Your Book – Quantified Self App

PROJECT-II REPORT

Submitted by

Pushpak Ruhil | 2019-310-128

in partial fulfillment for the award of the degree of

B.Tech CSE

Under the supervision of

Dr. Shah Imran Alam

Department of Computer Science& Engineering School of Engineering Sciences & Technology

JAMIA HAMDARD

New Delhi-110062

(2022)

DECLARATION

I, **Mr. Pushpak Ruhil** a student of **B.Tech CSE (Enrolment No: 2019-310-128)** hereby declare that the Project/Dissertation entitled**"……..”** which is being submitted by me to the Department of Computer Science, Jamia Hamdard, New Delhi in partial fulfillment of the requirement for the award of the degree of **B.Tech CSE,** is my original work and has been done by me under the general supervision of my supervisor, Mr. Shah Imran Alam.

(Signature and Name of the Applicant)

Date:

Place: Jamia Hamdard University

ACKNOWLEDGEMENT

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this report. Special thanks are due to my supervisor Mr. Shah Imran Alam whose help, stimulating suggestions and encouragement helped me in all time of fabrication process and in writing this report.
I also sincerely thanks for the time spent proofreading and correcting my many mistakes.

Many thanks go to the all lecturer and supervisors who have given their full effort in guiding the team in achieving the goal as well as their encouragement to maintain our progress in track.
My profound thanks go to all classmates, especially to my friends for spending their time in helping and giving support whenever I need it in fabricating my project.

_____

Pushpak Ruhil Date:

INTRODUCTION

With the world shifting towards Artificial Intelligence and Machine learning, we are using many technologies in our day-to-day life to ease our work.
It is also important to understand the need of creating a real life environment when, say, talking to an Artificial Intelligent agent.

Easy Life Bot is a chatbot.

Chatbots are essentially artificial intelligent software application used to conduct a chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent.
You send a text to the chatbot and it automatically responds back based on its learning with the data.

PROBLEM STATEMENT

- To develop an artificial intelligent chatbot which can be used for chatting, small talks, or performing certain key tasks like searching terms for you, telling you the weather or even converting currencies.

- The chatbot should be able to run seamlessly on any device.

- The chatbot should understand basic questions and answer accordingly.

- Chatbot should be easy for users to understand and use.

SOFTWARE REQUIREMENTS SPECIFICATIONS

PURPOSE:

The purpose of this AI enabled chatbot is to provide and assist the users with certain daily tasks that can be handled from one platform at once. It helps users to perform certain search actions, check for weather of any location, convert currencies etc.
All things happening at once place, along with being able to have an interesting small talk with the same chatbot is indeed something to make the life easy.

PROJECT SCOPE:

This project is accessible through Telegram platform, which is available on various different devices like Android, iOS, windows, macOS, etc.
Telegram also works on the web, hence, you can access the chatbot on any platform.
The chatbot has been designed for best user experience with a clean user interface which is easy to use and understand for anyone.

OVERALL DESCRIPTION:

Easy Life Bot is a chatbot developed using Python programming language and Google's DialogFlow framework and can be used to perform certain search actions as well as check for weather, convert currencies, or for small talks.

Telegram is a platform which is used widely across the globe, just like WhatsApp. The chatbot has been deployed on Telegram.

This chatbot brings all major functionalities to one single platform for you, to easy the work.

PRODUCT PERSPECTIVE:

The proposed application is **Easy Life Bot** which is an AI enabled chatbot that has been developed using Python(Flask), for the backend & logic, and Google's DialogFlow, which is a framework which helps in creating chatbot agents. DialogFlow is a NLP platform, which helps in creating agents with the capabilities of natural language understanding.

PRODUCT FEATURES:

The backend is implemented with the help of Python's Flask framework.

The chatbot comes with a number of features like –

- Having NLP and NLU enabled small talks.
- Convert any available currency into another currency.
- Search for anything on the web and get the search results instantly.
- Functionality to check the weather.
- Deployed on telegram, which makes it available to use for anyone, given, the server is on.
- Funny and annoying emotions given for small talks to make the conversation funny and interesting.

HARDWARE INTERFACE:

For the users to use the chatbot, the hardware requirements are:

- A decent internet connection.
- A device which supports Telegram.

USER INTERFACE:

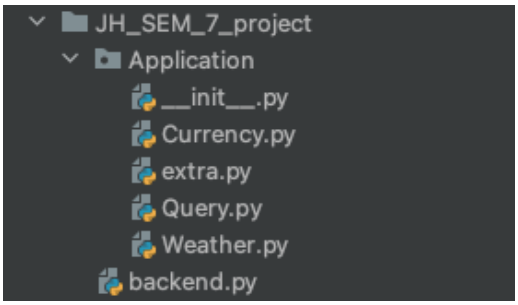
The user interface is the same as that of Telegram.
You will get a clean chatting experience. The replies given by the chatbot agent are optimized to give clean formatted results.

DESIGN AND IMPLEMENTATION CONSTRAINTS:

Due to the chatbot being integrated with telegram, there are minimal implementation constraints. None to be concerned about.

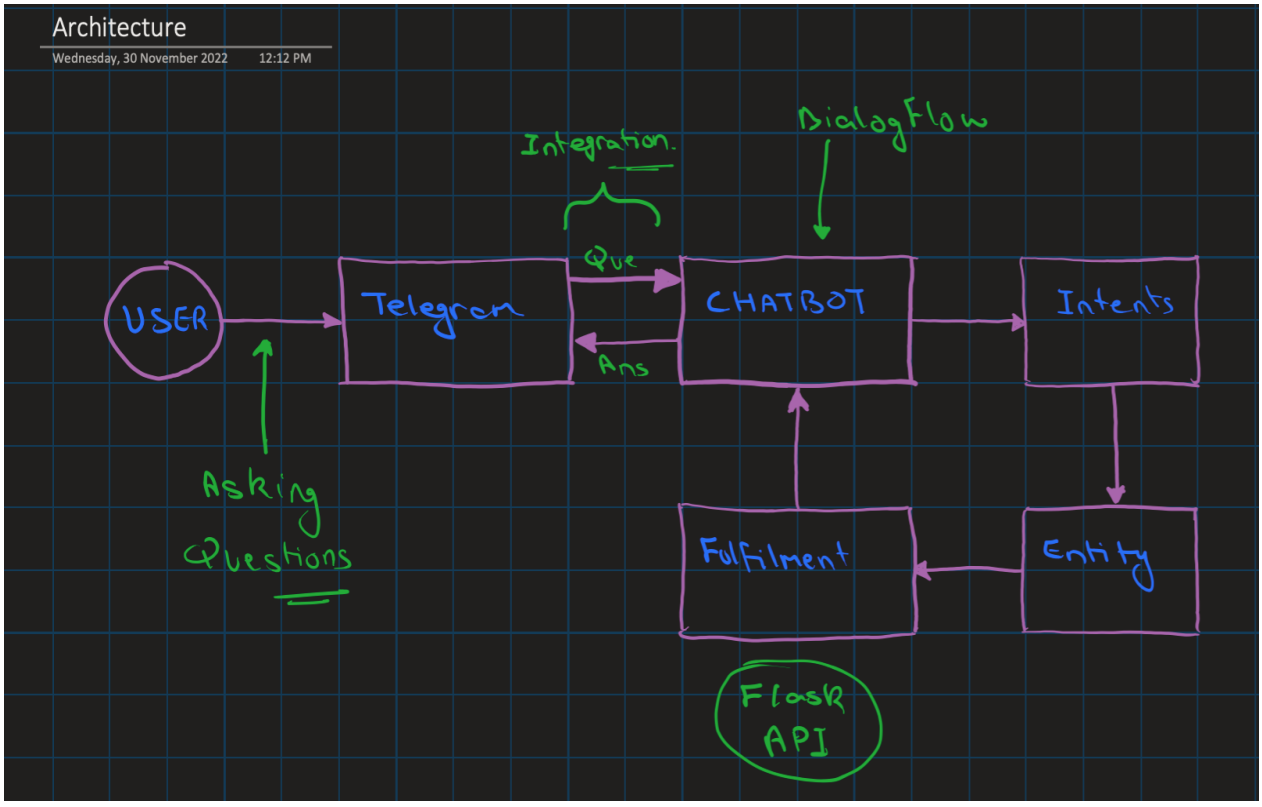
Recommended device is any device which can run telegram. Android, macOS, windows, etc.

ARCHITECTURE



backend.py file contains the main Flask code; Application folder is the module that I have created.
File names are pretty obvious for their roles. Each file contains the code logic for their respective task.

Flow Diagram to understand the working architecture of the project



TECHNOLOGIES USED

Here's are the technologies I used in this project.

1) Python – Flask, Request, Jsonify, BeautifulSoup
2) Ngrok
3) Google's DialogFlow framework
4) API for converting currency.
5) Webhooks

- Python is the core programming language used
- Flask is the main framework used for the backend jobs.
- Request library for handling to HTTP requests
- Jsonify to return the JSON object as a response
- BeautifulSoup for scrapping data from the web
- Ngrok to create a localhost tunnelling. Exposing localhost port to the web, temporarily (to make the webhooks work)
- Dialogflow to create our chatbot agent and train various intents in it.
- APIs are used to get the current currency exchange rate for converting currencies.
- Webhook is the most crucial part for the project to connect the chatbot agent to the backend, thereby, enabling us to handle all the HTTP requests.

SETUP TO RUN THE APP

Initial requirements to start the backend server for the chatbot to perform its actions-

- Python on the system
- A working internet connection

1 – Install the necessary required modules/packages from the requirements.txt in the project folder. This will install all required python packages

```
pip install -r requirements.txt
```

2 – Install Ngrok from the web. This is required for the localhost tunnelling to expose your port to the web to make the webhooks work.

3 – Start the flask server in python using the following command.

```
python3 backend.py
```

4 – Start the ngrok using the following command. Assuming that the flask server is running on the port 8000

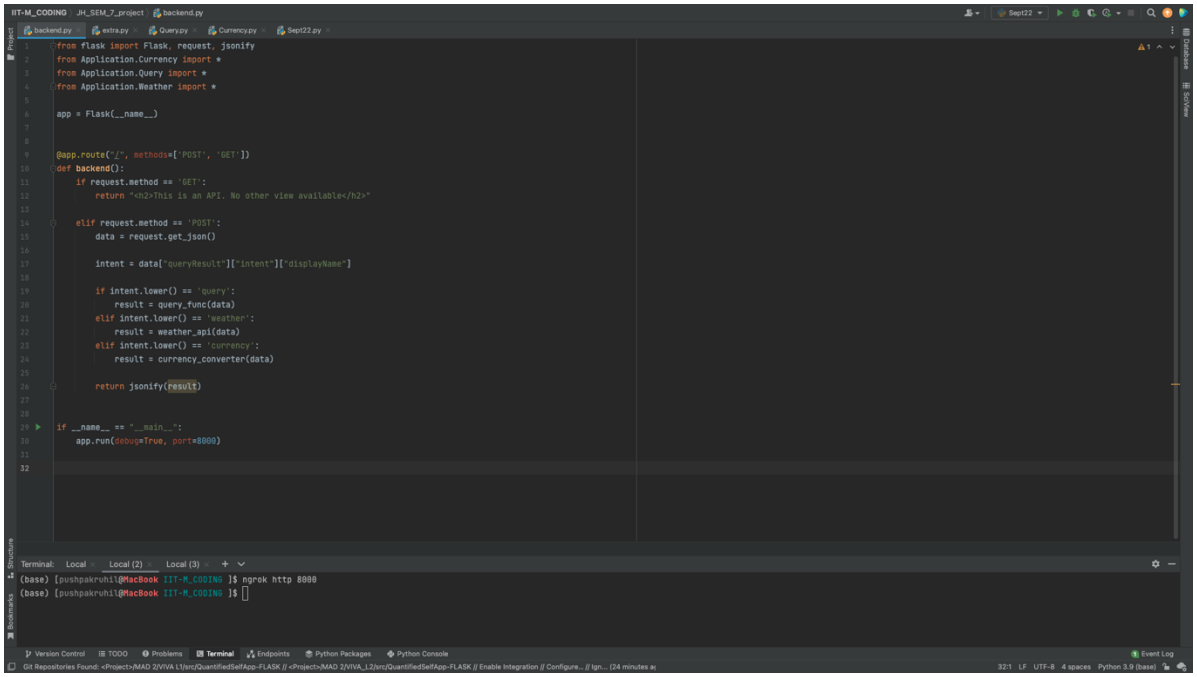
```
ngrok http 8000
```

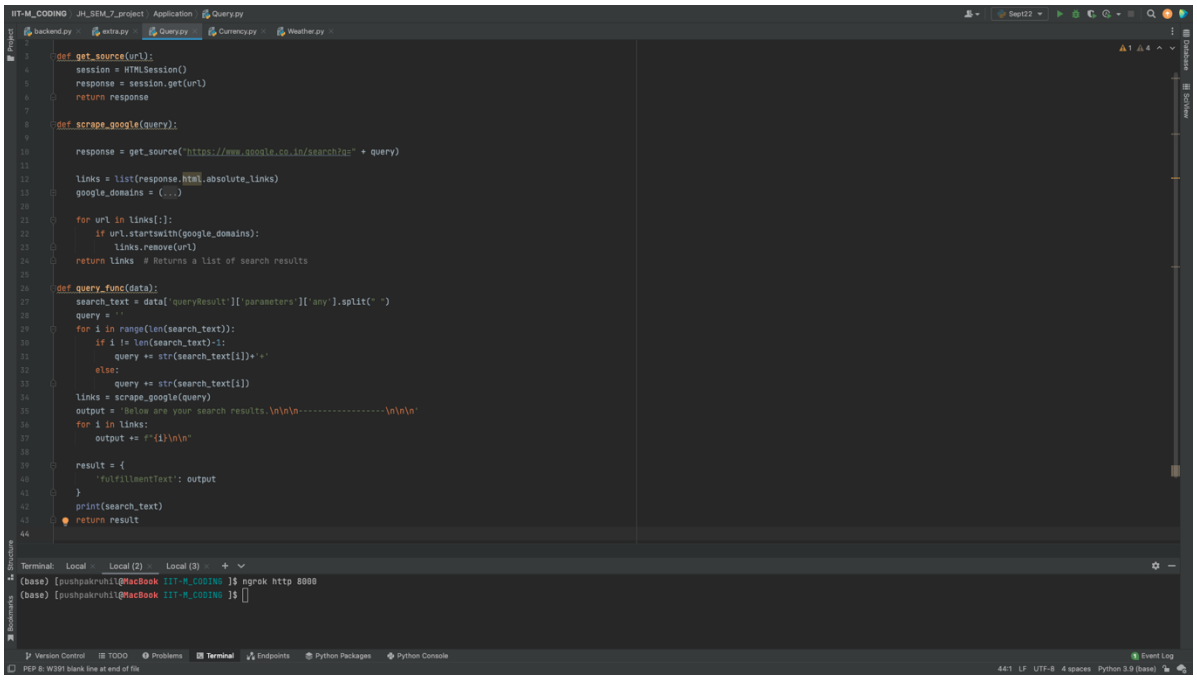
5 – Now, go to Telegram app and search for the username @PushpakRuhilBot

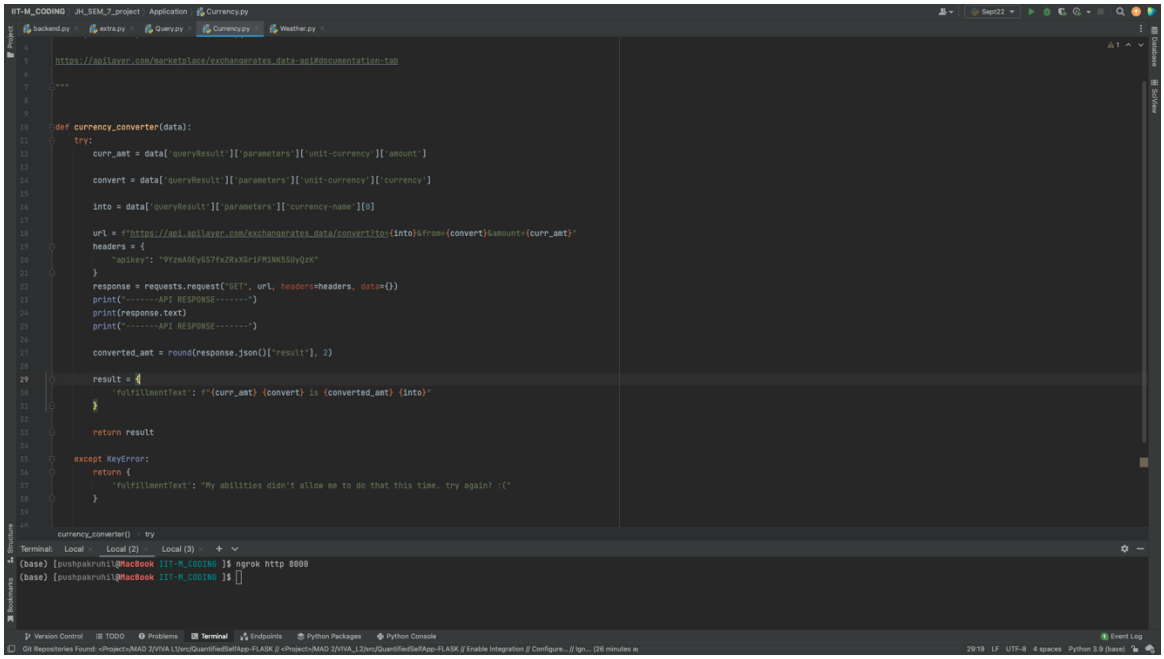
6 – You are now good to go and chat with the chatbot. All setup has been done.

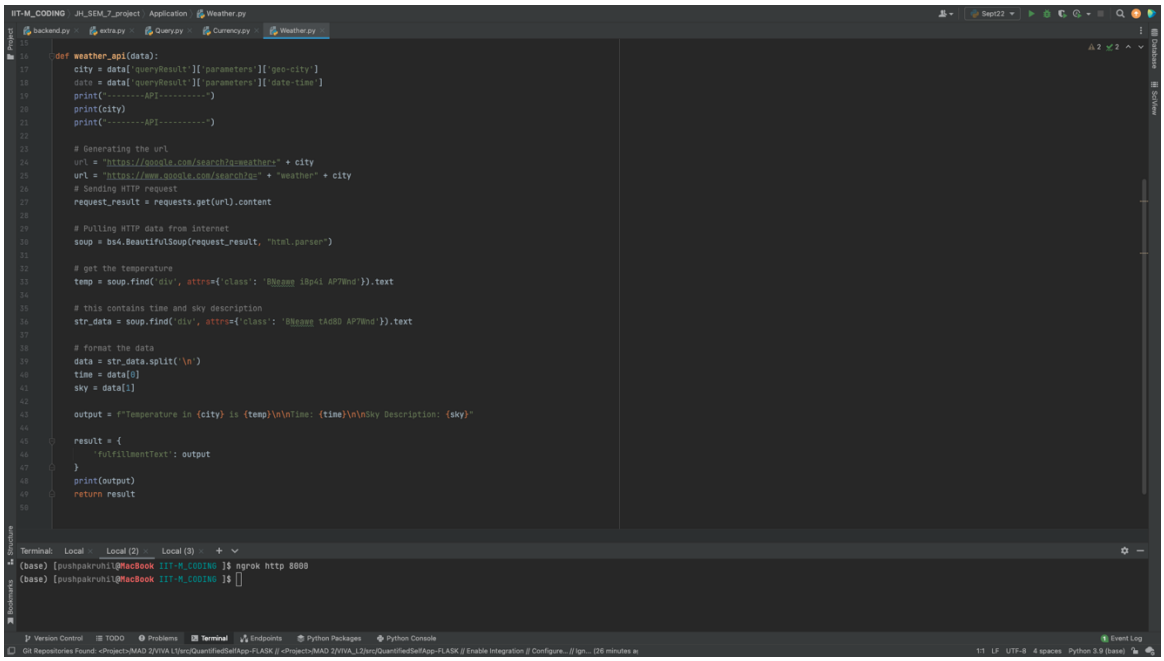
DIFFERENT TEST CASES/SNAPSHOTS

Backend code -

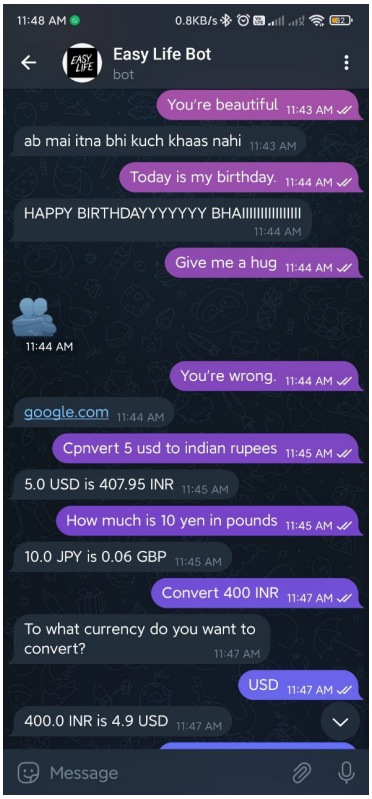


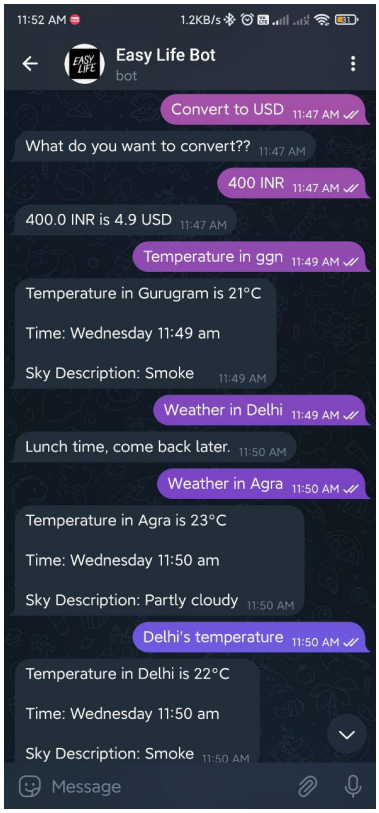


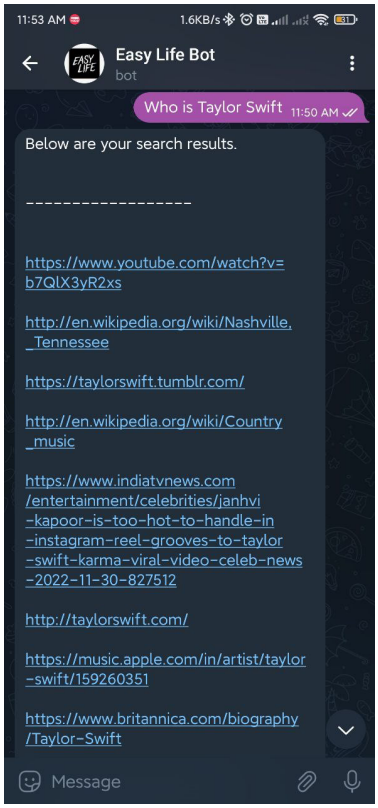

IIT-M_CODING / JH_SEM_7_project / Application / Currency.py
backend.py
extra.py
Query.py
Currency.py
Weather.py
https://apilayer.com/marketplace/exchangerates_data-api#documentation-tab
"""
def currency_converter(data):
try:
curr_amt = data['queryResult']['parameters']['unit-currency']['amount']
convert = data['queryResult']['parameters']['unit-currency']['currency']
into = data['queryResult']['parameters']['currency-name'][0]
url = f"https://api.apilayer.com/exchangerates_data/convert?to={into}&from={convert}&amount={curr_amt}"
headers = {
"apikey": "YYzmAOEyGS7fxZRxXGriFM1NK5SUyQzK"
}
response = requests.request("GET", url, headers=headers, data={})
print("-------API RESPONSE-------")
print(response.text)
print("-------API RESPONSE-------")
converted_amt = round(response.json()["result"], 2)
result = {
'fulfillmentText': f"{curr_amt} {convert} is {converted_amt} {into}"
}
return result
except KeyError:
return {
'fulfillmentText': "My abilities didn't allow me to do that this time. try again? :("
}
currency_converter() › try
Terminal: Local Local (2) Local (3)
(base) [pushpakruhil@MacBook IIT-M_CODING]$ ngrok http 8000
(base) [pushpakruhil@MacBook IIT-M_CODING]$
Version Control TODO Problems Terminal Endpoints Python Packages Python Console
Event Log
Git Repositories Found: <Project>/MAD 2/VIVA L1/src/QuantifiedSelfApp-FLASK // <Project>/MAD 2/VIVA_L2/src/QuantifiedSelfApp-FLASK // Enable Integration // Configure... // Ign... (26 minutes ag
29:19 LF UTF-8 4 spaces Python 3.9 (base)

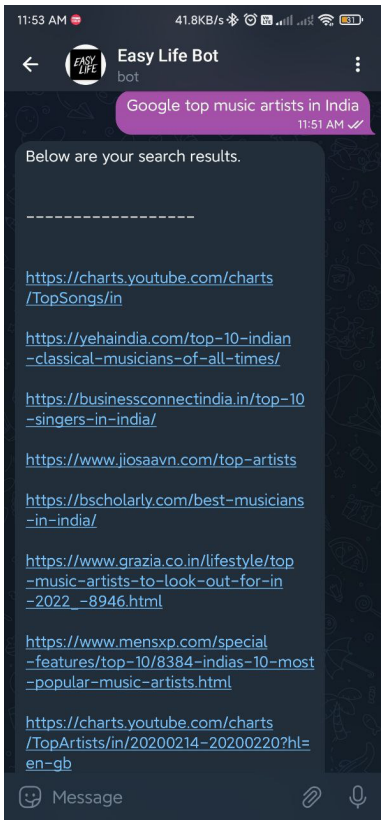

IIT-M_CODING / JH_SEM_7_project / Application / Weather.py
backend.py
extra.py
Query.py
Currency.py
Weather.py
def weather_api(data):
city = data['queryResult']['parameters']['geo-city']
date = data['queryResult']['parameters']['date-time']
print("--------API----------")
print(city)
print("--------API----------")
# Generating the url
url = "https://google.com/search?q=weather+" + city
url = "https://www.google.com/search?q=" + "weather" + city
# Sending HTTP request
request_result = requests.get(url).content
# Pulling HTTP data from internet
soup = bs4.BeautifulSoup(request_result, "html.parser")
# get the temperature
temp = soup.find('div', attrs={'class': 'BNeawe iBp4i AP7Wnd'}).text
# this contains time and sky description
str_data = soup.find('div', attrs={'class': 'BNeawe tAd8D AP7Wnd'}).text
# format the data
data = str_data.split('\n')
time = data[0]
sky = data[1]
output = f"Temperature in {city} is {temp}\n\nTime: {time}\n\nSky Description: {sky}"
result = {
'fulfillmentText': output
}
print(output)
return result
Terminal: Local Local (2) Local (3)
(base) [pushpakruhil@MacBook IIT-M_CODING]$ ngrok http 8000
(base) [pushpakruhil@MacBook IIT-M_CODING]$
Version Control TODO Problems Terminal Endpoints Python Packages Python Console
Event Log
Git Repositories Found: <Project>/MAD 2/VIVA L1/src/QuantifiedSelfApp-FLASK // <Project>/MAD 2/VIVA_L2/src/QuantifiedSelfApp-FLASK // Enable Integration // Configure... // Ign... (26 minutes ag
1:1 LF UTF-8 4 spaces Python 3.9 (base)

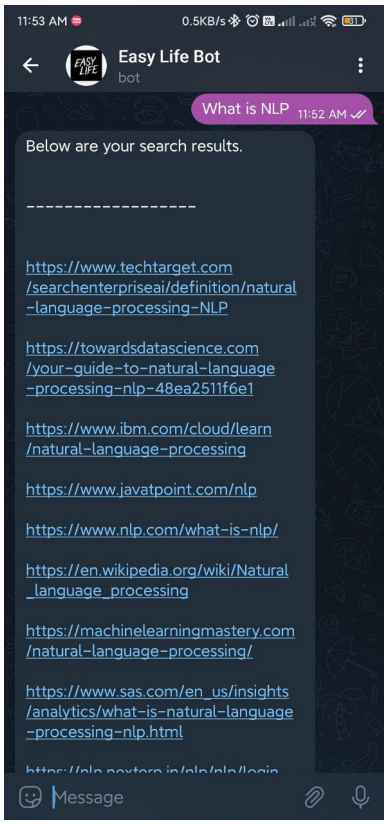
Demo Screenshots –








11:53 AM
41.8KB/s
Easy Life Bot
bot
Google top music artists in India
11:51 AM
Below are your search results.
---
https://charts.youtube.com/charts/TopSongs/in
https://yehaindia.com/top-10-indian-classical-musicians-of-all-times/
https://businessconnectindia.in/top-10-singers-in-india/
https://www.jiosaavn.com/top-artists
https://bscholarly.com/best-musicians-in-india/
https://www.grazia.co.in/lifestyle/top-music-artists-to-look-out-for-in-2022_-8946.html
https://www.mensxp.com/special-features/top-10/8384-indias-10-most-popular-music-artists.html
https://charts.youtube.com/charts/TopArtists/in/20200214-20200220?hl=en-gb
Message


11:53 AM
0.5KB/s
Easy Life Bot
bot
What is NLP
11:52 AM
Below are your search results.
---
https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP
https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1
https://www.ibm.com/cloud/learn/natural-language-processing
https://www.javatpoint.com/nlp
https://www.nlp.com/what-is-nlp/
https://en.wikipedia.org/wiki/Natural_language_processing
https://machinelearningmastery.com/natural-language-processing/
https://www.sas.com/en_us/insights/analytics/what-is-natural-language-processing-nlp.html
Message

CONCLUSION

As I conclude my project report, I would like to summarize this project.
This AI enabled chatbot is an easy to use application to perform daily life actions, all at one single place. This can also be used for small talks which can help lighten your mood.

This is just one single use case of a chatbot that I have implemented, but there are various industry level use cases as well. Many companies use such chatbot agents to reduce the manual labor, say for customer service.

Limitations:

- Need a platform for integration to be able to use. Here, it's telegram.

- Here, we are making use of localhost tunnelling. The URL for the tunnel changes every time (because of the free version). This can be fixed by either purchasing a paid version or hosting the server on some other free service like Heroku.

- Sometimes the webhook response received is slow(due to computational complexities). This triggers to chatbot to reply with an automated reply that the server is down, even if it's not down. Hence, optimization is required.

BIBLIOGRAPHY

- Flask: https://flask.palletsprojects.com/
- Ngrok: https://ngrok.com/
- DialogFlow: https://dialogflow.cloud.google.com/
  https://cloud.google.com/dialogflow/docs