

Major Project Report
On
**Ai ChatBot for Polytechnic College
Counselling**

Submitted in Partial Fulfillment for the Award
Of
Diploma in Computer Science Engineering
(2021-22)



**RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALYA
BHOPAL (M.P.)**

Submitted

By

Abhishek Dwivedi

Under the Supervision

of

Mr. Mahendra Gupta



**GOVERNMENT POLYTECHNIC COLLEGE ANUPPUR
MP-484224**

Certificate

This is to certify that project entitled “**Ai ChatBot for Polytechnic College Counselling** ” is being submitted by **Mr. Abhishek Dwevidi** to the **Dept. of Computer Science and Engineering, Govt Polytechnic College Anuppur, M.P.-484224, India**, in the partial fulfillment of the requirements for the award of the diploma in “**Computer Science and Engineering**”. This work is carried out by himself in the Dept. of Computer Science and Engineering under the supervision of **Mr. Mahendra Gupta**. The matter personified in the project report has not been submitted for the award of any other degree or diploma.

Mr. Raju Singh Paraste
(Principal)

Mr. Utkarsh Agrawal
(HoD)

Mr. Mahendra Gupta
(Supervisor)

Contents

Certificate	2
DECLARATION	4
1. INTRODUCTION	6
2. OBJECTIVE	7
3. FLOW CHART	8
4. HARDWARE REQUIREMENT	9
5. SOFTWARE REQUIREMENT	9
5.1 VS CODE	9
5.2 PYTHON	9
5.3 RASA FRAMEWORK.....	9
5.4 INSTALLING PROCESS OF RASA:-	9
6. WORKING	11
6.1. Actions.....	12
6.2. DATA:-	12
6.3. NLU.yml:-.....	13
6.4. STORIES.yml:-	14
6.5. RULE.yml:-	15
6.6. MODEL:-	16
6.7. CONFIG.yml:-	17
6.7.1 PIPELINE:-	17
6.7.2 Tokenizers:-.....	18
6.7.3 FEATURES:-	19
6.7.4 Intent Classifiers:-	19
6.7.5 Entity Extraction:-.....	20
7. credentials.yml:-	21
8. Domain.yml:-	22
9. Process to Deploy Chatbot in Website:-	23
10. Process to Integrate into Whatsapp using rasa:-	24
10.1 Stpes to make Whats App Chat bot:-	24
11. REFERENCES: -	27

DECLARATION

We hereby declare that the work which is being presented in the project report fulfillment of the requirement of the “Diploma in Computer Science” branch is an Authentic record of our work carried out under the guidance of “Mr. Mahendra Gupta(lecturer)”. The work has been carried out at Govt. Polytechnic College Anuppur(M.P).

PROJECT ASSOCIATES:-

 → ABHISHEK DWIVEDI

ACKNOWLEDGMENT

A project like one involves many people and it would be complete without the mention of all those people whose guidance and encouragement helped in the successful completion of this project.

We heartily thank our faculty member of the Department of Computer Science Govt. Polytechnic College Anuppur for their efforts towards our project.

We would like to thank our project in-charge Mr. Mahendra Gupta who has been a great source of inspiration for us and without whose humble guidance the project was never shaped.

We are also indebted to our guide Mr. Mahendra Gupta for the Encouragement Guidance and Support.

We are also thankful to all the many people whose timely help over the paucity of space is restricting us from their name.

And finally, we also thank all my colleagues who were constant support during the whole project.

1. INTRODUCTION

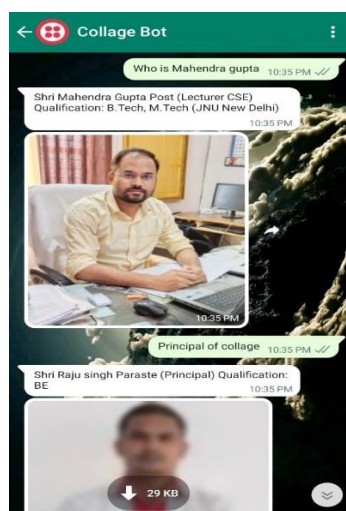
I'm sure each of us would have interacted with a bot, sometimes without even realizing it! We are in the era of Conversational AI now. Every website uses a Chatbot to interact with the users and help them out. This has proven to reduce the time and resources to a great extent. At the same time, bots that keep sending "Sorry I did not get you" just irritate us. You need to ensure that the performance is satisfactory too.

Rasa open source provides an advanced and smooth way to build your chatbot that can provide satisfactory interaction.

We can put RASA Chatbot on your website and connect to the database after that RASA will manage the user coming to our website like this:-



And with this, we can also make a Bot for WhatsApp like this: -



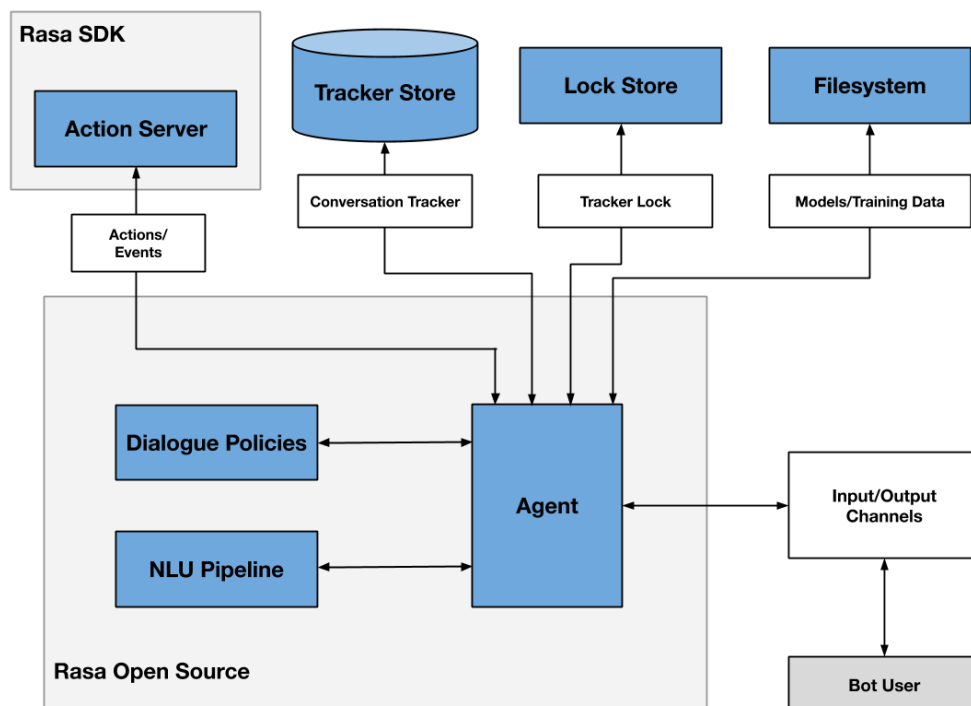
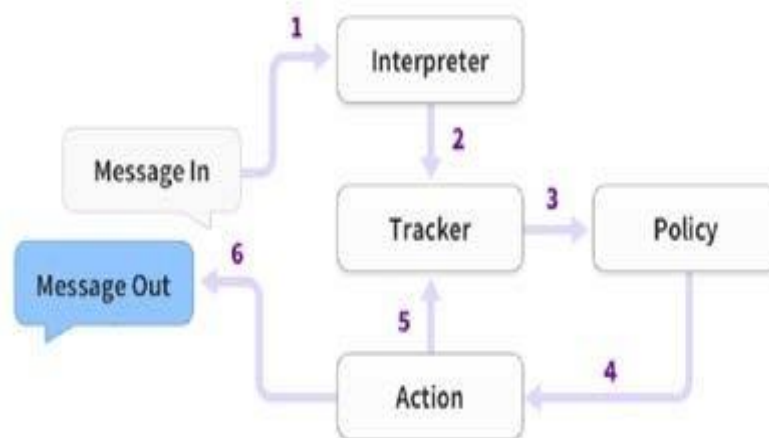
2. OBJECTIVE

A chatbot can communicate with a real person by behaving like a human.

The objective of the Rasa chatbot is that it can help the college. In most cases, the student does not know how to take admission. How many branches in this college and there are many other problems. With the help of this chatbot, we can solve those problems.

This chatbot can solve all the problems related to the college like can tell the process of admission. It can tell how many branches are there in the college. It can tell about the staff of the college and it can do many other things related to the college.

3. FLOW CHART



4. HARDWARE REQUIREMENT

PROCESSOR	i3
RAM	8GB
STORAGE	512 GB

5. SOFTWARE REQUIREMENT

IDE	Vs Code
LANGUAGE	Python
FRAMEWORK	Rasa FrameWork

5.1 VS CODE

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS, and Linux. It comes with built-in support for JavaScript, TypeScript, and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity)

5.2 PYTHON

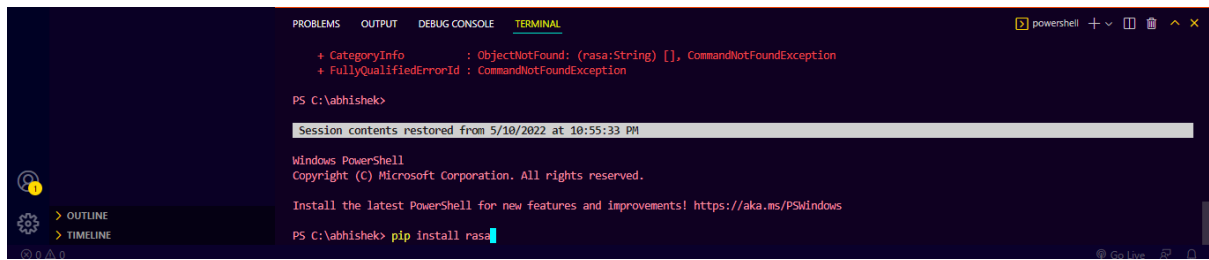
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components.

5.3 RASA FRAMEWORK

Rasa is a framework for developing AI-powered, industrial-grade chatbots. It's incredibly powerful and is used by developers worldwide to create chatbots and contextual assistants. In this project, we are going to understand some of the most important basic aspects of the Rasa framework and chatbot development.

5.4 INSTALLING PROCESS OF RASA:-

To install rasa you just have to type a simple command '`pip install rasa`'.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
+ CategoryInfo          : ObjectNotFound: (rasa:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\abhishek>

Session contents restored from 5/10/2022 at 10:55:33 PM

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

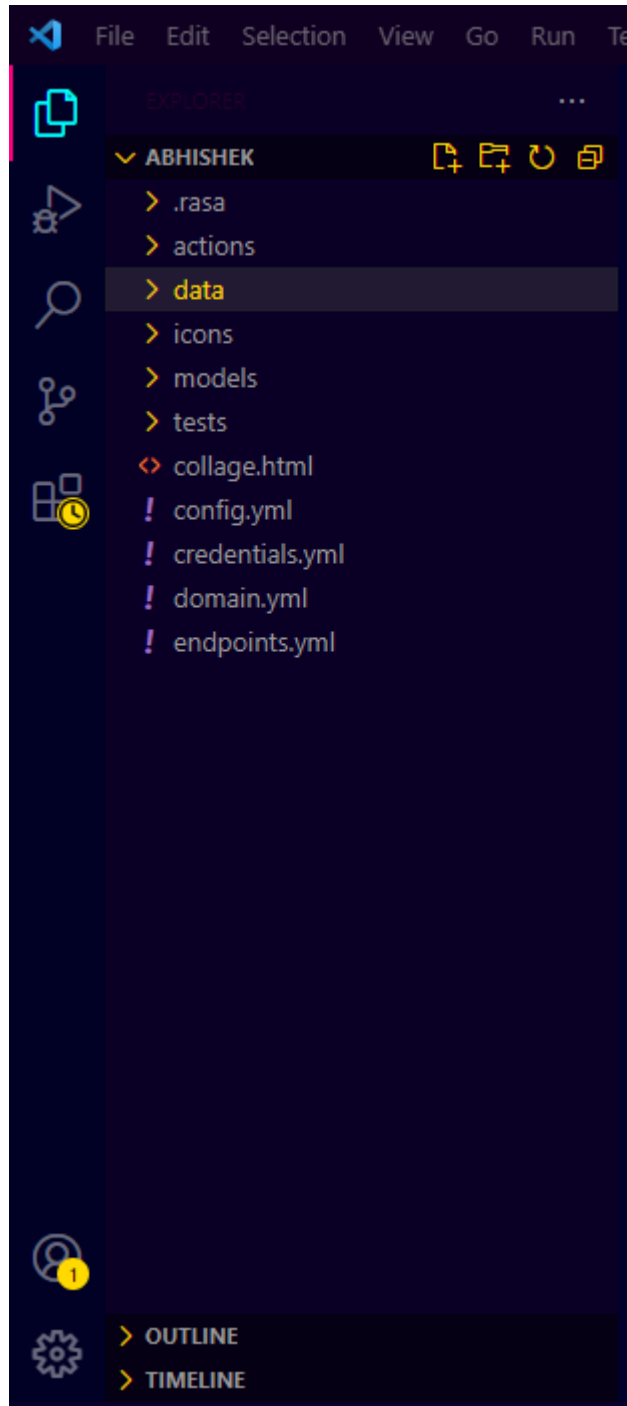
PS C:\abhishek> pip install rasa
```

After that ,your rasa framework will start installing

6. WORKING

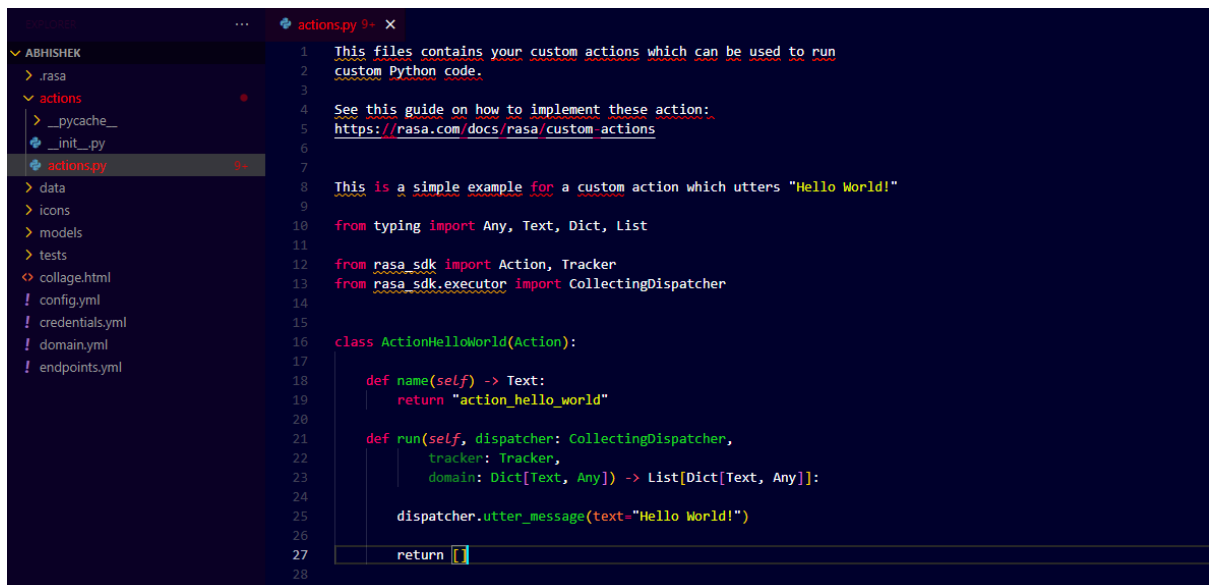
When we install Rasa Framework, we get some preinstalled files and folders these are:-

Action, Data, Model, Domain.yml, Endpoint.yml, Config.yml, and credentials.yml



6.1. Actions

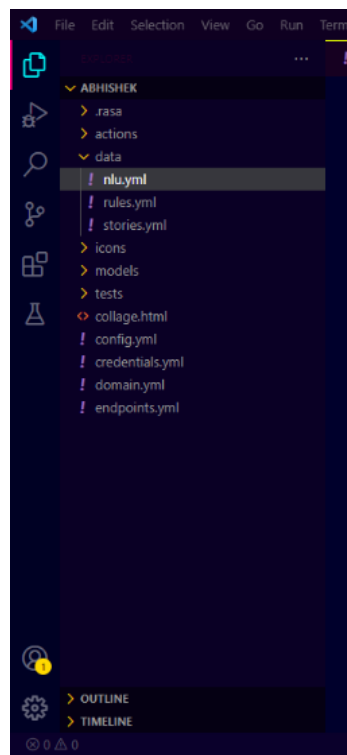
Action is a folder where we get the Action.py file it's a Python Which we use to do things like fetch data from API, to fetch data from the database, etc.



```
1 This files contains your custom actions which can be used to run
2 custom Python code.
3
4 See this guide on how to implement these action:
5 https://rasa.com/docs/rasa/custom-actions
6
7
8 This is a simple example for a custom action which utters "Hello World!"
9
10 from typing import Any, Text, Dict, List
11
12 from rasa_sdk import Action, Tracker
13 from rasa_sdk.executor import CollectingDispatcher
14
15
16 class ActionHelloWorld(Action):
17
18     def name(self) -> Text:
19         return "action_hello_world"
20
21     def run(self, dispatcher: CollectingDispatcher,
22            tracker: Tracker,
23            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
24
25         dispatcher.utter_message(text="Hello World!")
26
27     return []
28
```

6.2. DATA:-

Data is a folder inside which we get some files like Nlu.yml,Rule.yml,Stories.yml

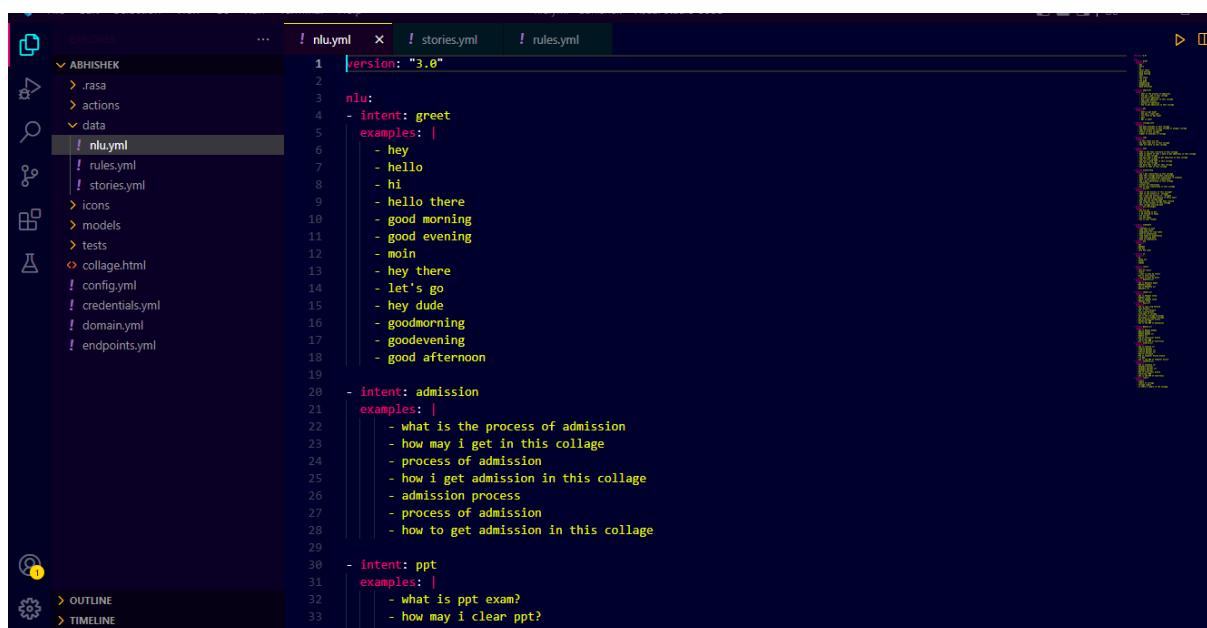


6.3. NLU.yml:-

NLU a file where we train your bot here we specify **Intent** and **Examples**

Intent:- If the intent is explained in simple words, then the intent is also called the subject, which we are talking about like **College Information** it's intent in this we ask things related to college like the location of the college, how many branches in college.

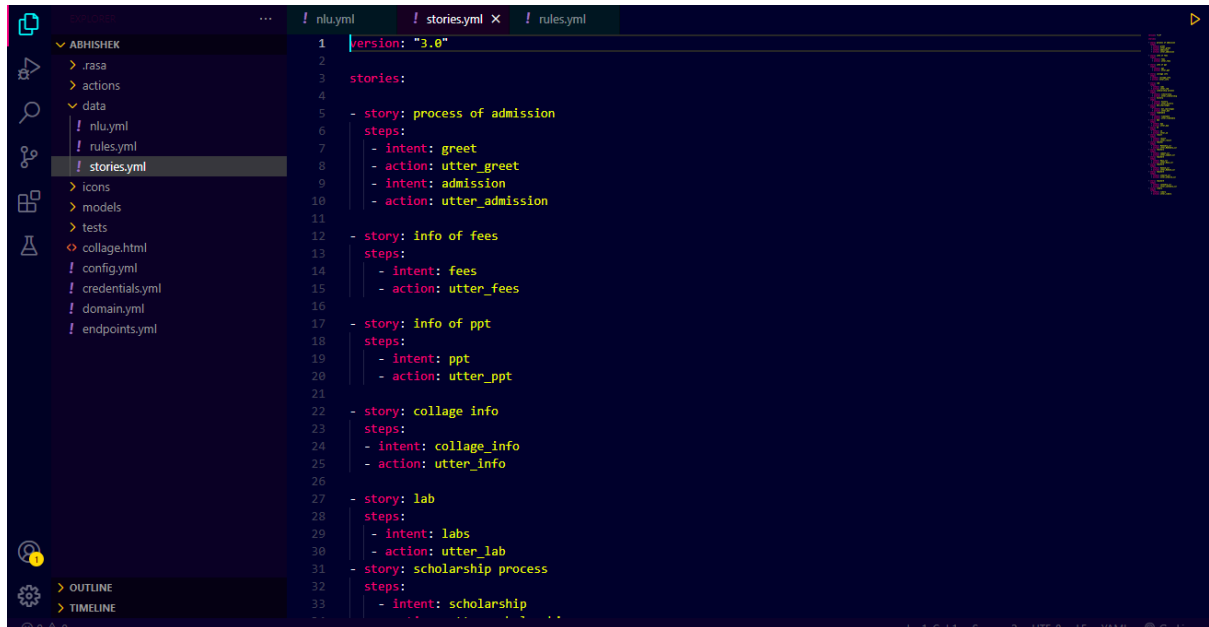
Example:- Example is that in which we tell the bot how the user can ask the question to you like Branches in this collage, How many Branches in this collage, Is there C.s Branch in this collage



```
1 version: "3.0"
2
3 nlu:
4   - intent: greet
5     examples: |
6       - hey
7       - hello
8       - hi
9       - hello there
10      - good morning
11      - good evening
12      - moin
13      - hey there
14      - let's go
15      - hey dude
16      - goodmorning
17      - goodevening
18      - good afternoon
19
20   - intent: admission
21     examples: |
22       - what is the process of admission
23       - how may i get in this collage
24       - process of admission
25       - how i get admission in this collage
26       - admission process
27       - process of admission
28       - how to get admission in this collage
29
30   - intent: ppt
31     examples: |
32       - what is ppt exam?
33       - how may i clear ppt?
```

6.4. STORIES.yml:-

Stories.yml is a file that is found inside the Data folder, as we have seen in **nlu.yml** where we specify the **intent**, in **Stories.yml** we specify **an action**, that if this intent occurs then perform this action



```
1 |version: "3.0"
2
3 stories:
4
5 - story: process of admission
6   steps:
7     - intent: greet
8     - action: utter_greet
9     - intent: admission
10    - action: utter_admission
11
12 - story: info of fees
13   steps:
14     - intent: fees
15     - action: utter_fees
16
17 - story: info of ppt
18   steps:
19     - intent: ppt
20     - action: utter_ppt
21
22 - story: collage info
23   steps:
24     - intent: collage_info
25     - action: utter_info
26
27 - story: lab
28   steps:
29     - intent: labs
30     - action: utter_lab
31
32 - story: scholarship process
33   steps:
34     - intent: scholarship
```

6.5. RULE.yml:-

Like Stories.yml Rule.yml is also found in the Data folder, in **Rule.yml** we define some Rules for a bot like whenever a user say Bye, **you immediately exit and says bye to him**, and also whenever the user asked who are you, **you should immediately tell them that I am a bot**

```
1  version: "3.0"
2
3  rules:
4
5  - rule: Say goodbye anytime the user says goodbye
6    steps:
7      - intent: goodbye
8      - action: utter_goodbye
9
10 - rule: Say 'I am a bot' anytime the user challenges
11   steps:
12     - intent: bot_challenge
13     - action: utter_iamabot
14
```

6.6. MODEL:-

Model is a folder where our Rasa model Store Whenever we train a model, it becomes stored in this folder



Process of Training Model:-

Whenever we make a chatbot we need to train them it is similar to how we compile a program by creating it

To train the bot we need to use the command “Rasa Train”

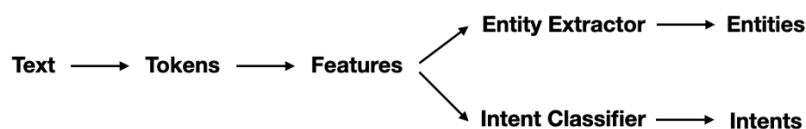
6.7. CONFIG.yml:-

If we talk about the most important thing in the rasa chatbot, it's a config.yml because it has a **Pipeline**

```
7 language: en
8
9 pipeline:
10 # No configuration for the NLU pipeline was provided. The following default pipeline was used to train your model.
11 # If you'd like to customize it, uncomment and adjust the pipeline.
12 # See https://rasa.com/docs/rasa/tuning-your-model for more information.
13 - name: WhitespaceTokenizer
14 - name: RegexFeaturizer
15 - name: LexicalSyntacticFeaturizer
16 - name: CountVectorsFeaturizer
17 - name: CountVectorsFeaturizer
18   analyzer: char_wb
19   min_ngram: 1
20   max_ngram: 4
21 - name: DIETClassifier
22 epochs: 100
23 constrain_similarities: true
24 - name: EntitySynonymMapper
25 - name: ResponseSelector
26 epochs: 100
27 constrain_similarities: true
28 - name: FallbackClassifier
29 threshold: 0.3
30 ambiguity_threshold: 0.1
31
32 Configuration for Rasa Core.
33 https://rasa.com/docs/rasa/core/policies/
34 policies:
35 # No configuration for policies was provided. The following default policies were used to train your model.
36 # If you'd like to customize them, uncomment and adjust the policies.
37 # See https://rasa.com/docs/rasa/policies for more information.
38 - name: MemoizationPolicy
39 - name: RulePolicy
```

6.7.1 PIPELINE:-

The NLU pipeline defines the processing steps that convert unstructured user messages into intents and entities. It consists of a series of components, which can be configured and customized by developers.



There are different types of components that you can expect to find in a pipeline. The main ones are:

- 1) Tokenizers
- 2) Features
- 3) Intent Classifiers
- 4) Entity Extractors

6.7.2 Tokenizers:-

The first step is to split an utterance into smaller chunks of text, known as tokens. This must happen before text is featured for machine learning, which is why you'll usually have a tokenizer listed first at the start of a pipeline.

"Hi, my name is Vincent." → **["Hi", "my", "name", "is", "Vincent"]**

Details on Tokenizers.

Just Tokenisation

"He likes dogs" → ["He", "likes", "dogs"]

Tokenisation and Lemmatisation

"He likes dogs" → ["He", "like", "dog"]

Some tokenizers also add extra information to the tokens. For example, spaCy is able to also generate lemmas of the tokens which can later be used by the CountVectorizer.

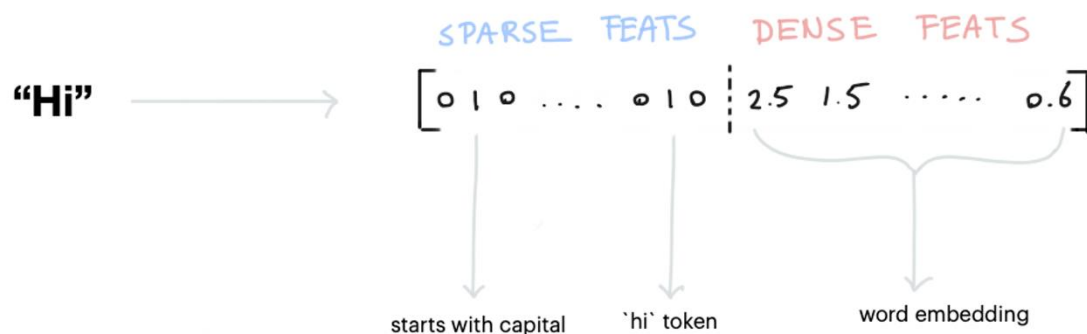
The tokenizer splits each individual word in the utterance into a separate token, and commonly the output of the tokenizer is a list of words. We might also get separate tokens for punctuation depending on the tokenizer and the settings that we pass through.

For English, we usually use the WhiteSpaceTokenizer but for non-English it can be common to pick other ones. SpaCy is a good choice for non-English European languages but Rasa also supports Jieba for Chinese.

Note that tokenizers don't change the underlying text, they only separate text into tokens. That means, for example, that capitalisation remains untouched. It might be that you'd like to only encode the lower case text for your pipeline, but adding this kind of information is the job of a featurizer, which we'll discuss next.

6.7.3 FEATURES:-

Features generate numeric features for machine learning models. The diagram below shows how the words "Hi", "might", and "be" are encoded.



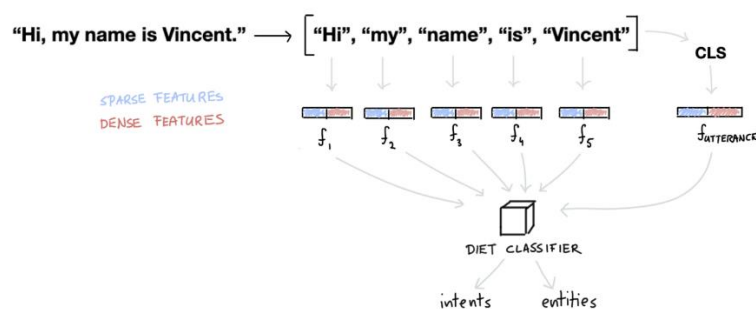
There are two types of features:

Sparse Features: usually generated by a CountVectorizer. Note that these counts may represent subwords as well. We also have a LexicalSyntacticFeaturizer that generates window-based features useful for entity recognition. When combined with spaCy, the LeticalSyntacticFeaturizer can be configured to also include part of speech features.

Dense Features: these consist of many pre-trained embeddings. Commonly from SpaCyFeaturizers or hugging face via LanguageModelFeaturizers. If you want these to work, you should also include an appropriate tokenizer in your pipeline. More details are in the documentation.

6.7.4 Intent Classifiers:-

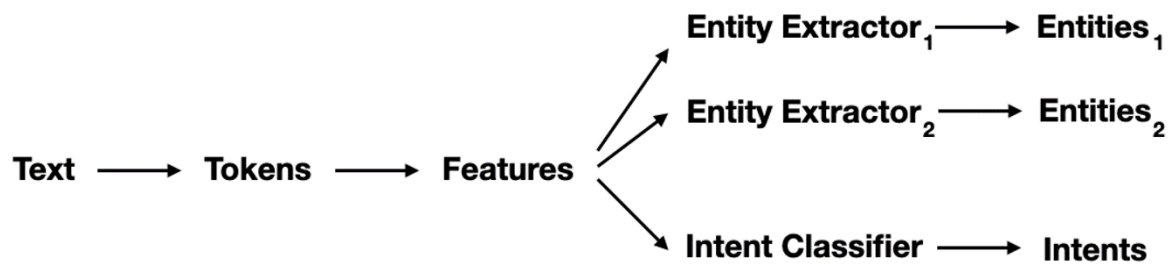
Once we've generated features for all of the tokens and the entire sentence, we can pass it to an intent classification model. We recommend using Rasa's DIET model which can handle both intent classification as well as entity extraction. It is the token- as well as sentence features



6.7.5 Entity Extraction:-

Even though DIET is capable of learning how to detect entities, we don't necessarily recommend using it for every type of entity out there. For example, entities that follow a structured pattern, like phone numbers, don't need an algorithm to detect them. You can just handle it with a `RegexEntityExtractor` instead.

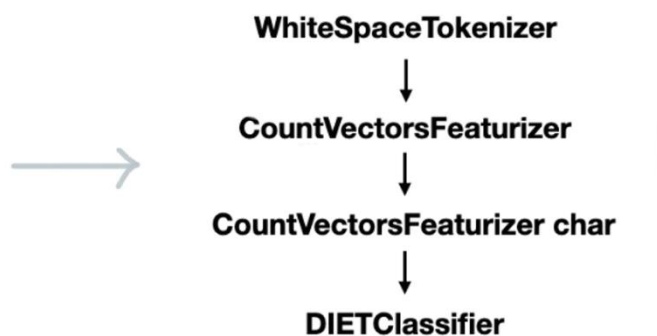
This is why it's common to have more than one type of entity extractor in the pipeline



Interaction: Message Passing:-

As you can imagine, the components in a Rasa pipeline depend on each other. So you might be wondering how they interact.

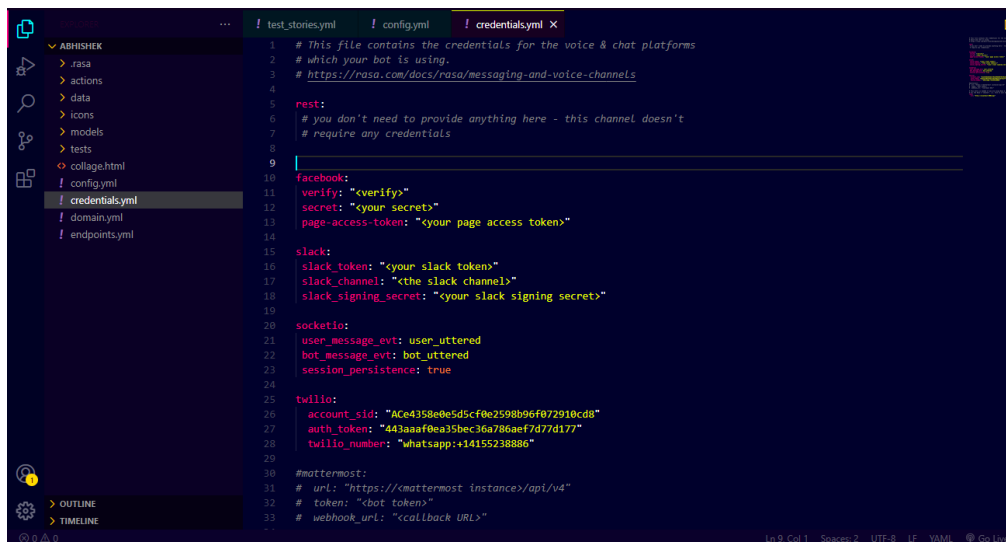
```
pipeline:
- name: WhitespaceTokenizer
- name: CountVectorsFeaturizer
- name: CountVectorsFeaturizer
  analyzer: char_wb
  min_ngram: 1
  max_ngram: 4
- name: DIETClassifier
  epochs: 100
```



7. credentials.yml:-

Here things are related to server like in which port rasa chatbot will run, we need to specify port in which chatbot runs like:- **Port:5005**

Here also we write program /code to link the website,Whatspps,Teligram



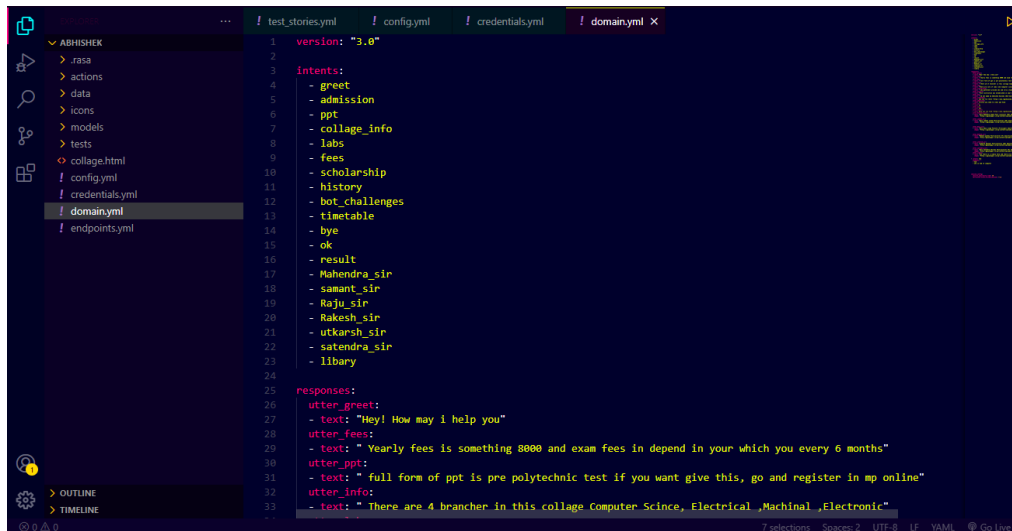
```
1 # This file contains the credentials for the voice & chat platforms
2 # which your bot is using.
3 # https://rasa.com/docs/rasa/messaging-and-voice-channels
4
5 rest:
6 # you don't need to provide anything here - this channel doesn't
7 # require any credentials
8
9
10 facebook:
11   verify: "<verify>"
12   secret: "<your secret>"
13   page-access-token: "<your page access token>"
14
15 slack:
16   slack_token: "<your slack token>"
17   slack_channel: "<the slack channel>"
18   slack_signing_secret: "<your slack signing secret>"
19
20 socketio:
21   user_message_evt: user_uttered
22   bot_message_evt: bot_uttered
23   session_persistence: true
24
25 twilio:
26   account_sid: "Aced358e0e5d5cf0e2598b96f872910cd8"
27   auth_token: "443aaa0ea35bec36a786aef7d77d177"
28   twilio_number: "whatsapp:+14155238886"
29
30 #mattermost:
31 # url: "https://<mattermost instance>/api/v4"
32 # token: "<bot token>"
33 # webhook_url: "<callback URL>"
```

As you can see in the above image that there is **Facebook, Twilio** is connected to all these websites, To put chatbot in a **Website, Account, Server** we need to specify some things like **Account, Number, server name, Token**

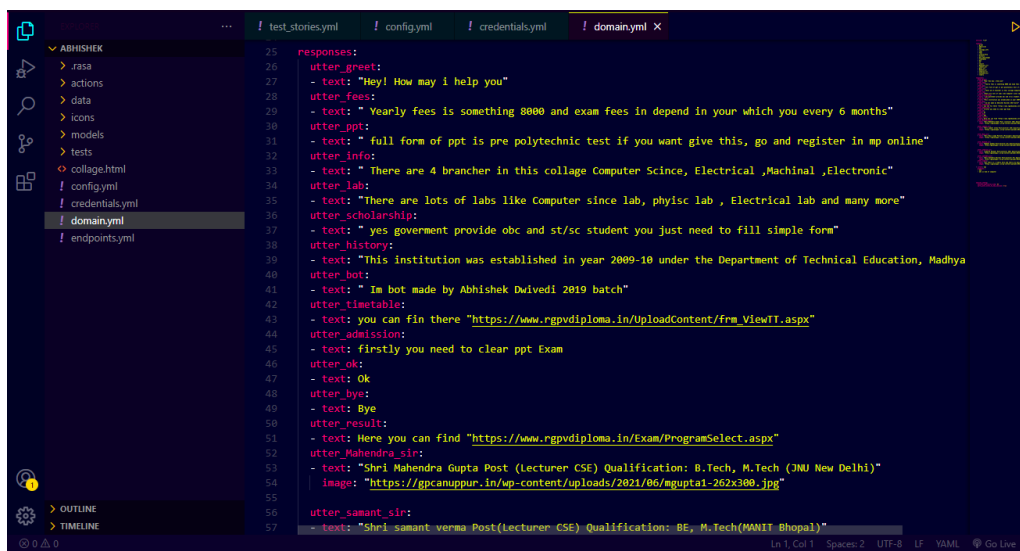
8. Domain.yml:-

Domain.yml is a very important file. Earlier you saw in stories.yml that we used to classify actions, These actions are written here

Here we have to write intent which we keep writing in nlu.yml. After that we have to write the actions for intent



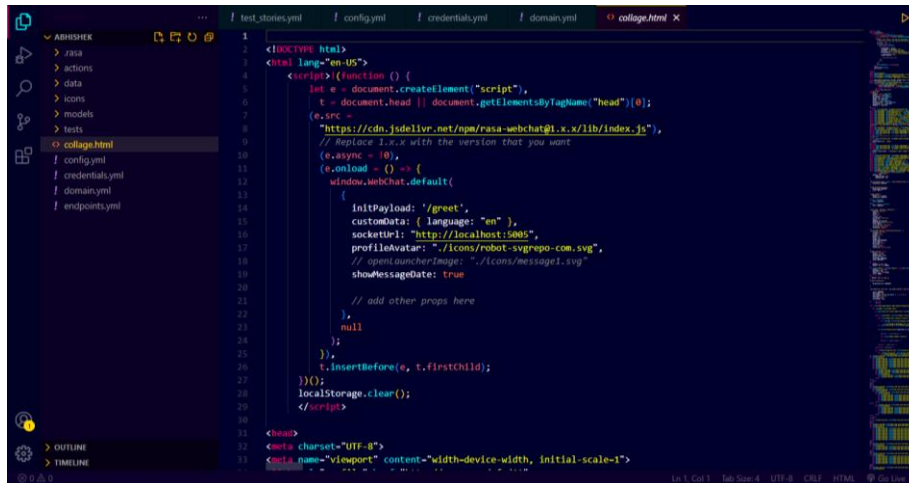
```
1 version: "3.0"
2
3 intents:
4   - greet
5   - admission
6   - ppt
7   - collage_info
8   - labs
9   - fees
10  - scholarship
11  - history
12  - bot_challenges
13  - timetable
14  - bye
15  - ok
16  - result
17  - Mahendra_sir
18  - samant_sir
19  - Raju_sir
20  - Rakesh_sir
21  - utkarsh_sir
22  - satendra_sir
23  - library
24
25 responses:
26   utter_greet:
27     - text: "Hey! How may i help you"
28   utter_fees:
29     - text: " Yearly fees is something 8000 and exam fees in depend in your which you every 6 months"
30   utter_ppt:
31     - text: " full form of ppt is pre polytechnic test if you want give this, go and register in mp online"
32   utter_info:
33     - text: " There are 4 brancher in this collage Computer Scince, Electrical ,Machinal ,Electronic"
```



```
25 responses:
26   utter_greet:
27     - text: "Hey! How may i help you"
28   utter_fees:
29     - text: " Yearly fees is something 8000 and exam fees in depend in your which you every 6 months"
30   utter_ppt:
31     - text: " full form of ppt is pre polytechnic test if you want give this, go and register in mp online"
32   utter_info:
33     - text: " There are 4 brancher in this collage Computer Scince, Electrical ,Machinal ,Electronic"
34   utter_lab:
35     - text: "There are lots of labs like Computer since lab, physis lab , Electrical lab and many more"
36   utter_scholarship:
37     - text: " yes government provide obc and st/sc student you just need to fill simple form"
38   utter_history:
39     - text: "This institution was established in year 2009-10 under the Department of Technical Education, Madhya
40   utter_bot:
41     - text: " Im bot made by Abhishek Dwivedi 2019 batch"
42   utter_timetable:
43     - text: you can fin there "https://www.rgpvdiploma.in/UploadContent/frw\_ViewTT.aspx"
44   utter_admission:
45     - text: firstly you need to clear ppt Exam
46   utter_ok:
47     - text: Ok
48   utter_bye:
49     - text: Bye
50   utter_result:
51     - text: Here you can find "https://www.rgpvdiploma.in/Exam/ProgramSelect.aspx"
52   utter_Mahendra_sir:
53     - text: "Shri Mahendra Gupta Post (Lecturer CSE) Qualification: B.Tech, M.Tech (JNU New Delhi)"
54       image: "https://gpcanuppur.in/wp-content/uploads/2021/06/mgupta1-262x300.jpg"
55   utter_samant_sir:
56     - text: "Shri samant verma Post(Lecturer CSE) Qualification: BE, M.Tech(MANIT Bhopal)"
```

9. Process to Deploy Chatbot in Website:-

To Deploy Rasa Chatbot in your website, you have to write some code of rasa chatbot in your **HTML** content .It is written with the help of JavaScript



```
1 <!DOCTYPE html>
2 <html lang="en-US">
3 <script>((function () {
4     let e = document.createElement("script"),
5         t = document.head || document.getElementsByTagName("head")[0];
6     (e.src =
7         "https://cdn.jsdelivr.net/npm/rasa-webchat@1.x.x/lib/index.js"),
8         // replace 1.x.x with the version that you want
9     (e.async = !0),
10    (e.onload = () => {
11        window.webChat.default(
12            {
13                initPayload: {/greet/,
14                customData: { language: "en" },
15                socketUrl: "http://localhost:5005",
16                profileAvatar: "/icons/robot-svgrepo-com.svg",
17                // openLauncherImage: "/icons/message1.svg"
18                showMessageDate: true
19            },
20            // add other props here
21            null
22        );
23    });
24    t.insertBefore(e, t.firstChild);
25    })());
26    localStorage.clear();
27    </script>
28 </html>
29 <meta charset="UTF-8">
30 <meta name="viewport" content="width=device-width, initial-scale=1">
```

In this code we need to specify Socket Url is where your website is running if your website is running on your Server you need to specify server Address / Domain name. If your website is not running in any server then simply write **Localhost**

Also in this code there is a **Payload**, you can understand payload in such a way that whether the bot will say something automatically, or the user sends a message to him, then he will response

I specify Greeting intent in **Payload** in greeting intent there are some message like hii,hello,how are, goodmorning its means when ever user truns chatbot ,bot will message it self like:-



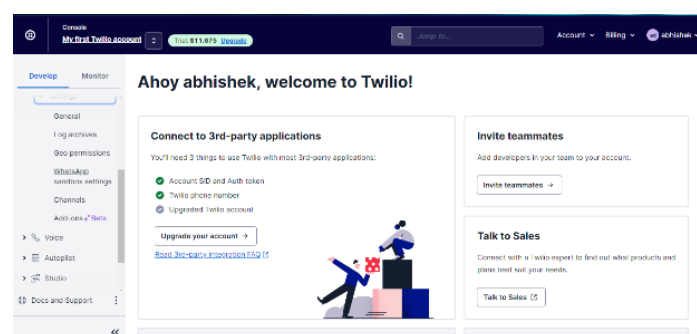
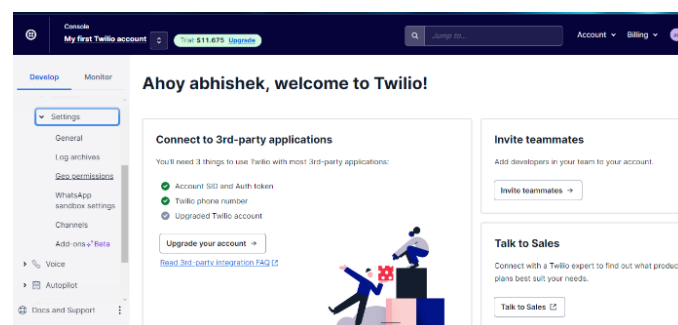
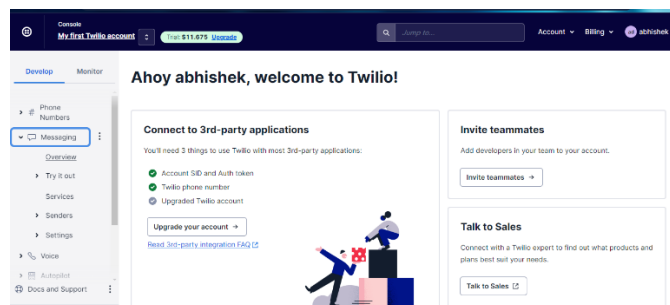
10. Process to Integrate into Whatsapp using rasa:-

If you don't have a server then you need to use a third-party server and ngrok software, ngrok software is software that puts local host on internet

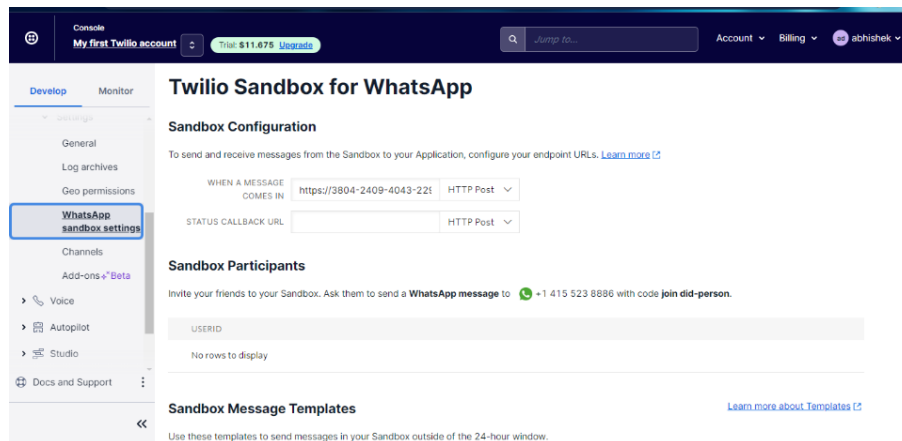
10.1 Steps to make WhatsApp Chat bot:-

Step 1:- Create Account in Twilio, Twilio is a server and also provides WhatsApp business API

Step 2:- After creating account, you have to go to the **Message Section** then in the **Settings** section the **WhatsApp sand Box settings**



Sept4:- When you reach here, you see like this Inter Face



Here you can see a section **When a Message comes in**, here you have to enter the server link where your chat bot is running, because your bot is running in local host then you need help of ngrok to put localhost in the internet

Step5:- But before doing all this, you have to tell rasa chatbot that on which number it will have to be activated or not and which server it will run

Earlier I told you about **credential.yml** where things related to server are written here you have to **specify SID, Authorize Token, and Twilio number** if you do not have WhatsApp business api, if you have then simply paste it

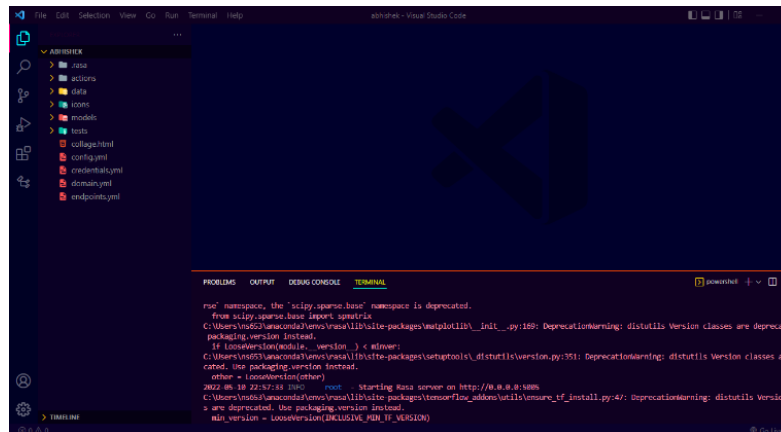
```

1  # you don't need to provide anything here - this channel doesn't
2  # require any credentials
3
4
5
6
7
8
9
10 #facebook:
11 #  verify: "<verify>"
12 #  secret: "<your secret>"
13 #  page-access-token: "<your page access token>"
14
15 #slack:
16 #  slack_token: "<your slack token>"
17 #  slack_channel: "<the slack channel>"
18 #  slack_signing_secret: "<your slack signing secret>"
19
20 # socketio:
21 #  user_message_evt: user_uttered
22 #  bot_message_evt: bot_uttered
23 #  session_persistence: true
24
25 twilio:
26   account_sid: "Ac4358e0e5d5cf0e2598b96f072910cd8"
27   auth_token: "443aaaf0ea35bec36a786aef7d77d177"
28   twilio_number: "whatsapp:+14155238886"
29
30
31 #mattermost:
32 #  url: "https://<mattermost instance>/api/v4/"
33 #  token: "<bot token>"
34 #  webhook_url: "<callback URL>"
35
36 # This entry is needed if you are using Rasa X. The entry represents credentials
37 # for the Rasa X "channel", i.e. Talk to your bot and Share with guest testers.
38 rasa:
39   url: "http://localhost:5002/api"
40

```

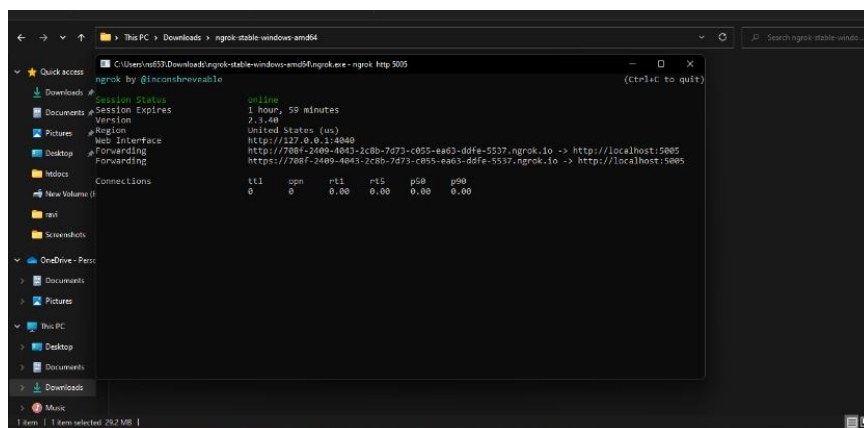
Step6:- After doing all this you simply have to train your rasa model one more time so that he know every thing

Step7:- After doing all this, you just have to type a simple command ,so that your rasa chat bot starts running like this



```
python -m rasa run
rasa namespace, the 'scipy.sparse.base' namespace is deprecated.
from scipy.sparse.base import sparse
C:\Users\india\anaconda3\envs\rasa\lib\site-packages\matplotlib\_init_.py:189: DeprecationWarning: distutils Version classes are deprecated
  packaging.version instead.
  if LooseVersion(packlib._version_) < MINVER:
C:\Users\india\anaconda3\envs\rasa\lib\site-packages\setuptools\distutils\version.py:351: DeprecationWarning: distutils Version classes are
  deprecated. Use packaging.version instead.
  other = LooseVersion(other)
2022-08-10 20:22:54.000 - root - Starting Rasa server on http://0.0.0.0:5005
C:\Users\india\anaconda3\envs\rasa\lib\site-packages\tensorflow\utils\tf_install.py:47: DeprecationWarning: distutils Version
  are deprecated. Use packaging.version instead.
  min_version = LooseVersion(distutils.version)
min_version = LooseVersion(distutils.version)
```

Step8:- After that you have to put port in which rasa chat bot is running like this



Step9:- As you can see , here after specifying the port we get link we have to put this link in Tiwlio

After that we need to type small command then after our bot start given response like that:-



11. REFERENCES: -

[1]For web site <https://youtu.be/eJMT2FovZsM>

[2]For whatsapp chatbot <https://youtu.be/K7boxP8Q50M>

[3] Pipeline <https://rasa.com/blog/intents-entities-understanding-the-rasa-nlu-pipeline/>

[4] What is Rasa <https://rasa.com/>