

**An**  
**Internship Project Report**  
**On**  
**ChatBot with Python**



**Supervisor:**

**Mr Ashvini Jangir**  
**(Corporate Trainer &**  
**Software Developer,**  
**C-DAC, Jaipur)**

**Submitted By:**

**Pradeep Kumar Yadav**  
**(16TEC2CS018)**

**C-DAC ATC NETCOM, JAIPUR**

IT-23, SITAPURA INDUSTRIAL AREA, EPIP, NEAR GLOBAL CIRCLE, Jaipur

**May 2020**

## **Certificate**

This is to be certified that the work, which is being presented in the project report titled **“ChatBot with Python”**, submitted by **“Pradeep Kumar Yadav”**, in partial fulfillment of the award of **“Internship”** in **“Data Science”**, submitted in **“C-DAC ATC Netcom, Jaipur”** is an authentic record of the work under the supervision and valuable guidance of **“Mr Ashvini Jangir”**.

The Matter Presented in the report embodies the result of the studies carried out by the students and has not been submitted for the award of any degree in this any other institute.

**Mr Ashvini Jangir**

**Submitted Date:- \_\_/\_\_/20\_\_**

**(Corporate Trainer &**

**Software Developer, C-DAC, Jaipur)**

## **Acknowledgement**

It is our pleasure to express our heartfelt thanks to **Mr. Ashvini Jangir**, Corporate Trainer and Software Developer (C-DAC, Jaipur) for his supervision and guidance which enabled us to understand and develop this project.

I am highly indebted to **Net PARAM Technologies (C-DAC ATC)** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express our gratitude towards our parents & the members of **Net PARAM Technologies (C-DAC ATC)** for their kind co-operation and encouragement which helped us in the completion of this project and internship. I would like to express our gratitude and thanks to industry persons for giving us such attention and time.

Lastly, we take this opportunity to offer our regards to all of those who have supported us directly or indirectly in the successful completion of this project work.

**Pradeep Kumar Yadav**

## Company Profile



### About C-DAC ATC NETCOM Jaipur

Centre for Development of Advanced Computing (C-DAC) is the premier R&D organization of the Department of Electronics and Information Technology (Deity), Ministry of Communications & Information Technology (MCIT) for carrying out R&D in IT, Electronics and associated areas. Different areas of C-DAC, had originated at different times, many of which came out as a result of identification of opportunities. While C-DAC was being setup for the indigenous design, development and delivery of the supercomputing technologies for the country, the mandate given was to not only develop the supercomputing technologies in the shortest possible time, but also continue to develop the high quality human resource, which will continue to develop such advanced technologies. C-DAC's Advanced Computing Training School (ACTS) is dedicated to creating high quality manpower for C-DAC in particular and the IT industry in general through the designing and delivering various courses. The projects are offered through a network of Authorized Training Centers (ATC's) as well as C-DAC's own centers.

NETCOM has its own Software Development Company “Net PARAM Technologies Pvt. Ltd.” which is completely involved in IT products and service delivery. NTPL develops world class Websites, Android Apps, iPhone Apps, Cloud Based Application, Hybrid Mobile Apps & Big Data and Analysis. It works for the government and private projects and here the resources are placed from the top of the world. It is one of the most comprehensive and global leaders in IT services that helps its clients to simplify, strengthen and transform their businesses. NETCOM is equipped with the latest infrastructure which provides a congenial and a healthy learning environment to the students. In Netcom, Experts and Developers mentors the students and develops the professional quality in them.



### **About NET PARAM Technologies Pvt. Ltd.**

Innovation and proper aim are key factors in the growth of technology for any ICT solutions provider companies. Here at NETPARAM Technologies Private Limited encourage innovation throughout the organization and emphasizes strategic planning in starting and developing operations.

We start with the data. We conduct independent research and draw on the latest technology to develop new insights and recommendations. Our rigorous analysis identifies risks, unveils opportunities, and informs smart strategies. We focus our efforts on influential and emerging economies where the future of sustainability will be determined.

We use our research to influence government policies, business strategies, and civil society action. We test projects with communities, companies, and government agencies to build a strong evidence base. Then, we work with partners to deliver change on the ground that alleviates poverty and strengthens society. We hold ourselves accountable to ensure our outcomes will be bold and enduring.

We don't think small. Once tested, we work with partners to adopt and expand our efforts regionally and globally. We engage with decision-makers to carry out our ideas and elevate our impact. We also measure success through human well-being that improves people's lives and sustains a healthy environment.

### **Vision:**

We envision to be a recognized platform to create a world where the actions of the government, business, and communities combine to eliminate chaotic ICT solutions.

## **Abstract**

Chatbots, or conversational interfaces as they are also known, present a new way for individuals to interact with computer systems. Traditionally, to get a question answered by a software program involved using a search engine, or filling out a form. A chatbot allows a user to simply ask questions in the same manner that they would address a human.

The technology at the core of the rise of the chatbot is natural language processing (“NLP”). Recent advances in machine learning have greatly improved the accuracy and effectiveness of natural language processing, making chatbots a viable option for many organizations. This improvement in NLP is firing a great deal of additional research which should lead to continued improvement in the effectiveness of chatbots in the years to come.

This chatbot is created by loading an FAQ (frequently asked questions) into chatbot software. The functionality of the chatbot can be improved by integrating it into the organization’s enterprise software, allowing more personal questions to be answered, for example this chatbot contents like “What is CDAC?”, or “tell me about foundations and programs?”.

## TABLE OF CONTENT

Chapter No.	Topic	Page No.
	<b>Certificate</b>	2
	<b>Acknowledgement</b>	3
	<b>Company Profile</b>	4-5
	<b>Abstract</b>	6
<b>1</b>	<b>Introduction</b>	
	1.1 Overview	9
	1.2 ChatBot and Machine learning	10
	1.3 Artificial Intelligence	11-13
	1.4 AI Applications	13
<b>2</b>	<b>Background and Literature Review</b>	
	2.1 Literature Review	14
	2.2 Natural Language Processing	14
	2.3 Machine Learning	15
<b>3</b>	<b>Problem Identification</b>	<b>16</b>
<b>4</b>	<b>Requirements and Specification</b>	
	4.1 Hardware Requirement	17
	4.2 Software Requirement	17
<b>5</b>	<b>Proposed Solution</b>	
	5.1 .ipynb Scripting	18
	5.2 Importing Libraries	19
	5.3 Tokens	20
	5.4 Creating Response Function	21
	5.5 File Path	22
<b>6</b>	<b>Implementation</b>	<b>23</b>
<b>7</b>	<b>Advantages and Disadvantages</b>	
	7.1 Advantages	24-25
	7.2 Disadvantages	25-26
<b>8</b>	<b>Technology</b>	
	8.1 About Python	27-29
<b>9</b>	<b>Conclusion and Scope for Future Research</b>	<b>31</b>

## TABLE OF FIGURES AND TABLES

<b>Name of Fig and Tab</b>	<b>PAGE</b>
Fig 1.1 NLP Process	10
Fig 2.1(a) Classification	15
Fig 2.1(b) Regressing	15
Fig 5.1 .ipynb Scripting	18
Fig 5.2 Importing Libraries	19
Fig 5.3 Tokens	20
Fig 5.4 Response Function	21
Fig 5.5 File Path	22
Fig 6.1 Output of main file	23
Fig 9.1 ChatBot	31
Tab 2.1 Literature Survey	14
Tab 4.1 HardWare Requirement	17
Tab 4.2 SoftWare Requirement	17



## Chapter 1

### Introduction

#### 1.1 Overview

A **chatbot** is a software application used to conduct an on-line chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent. Designed to convincingly simulate the way a human would behave as a conversational partner, chatbot systems typically require continuous tuning and testing, and many in production remain unable to adequately converse or pass the industry standard Turing test. The term "ChatterBot" was originally coined by Michael Mauldin (creator of the first Verbot) in 1994 to describe these conversational programs.

**Chatbots** are typically used in dialog systems for various purposes including customer service, request routing, or for information gathering. While some chatbot applications use extensive word-classification processes, Natural Language processors, and sophisticated [AI](#), others simply scan for general keywords and generate responses using common phrases obtained from an associated library or database.

In 1950, Alan Turing's famous article "Computing Machinery and Intelligence" was published, which proposed what is now called the Turing test as a criterion of intelligence. This criterion depends on the ability of a computer program to impersonate a human in a real-time written conversation with a human judge to the extent that the judge is unable to distinguish reliably—on the basis of the conversational content alone—between the program and a real human. The notoriety of Turing's proposed test stimulated great interest in Joseph Weizenbaum's program ELIZA, published in 1966, which seemed to be able to fool users into believing that they were conversing with a real human.

The most well known chatbots currently are voice chatbots: **Alexa** and **Siri**. However, **chatbots** are currently being adopted at a high rate on computer chat platforms.

## 1.2 ChatBot and Machine Learning

Interacting with the machine via natural language is one of the requirements for general artificial intelligence. This field of AI is called dialogue systems, spoken dialogue systems, or chatbots. The machine needs to provide you with an informative answer, maintain the context of the dialogue, and be indistinguishable from the human (ideally).

The former help people to solve everyday problems using natural language, while the latter attempt to talk with people on a wide range of topics.

General conversation models can be simply divided into two major types — generative and selective (or ranking) models. Also, hybrid models are possible. But the common thing is that such models conceive several sentences of dialogue context and predict the answer for this context. In the picture below, you can see the illustration of such systems.

In the modern “stylish” machine learning for chatbots, recognizing the type of question is called **intent recognition**, conducting a dialog according to a script or dialog tree is called **dialog management**, and assigning parameters from user cues (for example, names, tariff names, mobile number, etc.) is called **slot filling**.

We implement all of these stages within our projects using the rule-based approach, or the “correct” approach: bots are taught by possible user questions using templates that have quantifiers in the description language of dialogs, similar to using regular expressions.



**Fig 1.1 NLP Process**

## 1.3 Artificial Intelligence

People viewed computers as machines which can perform mathematical operations at a much quicker rate than humans. They were initially viewed as computational machines much like glorified calculators. Early scientists felt that computers could never simulate the human brain. Then, scientists, researchers and (probably most importantly) science fiction authors started asking “or could it?” The biggest obstacle to solving this problem came down to one major issue: the human mind can do things that scientists couldn’t understand, much less approximate. For example, how would we write algorithms for these tasks:

1. A song comes on the radio, most listeners of music can quickly identify the genre, maybe the artist, and probably the song.
2. An art critic sees a painting he’s never seen before, yet he could most likely identify the era, the medium, and probably the artist.
3. A baby can recognize her mom’s face at a very early age.

The simple answer is that you can’t write algorithms for these. Algorithms use mathematics. Humans that accomplish these tasks couldn’t explain mathematically how they drew these conclusions. They were able to achieve these results because they **learned** to do these things over time. Artificial Intelligence and Machine Learning were designed to simulate human learning on a computer.

The terms Artificial Intelligence(AI) and Machine Learning(ML) have been used since the 1950s. At that time, it was viewed as a futuristic, theoretical part of computer science. Now, due to increases in computing capacity and extensive research into Algorithms, AI is now a viable reality. So much so, that many products we use every day have some variation AI built into them (Siri, Alexa, Snapchat facial filters, background noise filtration for phones/headphones, etc...).

Simply put, AI means to program a machine to behave like a human. In the beginning, researchers developed algorithms to try to approximate human intuition. A way to view this code would be as a huge if/else statement to determine the answer. For example, here’s some pseudocode for a chatbot from this era:

```
if(input.contains('hello'))
    response='how are you'
elif (input.contains('problem'))
    response = 'how can we help with your problem?')
...
print(response)
```

## **Types of artificial intelligence-**

AI can be categorized in any number of ways, but here are two examples.

AI can be categorized in any number of ways, but here are two examples. The first classifies AI systems as either weak AI or strong AI. Weak AI, also known as narrow AI, is an AI system that is designed and trained for a particular task. Virtual personal assistants, such as Apple's Siri, are a form of weak AI. Strong AI, also known as artificial general intelligence, is an AI system with generalized human cognitive abilities so that when presented with an unfamiliar task, it has enough intelligence to find a solution. The Turing Test, developed by mathematician Alan Turing in 1950, is a method used to determine if a computer can actually think like a human, although the method is controversial. The second example is from Arend Hintze, an assistant professor of integrative biology and computer science and engineering at Michigan State University. He categorizes AI into four types, from the kind of AI systems that exist today to sentient systems, which do not yet exist. His categories are as follows:

### **Type 1: Reactive machines.**

An example is Deep Blue, the IBM chess program that beat Garry Kasparov in the 1990s. Deep Blue can identify pieces on the chess board and make predictions, but it has no memory and cannot use past experiences to inform future ones. It analyzes possible moves -- its own and its opponent -- and chooses the most strategic move. Deep Blue and Google's AlphaGO were designed for narrow purposes and cannot easily be applied to another situation.

### **Type 2: Limited memory.**

These AI systems can use past experiences to inform future decisions. Some of the decision-making functions in autonomous vehicles have been designed this way. Observations used to inform actions happening in the not-so-distant future, such as a car that has changed lanes. These Observations are not stored permanently.

### **Type 3: Theory of mind.**

This is a psychology term. It refers to the understanding that others have their own beliefs, desires and intentions that impact the decisions they make. This kind of AI does not yet exist.

### **Type 4: Self-awareness. In this category.**

AI systems have a sense of self, have consciousness. Machines with self-awareness understand their current state and can use the information to infer what others are feeling. This type of AI does not yet exist.

## **Examples of AI technology:**

Automation is the process of making a system or process function automatically. Robotic process automation, for example, can be programmed to perform high-volume, repeatable tasks normally performed by humans. RPA is different from IT automation in that it can adapt to changing circumstances.

Machine learning is the science of getting a computer to act without programming. Deep Learning is a subset of machine learning that, in very simple terms, can be thought of as the automation of predictive analytics. There are three types of machine learning algorithms: supervised learning, in which data sets are labeled so that patterns can be detected and used to label new data sets; unsupervised learning, in which data sets aren't labeled and are sorted according to similarities or differences; and reinforcement learning, in which data sets aren't labeled but, after performing an action or several actions, the AI system is given feedback. Machine vision is the science of making computers see. Machine vision captures and analyzes visual information using a camera, analog-to-digital conversion and digital signal processing. It is often compared to human eyesight, but machine vision isn't bound by biology and can be programmed to see through walls, for example. It is used in a range of applications from

signature identification to medical image analysis. Computer vision, which is focused on machine-based image processing, is often conflated with machine vision.

Natural language processing (NLP) is the processing of human -- and not computer -- language by a computer program. One of the older and best known examples of NLP is spam detection, which looks at the subject line and the text of an email and decides if it's junk. Current Approaches to NLP are based on machine learning. NLP tasks include text translation, sentiment analysis and speech recognition.

Pattern recognition is a branch of machine learning that focuses on identifying patterns in data. The term, today, is dated. Robotics is a field of engineering focused on the design and manufacturing of robots. Robots are often used to perform tasks that are difficult for humans to perform or perform consistently. They Are used in assembly lines for car production or by NASA to move large objects in space. More recently, researchers are using machine learning to build robots that can interact in social settings.

## **1.4 AI Applications**

AI in healthcare. The biggest bets are on improving patient outcomes and reducing costs. Companies are applying machine learning to make better and faster diagnoses than humans. One Of the best known healthcare technologies is IBM Watson. It understands natural language and is capable of responding to questions asked of it. The system mines patient data and other available data sources to form a hypothesis, which it then presents with a confidence scoring schema. Other AI applications include chatbots, a computer program used online to answer questions and assist customers, to help schedule follow-up appointments or aiding patients through the billing process, and virtual health assistants that provide basic medical feedback.

AI in business. Robotic process automation is being applied to highly repetitive tasks normally performed by humans. Machine learning algorithms are being integrated into analytics and CRM platforms to uncover information on how to better serve customers. Chatbots have been incorporated into websites to provide immediate service to customers. Automation of job positions has also become a talking point among academics and IT consultancies such as Gartner and Forrester.

AI in education. AI can automate grading, giving educators more time. AI can assess students and adapt to their needs, helping them work at their own pace. AI tutors can provide additional support to students, ensuring they stay on track. AI could change where and how students learn, perhaps even replacing some teachers. AI in finance. AI applied to personal finance applications, such as Mint or TurboTax, impending financial institutions. Applications such as these could collect personal data and provide financial advice. Other programs, IBM Watson being one, have been applied to the process of buying a home. Today, software performs much of the trading on Wall Street. AI in law. The discovery process, sifting through of documents, in law is often overwhelming for humans. Automating this process is a better use of time and a more efficient process. Startups are also building question-and-answer computer assistants that can sift programmed-to-answer questions by examining the taxonomy and ontology associated with a database. AI in manufacturing. This is an area that has been at the forefront of incorporating robots into the workflow. Industrial robots used to perform single tasks and were separated from human workers, but as the technology advanced that changed.

## Background and Literature Review

## 2.1 Literature Review

Title	Author	Publication year	Findings
Chabot's meet eHealth: automatizing healthcare	Flora Amato, Stefano Marrone, Vincenzo Moscato, Gabriele Piantadosi, Antonio Picariello, and Carlo Sansone	2017	In this recommendation system has been The recent advances of technologies for data processing and analytics have radically changed the healthcare giving rise to digital healthcare solutions, promising to transform the whole healthcare process to become more efficient, less expensive and higher quality.
Text-based Healthcare Chatbots Supporting Patient and Health Professional Teams	University of St. Gallen, Institute of Technology Management, St.Gallen, Switzerland.	2016	Technology-based self-service channels [41] and digital health interventions [1, 31] have the potential to support patients in their everyday life and health professionals likewise. Although there are scalable self-service channels in the form of digital voice

Tab 2.1 Literature Survey

## 2.2 Natural Language Processing

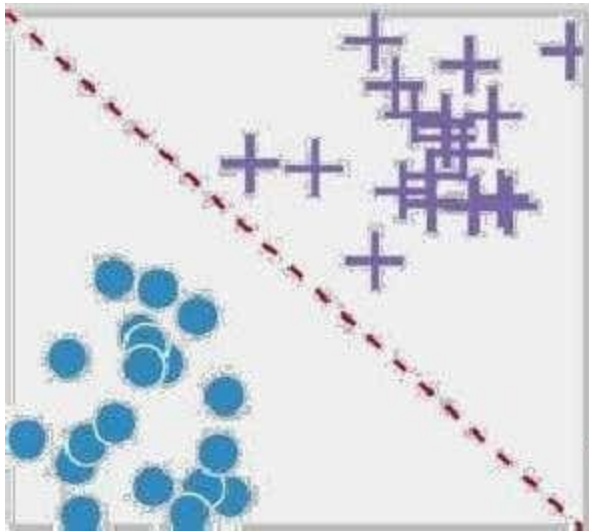
Natural Language Processing (NLP) is the study of letting computers understand human languages. Without NLP, human language sentences are just a series of meaningless symbols to computers. Computers don't recognize the words and don't understand the grammars. NLP can be regarded as a "translator", who will translate human languages to computer understandable information.

Traditionally, users need to follow well-defined procedures accurately, in order to interact with computers. For example, in Linux systems, all commands must be precise. A single replacement of one character or even a space can have significant differences. However, the emergence of NLP is changing the way of interacting. Apple Siri and Microsoft Cortana have made it possible to give commands in everyday languages and are changing the way of interacting.

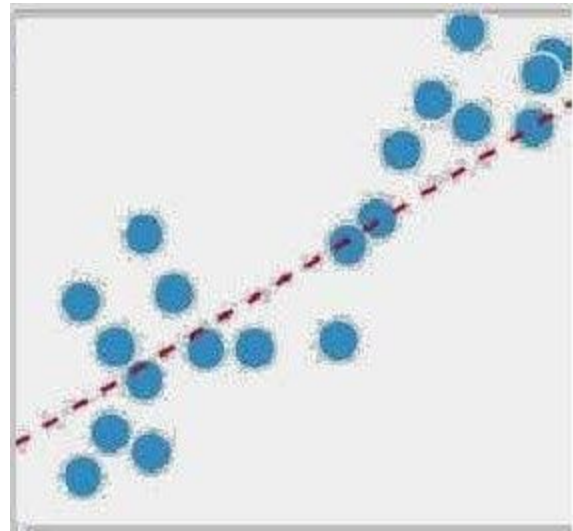
## 2.3 Machine Learning

Machine Learning (ML) is an area of computer science that "gives computers the ability to learn without being explicitly programmed". The parameter of the formulas is calculated from the data, rather than defined by the programmer. Two most common uses of ML are Classification And Regression. As shown in figure2(a) and 2(b), Classification means to categorize different types of data, while Regression means to find a way to describe the data. Basic ML program will have two stages, fitting and predicting.

In the fitting stage, the program will be given a large set (at least thousands) of data. The program will try to adjust its parameter based on some statistical models, in order to make it "fit" the input data best. In the predicting stage, the program will give prediction for a new input based on the parameters it just calculated out. For example, the famous Iris flower dataset contains the measurement of several features of three different species of flowers, such as the length of sepals and petals. A well-defined ML program can learn the pattern behind this feature and give predictions accordingly.



**Fig 2.1(a) Classification**



**Fig 2.1(b) Regressing**

#### 3.1 Problem Identification

Changing of requirements As an industrial project to build a product, we must follow the requirement from the user. We had to archive what we had done before and build a new one.

- **Lack of training data.-**

The quantity and quality of training data is critical to the performance to a machine learning model. However, for some confidential and privacy reasons, the business team cannot provide enough data for us, and we had to make up data on our own. For the machine learning model, we generate some fake data based on our daily life experience, which is really biased, although with a good accuracy on the fake data.

- **Unstable API version.**

Because the API service we are using is still under development, and we cannot fix to a version for the API, the API may change overtime. Moreover, there are inconsistencies between the APIs and their documents or sample codes.

- **Not familiar with the PHP language and .NET framework.**

I don't have knowledge of the PHP language. Programming in a new language in such a huge framework is quite challenging for us at the beginning of the project. However, when we come to the later phases, we are more used to that.



**Requirements and Specifications**

**4.1 Hardware Requirement**

Processor	Pentium IV (minimum)
HardDisk	40 GB (minimum)
RAM	256 MB (minimum)

**Tab 4.1 Hardware Requirement**

**4.2 Software Requirement**

Operating System	Windows and Linux
Technology	Python, AIML

**Tab 4.2 Software Requirement**

## 5.1 .ipynb Scripting

We created the .ipynb file that only handles one pattern, load python algorithms. When we enter the command to the bot, it will try to load a text file to answer. It won't work unless we actually create it. Here is what you can put inside the text file. We will match two basic patterns and respond.

```

In [2]: warnings.filterwarnings("ignore")

Ignore filter warnings

In [3]: GREETING INPUTS = ("hello", "hi", "greetings", "sup", "whats up","hey",)
GREETING RESPONSES = ["hi", "hey", "**nods**", "hi there", "hello", "I am glad! You are talking to me"]
LEMER = nltk.stem.WordNetLemmatizer()

Some greeting Inputs and thier responses

In [4]: def lematize_tokens(tokens):
return [LEMER.lemmatize(token) for token in tokens]

Lematization of tokens

In [5]: def lematization_normalizer(text):
return lematize_tokens(nltk.word_tokenize(text.lower().translate(remove_punct_dict)))

Normalization of tokens

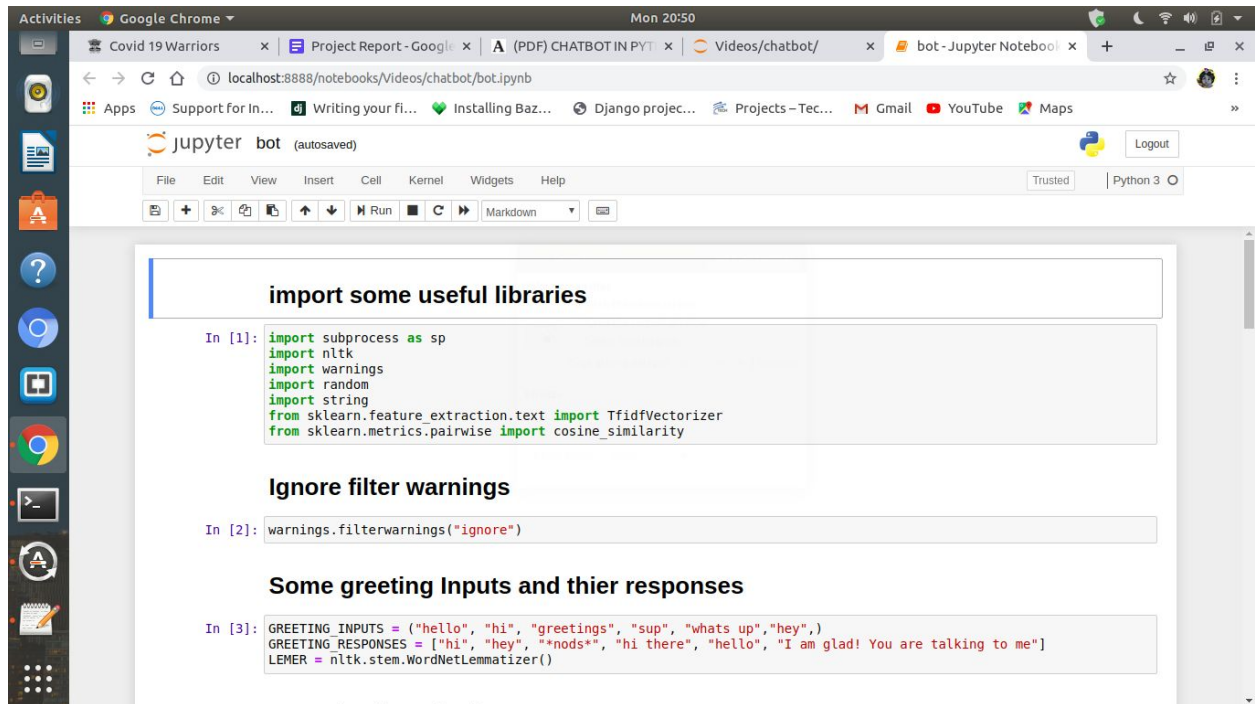
Soliting Sentences

```

Fig 5.1 .ipynb Scripting

## 5.2 Importing Libraries

To deploy machine learning and Artificial Intelligence Algorithms we need to import some functions and methods, in python there are various functions and methods in it, and they are known as **Python Libraries**. To implement these special functions we need to import some useful libraries. They are- NLTK, SkLearn, NumPy.



The screenshot shows a Jupyter Notebook titled 'bot' running in a Google Chrome browser. The notebook contains three code cells. The first cell, titled 'import some useful libraries', imports subprocess, nltk, warnings, random, string, TfidfVectorizer, and cosine\_similarity. The second cell, titled 'Ignore filter warnings', uses warnings.filterwarnings to ignore all warnings. The third cell, titled 'Some greeting Inputs and thier responses', defines GREETING\_INPUTS, GREETING\_RESPONSES, and a LEMER object using nltk.stem.WordNetLemmatizer.

```
In [1]: import subprocess as sp
import nltk
import warnings
import random
import string
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

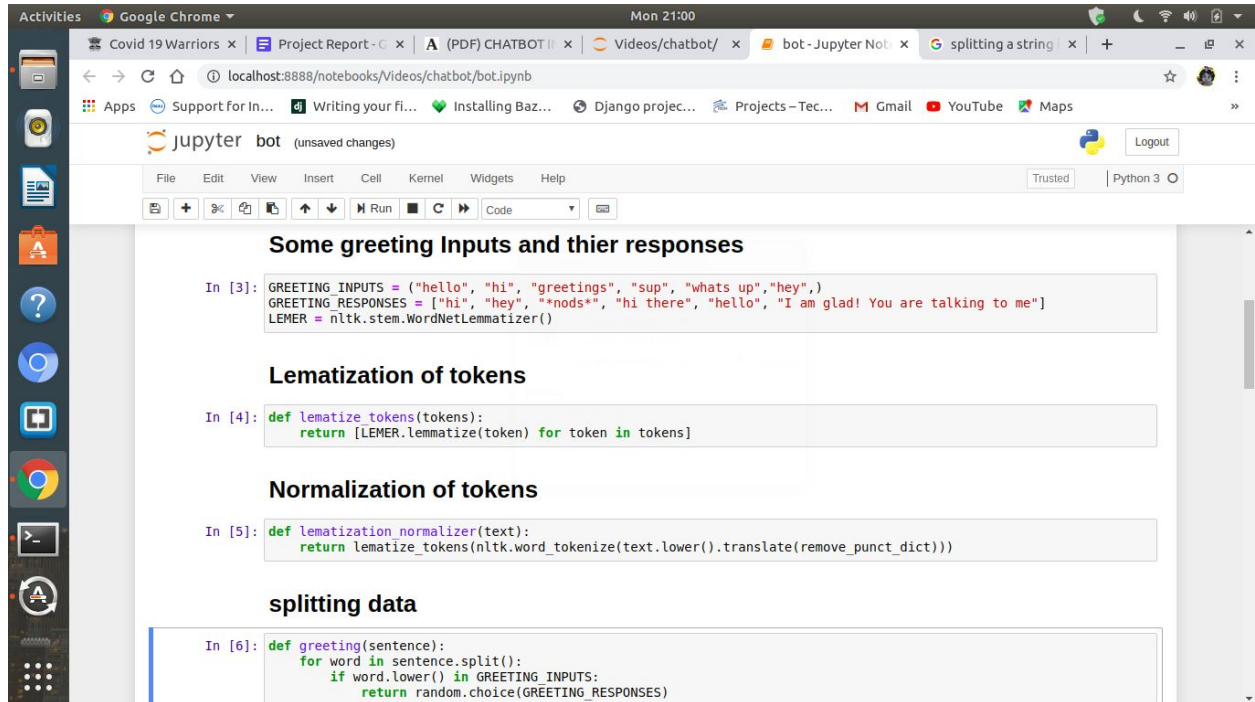
In [2]: warnings.filterwarnings("ignore")

In [3]: GREETING_INPUTS = ("hello", "hi", "greetings", "sup", "whats up", "hey",)
GREETING_RESPONSES = ["hi", "hey", "**nods**", "hi there", "hello", "I am glad! You are talking to me"]
LEMER = nltk.stem.WordNetLemmatizer()
```

Fig 5.2 Importing Libraries

## 5.3 Tokens

As we know that tokens are important for creating scripts and implementing algorithms. In the database, data can be in any form and may be in non-continuous form, so for arranging the data in a continuous form we need to split, Normalization, Lemmatization the data.



The screenshot shows a Jupyter Notebook titled 'bot' running on a local host. The notebook contains four code cells, each with a title and Python code. The first cell is titled 'Some greeting Inputs and thier responses' (note the typo 'thier'). The second cell is titled 'Lematization of tokens' (note the typo 'Lematization'). The third cell is titled 'Normalization of tokens'. The fourth cell is titled 'splitting data'. The code in the first cell defines greeting inputs and responses, and initializes a WordNetLemmatizer. The second cell defines a function to lematize tokens. The third cell defines a function to normalize tokens. The fourth cell defines a function to split data into tokens.

```
In [3]: GREETING_INPUTS = ("hello", "hi", "greetings", "sup", "whats up", "hey",)
GREETING_RESPONSES = ["hi", "hey", "**nods**", "hi there", "hello", "I am glad! You are talking to me"]
LEMER = nltk.stem.WordNetLemmatizer()

Lematization of tokens

In [4]: def lematize_tokens(tokens):
return [LEMER.lemmatize(token) for token in tokens]

Normalization of tokens

In [5]: def lematization_normalizer(text):
return lematize_tokens(nltk.word_tokenize(text.lower().translate(remove_punct_dict)))

splitting data

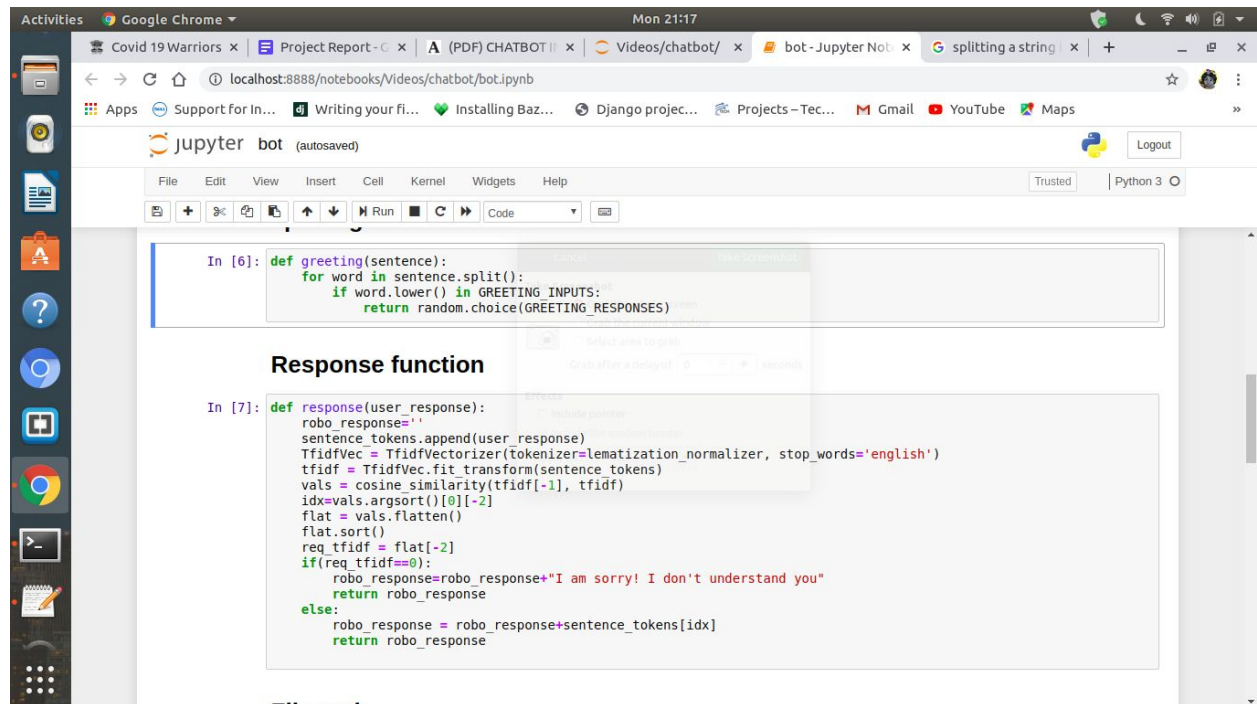
In [6]: def greeting(sentence):
for word in sentence.split():
if word.lower() in GREETING_INPUTS:
return random.choice(GREETING_RESPONSES)
```

Fig 5.3 Tokens

## 5.4 Creating Response Function

Here we have created a Response function. In this function we have given user\_response as an argument.

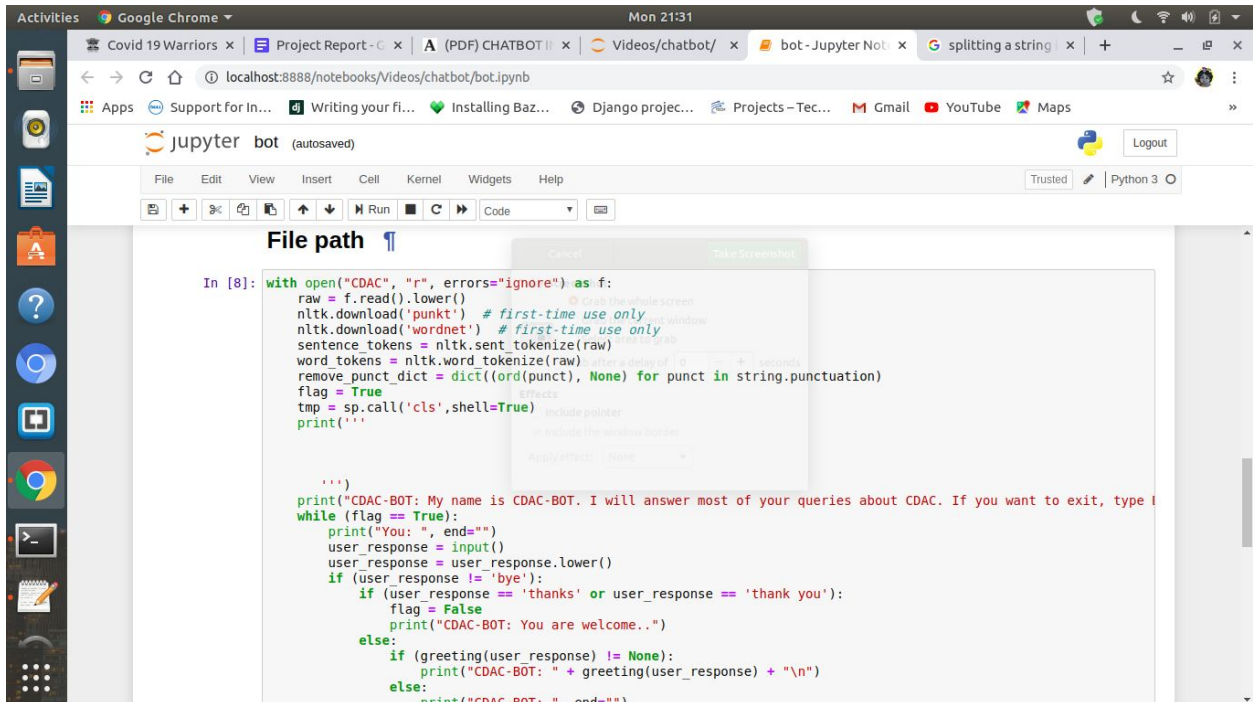
```
def response(user_response):
```



**Fig 5.4 Response Function**

## 5.5 File Path

We need a dataset to connect the chatbot, so the chatbot can respond to the user. All of the responses to be given to the user of the chatbot need some useful data, here we give the path to the program.



```
In [8]: with open("CDAC", "r", errors="ignore") as f:
    raw = f.read().lower()
    nltk.download('punkt') # first-time use only
    nltk.download('wordnet') # first-time use only
    sentence_tokens = nltk.sent_tokenize(raw)
    word_tokens = nltk.word_tokenize(raw)
    remove_punct_dict = dict((ord(punct), None) for punct in string.punctuation)
    flag = True
    tmp = sp.call('cls', shell=True)
    print('')

    ...

print("CDAC-BOT: My name is CDAC-BOT. I will answer most of your queries about CDAC. If you want to exit, type !")
while (flag == True):
    print("You: ", end="")
    user_response = input()
    user_response = user_response.lower()
    if (user_response != 'bye'):
        if (user_response == 'thanks' or user_response == 'thank you'):
            flag = False
            print("CDAC-BOT: You are welcome..")
        else:
            if (greeting(user_response) != None):
                print("CDAC-BOT: " + greeting(user_response) + "\n")
            else:
                print("CDAC-BOT: ". end="")
```

Fig 5.5 File Path

### Implementation

This section covers the design and implementation of different modules of the bot, which contains the design of the PYTHON module, the Translator API and the AIM module..

```
CDAC-BOT: My name is CDAC-BOT. I will answer most of your queries about CDAC. If you want to exit, type Bye!  
You: hello  
CDAC-BOT: I am glad! You are talking to me  
  
You: what is cdac  
CDAC-BOT: c-dac atc centre for development of advanced computing (c-dac) is the premier r&d organization of the de  
partment of electronics and information technology (deity), ministry of communications & information technology (m  
cit) for carrying out r&d in it, electronics and associated areas.  
  
You: tell me about foundation and programmes  
CDAC-BOT: foundation:  
founded by its parent company cdac pune, ministry of electronics and information technology, of the government of  
india in 1988.  
  
programms:  
cdac jaipur offers study programmes in it and electronics like- data science, artificial intelligence and computer  
vision, machine learning with nlp, software testing, python, java, php, bigdata analysis, iot with industrial appl  
ication, c_c++_dsa, reactjs, angular js with ionic, salesforce with crm, cyber security, ios development, android  
development.  
  
You: thanks  
CDAC-BOT: You are welcome..
```

---

**Fig 6.1 Output of main file**

### Advantages and Disadvantages

#### 7.1 Advantages

##### 1. Accessible anytime:

I'm sure most of you are always kept on hold while operators connect you to a customer care executive. On an average people spend around 7 minutes until they are assigned to a person. Gone are the frustrating days of waiting in a queue for the next available operative. They are replacing live chat and other forms of slower contact methods such as emails and phone calls. Since chatbots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break. This improves your customer UX and helps you rank highly in your sector. Another advantage of this instant response is that you can also skillfully craft your chatbot to maintain your image and brand.

##### 2. Handling Capacity:

Unlike humans who can only communicate with one human at a time, chatbots can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered immediately. Imagine you own a restaurant, and you have a good reputation for your food of which most of your revenues come from delivery. As the demand keeps rising, you will have more customers to take orders from but very few staff to attend them all. Having a chatbot would eliminate such problems and cater to each and every person and ensure that no order is missed. Companies like Taco Bell and Dominos are already using chatbots to arrange delivery of parcels.

##### 3. Flexible attribute:

Chatbots have the benefit that it can quite easily be used in any industry. Unlike other products where you have to do a lot of development and testing to change platforms, chatbots are relatively easy to switch. One has to just train the bot by giving the right conversation structure and flow to switch its current field or industry.

Or if there is a lot of back and forth between two sections of the industry, say customer support and sales, then you could have custom built presets which would already have the conversation flow and structure to carry out the interactions with the user.

##### 4. Customer Satisfaction:

Humans are bound to change their emotions. Chatbots, on the other hand, are bound by some rules and obey them as long as they're programmed to. They will always treat a customer in the perfect way no matter how rough the person is or how foul language the person uses. Not everyone orders the same food everyday, people's choices may change everyday. In this case, it can use your order history to make suggestions for the next order, learn your address details and much more. Customers love this smooth interaction and want all their transactions to be as simple as possible.



## **5. Cost Effective:**

Hiring a human for a job is never a cheap affair, and it will be expensive if your revenue are not high or sales targets are not met and would create havoc in the business. Due to the boundaries of human beings, a single human can only handle one or two people at the sametime. More than that would be extremely tough for the employee.Chatbots could help solve this age-old problem. As one chatbot is equal to loads of employees, it can easily communicate with thousands of customers at the same time. We Would only need a handful of people to jump into conversations sometimes when necessary.Hence, it would drastically bring down the expenses and bring about a steep rise in revenue and customer satisfaction.

## **6. Faster Onboarding:**

Before you want to accomplish a task, you first must learn how to work on the task and complete it. Only then will they be considered fit for the job. There is a continuous teaching involved in every level of hierarchy the employee will go through. Also, there will be a lot of change in the employees, some stay, some get fired, some more join in etc.What we want to say is, employees will change; it's a fact. And this would require you to allot a lot of time for your employees into grooming the new joiners. Chatbots could eliminate that time to almost zero, but provide a very clean and easy to understand conversation flow and structure that needs to be maintained by the chatbot. No doubt there will be changes in this too, but it will rather take a fraction of your time to resolve as compared to human employees.

## **7.2 Disadvantages**

### **1.Too many functions:**

Most developers strive to create a universal chatbot that will become a fully-fledged assistant to users. But in practice functional bots turn out not to cope with the majority of queries. They often do not understand the user, forget what they were told 5 minutes earlier,and have many other disadvantages. And that is no wonder, as the development of a universal bot, which would understand natural language and could evaluate context, takes years of hard work for a team of experienced programmers. And even in this case, such programs should be constantly improved while in service.However, modern technologies allow building rather useful bots to perform specific actions such as booking train tickets or providing support to bank customers.

### **2.Primitive algorithms:**

There are two types of bots:

based on artificial intelligence, being able to learn in the process of communication; programmed for specific behavior scenarios.

Artificial intelligence chatbots are considered to be better, as they can respond depending on the situation and context. But the development of complex algorithms is required for this purpose. Meanwhile, only IT giants and few developers possess such a powerful technological base.

Therefore, it would be better for ordinary companies to focus on the second variant of bots, as they are more reliable and simpler. Namely for the reason they do not possess intelligence, they will not be able to adopt rude communication patterns and get beyond the control of creators.

### **3.Complex interface:**

Talking to a bot implies talking in a chat, meaning that a user will have to write a lot. And in case a bot cannot understand the user's request, he will have to write even more. It takes time to find out which commands a bot can respond to correctly, and which questions are better to avoid. Thus, talking to a chatbot does not save time in the majority of cases.

Perhaps the efficiency of virtual assistants will increase due to the implementation of voice recognition function in the future. But for the time being their functional capabilities are very restricted, and they can be truly useful only in a few business areas.

#### 8.1 About Python:

Dating from 1991, Python is a relatively new programming language. From the start, Python was considered a gap-filler, a way to write scripts that “automate the boring stuff” (as one popular book on learning Python put it) or to rapidly prototype applications that will be implemented in one or more other languages. However, over the past few years, Python has emerged as a first-class citizen in modern software development, infrastructure management, and data analysis. It is no longer a back-room utility language, but a major force in web application development and systems management and a key driver behind the explosion in big data analytics and machine learning. Perfect for IT, Python simplifies many kinds of work, from system automation to working in cutting-edge fields like machine learning.

Python is easy to learn. The number of features in the language itself is modest, requiring relatively little investment of time or effort to produce one’s first programs. Python syntax is designed to be readable and straightforward. This simplicity makes Python an ideal teaching language, and allows newcomers to pick it up quickly. Developers spend more time thinking about the problem they’re trying to solve, and less time thinking about language complexities or deciphering code left by others.

Python is broadly used and supported. Python is both popular and widely used, as the high rankings in surveys like the Tiobe Index and the large number of GitHub projects using Python attest. Python runs on every major operating system and platform, and most minor ones too. Many major libraries and API-powered services have Python bindings or wrappers, allowing Python to interface freely with those services or make direct use of those libraries. Python may not be the fastest language, but what it lacks in speed, it makes up for versatility.

Python is not a “toy” language. Even though scripting and automation cover a large chunk of Python’s use cases (more on that below), Python is also used to build robust, professional-quality software, both as standalone applications and as web services.

#### What is Python used for?

The most basic use case for Python is as a scripting and automation language. Python isn’t just a replacement for shell scripts or batch files, but is also used to automate interaction with web browsers or application GUIs or system provisioning and configuration in tools such as Ansible and Salt. But scripting and automation represent only the tip of the iceberg with Python.

- Python is used for general application programming. Both CLI and cross-platform GUI applications can be created with Python and deployed as self-contained executables. Python doesn’t have the native ability to generate a standalone binary from a script, but third-party packages like `cx_Freeze` or `PyInstaller` can be used to accomplish that.

- Python is used for data science and machine learning. Sophisticated data analysis has become one of fastest moving areas of IT and one of Python's star use cases. The vast majority of the libraries used for data science or machine learning have Python Interfaces, making the language the most popular high-level command interface for machine learning libraries and other numerical algorithms.
- Python is used for web services and RESTful APIs. Python's native libraries and third-party web frameworks provide fast and convenient ways to create everything from simple REST APIs in a few lines of code, to full-blown, data-driven sites. Python's latest versions have powerful support for asynchronous operations, allowing sites to handle up to tens of thousands of requests per second with the right libraries.
- Python is used for metaprogramming. In Python, everything in the language is an object, including Python modules and libraries themselves. This allows Python to work as a highly efficient code generator, making it possible to write applications that manipulate their own functions and have the kind of extensibility that would be difficult or impossible to pull off in other languages.
- Python is used for glue code. Python is often described as a "glue language," meaning it can allow disparate code (typically libraries with C language interfaces) to interoperate. Its use in data science and machine learning is in this vein, but that's just one incarnation of the general idea.

Also worth noting are the sorts of tasks Python is not well-suited for. Python is a high-level language, so it's not suitable for system-level programming— device drivers or OS kernels are straight out. It's also not ideal for situations that call for cross-platform standalone binaries. You could build a standalone Python app for Windows, Mac, and Linux, but not elegantly or simply. Finally, Python is not the best choice when speed is an absolute priority in every aspect of the application. For that you're better off with C/C++ or another language of that caliber.

## **The Python language's pros and cons:**

Python syntax is meant to be readable and clean, with little pretense. A standard "helloworld" in Python 3.x is nothing more than:

- `print("Hello world!")`
- Python provides many syntactical elements that make it possible to concisely express many common program flows. Consider a sample program for reading lines from text file into a list object, stripping each line of its terminating newline character along the way:
- `with open('myfile.txt') as my_file:`
- `file_lines = [x.strip('\n') for x in my_file]`
- The `with/as` construction is a "context manager," which provides an efficient way to instantiate a given object for a block of code and then dispose of it outside of that block. In this case, the object in question is

`my_file`, instantiated with the `open()` function. This takes the place of several lines of boilerplate to open the file, read individual lines from it, then close it up.

- The `[x ... for x in my_file]` construction is another Python idiosyncrasy, the “list comprehension.” It allows a given item that contains other items (here, `my_file` and the lines it contains) to be iterated through, and to allow each iterated element (that is, each `x`) to be processed and automatically appended into a list.
- You could write such a thing as a formal `for... loop` in Python, much as you would in another language. The point is that Python has a way to economically express things like loops that iterate over multiple objects and perform some simple operation on each element in the loop, or work with things that require explicit instantiation and disposal. Constructions like this allow Python developers to balance terseness and readability.
- Python’s other language features are meant to complement common use cases. Most Modern object types—Unicode strings, for instance—are built directly into the language. Data structures—like lists, dictionaries (i.e., hashmaps), tuples (for storing immutable collections of objects), and sets (for storing collections of unique objects)—are available as standard-issue items.
- Like C#, Java, and Go, Python has garbage-collected memory management, meaning the programmer doesn’t have to implement code to track and release objects. Normally garbage collection happens automatically in the background, but if that poses a performance problem, it can be triggered manually or disabled entirely.
- An important aspect of Python is its dynamism. Everything in the language, including functions and modules themselves, are handled as objects. This comes at the expense of speed (more on that below), but makes it far easier to write high-level code. Developers can perform complex object manipulations with only a few instructions, and even treat parts of an application as abstractions that can be altered if needed.
- Python’s use of significant whitespace has been cited as both one of Python’s best and worst attributes. The indentation on the second line shown above isn’t just for readability; it is part of Python’s syntax. Python interpreters will reject programs that don’t use proper indentation to indicate control flow.
- Syntactical white space might cause noses to wrinkle, and some people do reject Python out of hand for this reason. But strict indentation rules are far less obtrusive in practice than they might seem in theory, even with the most minimal of code editors, and the end result is code that is cleaner and more readable.

## Python 2 versus Python 3:

- Python is available in two versions, which are different enough to trip up many new users. Python 2.x, the older “legacy” branch, will continue to be supported (i.e. receive official updates) through 2020, and it might even persist unofficially after that. Python 3.x, the current and future incarnation of the language, has many useful and important features not found in 2.x, such as better concurrency controls and a more efficient interpreter.
- Python 3 adoption was slowed for the longest time by the relative lack of third-party library support. Many Python libraries supported only Python 2, making it difficult to switch. But over the last couple of years, the number of libraries supporting only Python 2 has dwindled; most are now compatible with both versions. Today, there are few reasons against using Python 3.

### Conclusion and Future Scope

#### 9.1 Conclusions

In the fast-growing world of AI, consumers are getting technological help in all facets of their lives. The internet provides various ways to get information and has radically changed the way we communicate.

Innovation has enhanced our lives with more opportunities, and everything is quite simple for us. Everybody likes to collaborate and expect quick answers without much delay. You can use online networking platforms or websites regularly for various reasons to connect with others.

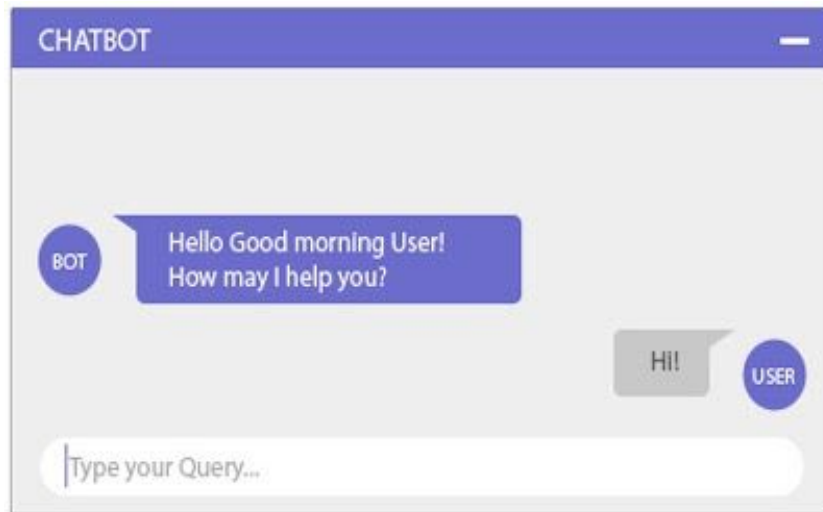
A chatbot is a program or service that easily connects with you to help solve your queries. The services that a chatbot can deliver are quite diverse, from providing important life-saving health messages to checking the weather forecast to purchasing a new pair of shoes. While interacting with a chatbot, you should feel as if you are talking with a real person.

A chatbot is a service, an intuitive operator, and a conversational space rooted in AI. It chats with clients/customers through sound or content, especially over the Internet.

Chatbots are outlined with a predesigned dialog box based on a natural language processing system. When a customer asks a question, the bot reacts quickly. The customer feels as if they are speaking with a human.

The bot chat logs page enables you to include new answers, greetings, phrases, and predefined reactions. You can see the discussions you've had with the bot and accordingly correct its responses to make business strategy by analyzing the chat logs.

The live chat facility implies that you're speaking with a human, though bots are completely robotized.



**Fig 9.1 ChatBot**

## **9.2 Future Scope**

There are limitations to what has been currently achieved with chatbots. The limitations of data processing and retrieval are hindering chatbots to reach their full potential. It is not that we lack the computational processing power to do so. However, there is a limitation on

“How” we do it. One of the biggest examples is the retail customer market. Retail customers are primarily interested in interacting with humans because of the nature of their needs. They don’t want bots to process their needs and respond accordingly.

## References

- <https://chatbotslife.com/what-you-need-to-know-about-chatbot-development-4900e9fbf702>
- <https://towardsdatascience.com/how-to-create-a-chatbot-with-python-deep-learning-in-less-than-an-hour-56a063bdfc44>
- <https://www.edureka.co/blog/how-to-make-a-chatbot-in-python/>



## **Appendix 1: Live Project of CDAC for Indian Air Force**

**Title: STAR EXAM MARCH 2020**



**CDAC ATC NETCOM JAIPUR** appointed me as **Exam Server Manager/Chief Invigilator (CI)** for the **STAR Exam** and my duties were to manage Exam Servers and Testing Nodes as well as coordination with other technical & operational staff at the center.

**Status:** - currently working

With this live project, I basically learned about various things like:

1. To Deploy Exam Server.
2. To manage every problem encountered during the exam duration.
3. To manage problems faced due to server failure cases.
4. To recover the server from failure case scenario.
5. Learned thoroughly about each and every prerequisite learnings which were needed for deploying the exam server with hands-on practices.

--Thank You--











