_acess_token varies for each page
acess_token="2P5AWUHZ3R55RZ0QXB45FHH5OA6BU6VE"
client=Wit(access_token=acess_token)
bot=Bot(page_acess_token)
app=Flask(__name__)
@app.route('/',methods=['GET'])
def verify():
    # Webhook verification
    if request.args.get("hub.mode") == "subscribe" and request.arg
        if not request.args.get("hub.verify_token") == "Shiva":
```

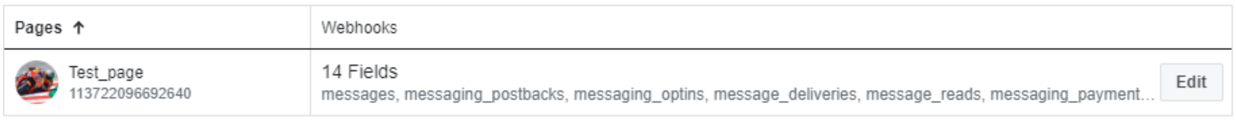
Here Shiva is the Verify token
 thats it you are good to go now start sending the messages and let your bot do the trick

<u>For User</u>

- **<u>Head to the messenger of the Page you linked and start chatting</u>**
- **<u>the bot will respond Accordingly</u>**
- **<u>The bot will only return 5 jobs</u>**

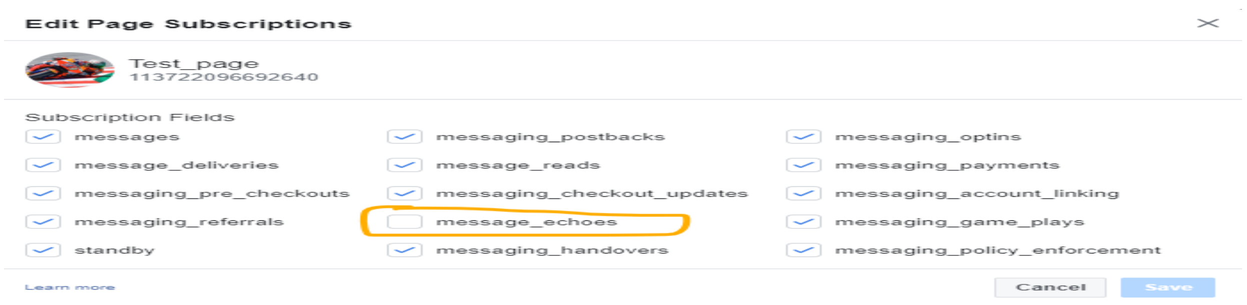
<u>Issues Faced</u>

- **Wit.Ai stopped being a Conversational bot and now only Supports NLP Hence all Conversations where created using if and else**
- **For some reason Wit.ai could only recognise Words which it already has hence nay jobs which it is not in its list wont be recognised**
- **Facebook Messenger has a feature called Echoing where the bots reply was considered as the next input from user this caused trouble in Wit. Hence Manually turning off echoing is recommended(Once you provide the callback URL a new tab will come in which you have to Specify the Various Aspects the Web-hook can Access**



Pages ↑	Webhooks	
Test_page 113722096692640	14 Fields messages, messaging_postbacks, messaging_optins, message_deliveries, message_reads, messaging_payment...	Edit

<u>When you click edit another Window will pop in that select the necessary criteria and Also remember not to select Echo</u>



Edit Page Subscriptions

Test_page
113722096692640

Subscription Fields

✓ messages	✓ messaging_postbacks	✓ messaging_optins
✓ message_deliveries	✓ message_reads	✓ messaging_payments
✓ messaging_pre_checkouts	✓ messaging_checkout_updates	✓ messaging_account_linking
✓ messaging_referrals	☐ message_echoes	✓ messaging_game_plays
✓ standby	✓ messaging_handovers	✓ messaging_policy_enforcement

- **Messenger Provides a option to run wit.ai internally without the need of manually passing but we found out that the JSON used there is Complex and Also Inconsistent And there wasn't any notable Improvement in Time or Performance**
- **Slow network Connection in May Result in the Bot taking time to reply**
- **Also messenger only passes on the next message when a 200Ok is received for previous message when it dosent receive 200OK (waiting period is 20 secs)it will cache the messages hence slowing down the system**