Done By: Eng. Mohammed Marwan Shahin
Technical Training Specialist
Technology Academy

Outlines

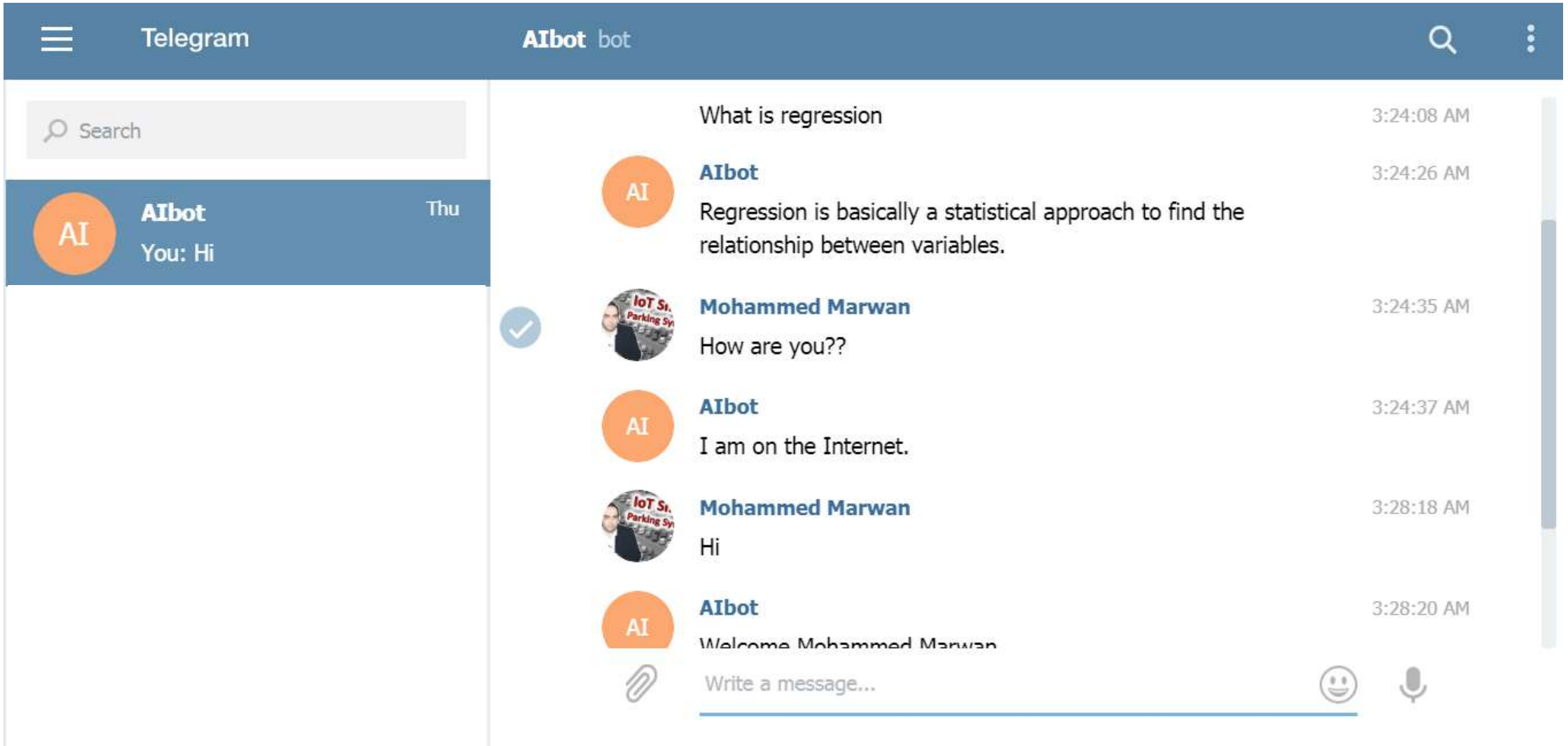
Day 1:

- Introducing Chabot
- Anaconda Installation & overview
- NLP Techniques
- Chatterbot Library
- Creating our 1st Chabot
- Chatterbot features (Adapters) and enhancements

Day 2:

- What is a Virtual Assistant?
- Speech Recognition for virtual assistant
- How to create virtual assistant?
- Using API to empower our Chatbot
- Deploying & Integrating a Chabot with Telegram.

What you will learn by the end of this course?



What is a Chatbot?

Chatbot : Also known as a bot, artificial agent, etc is basically software program driven by artificial intelligence which serves the purpose of making a conversation with the user by texts or by speech. Famous examples include Siri, Alexa, etc.

NLP Techniques & Tasks

Natural Language Processing

Natural Language:

- It is any language that has evolved naturally through use and repetition without conscious planning

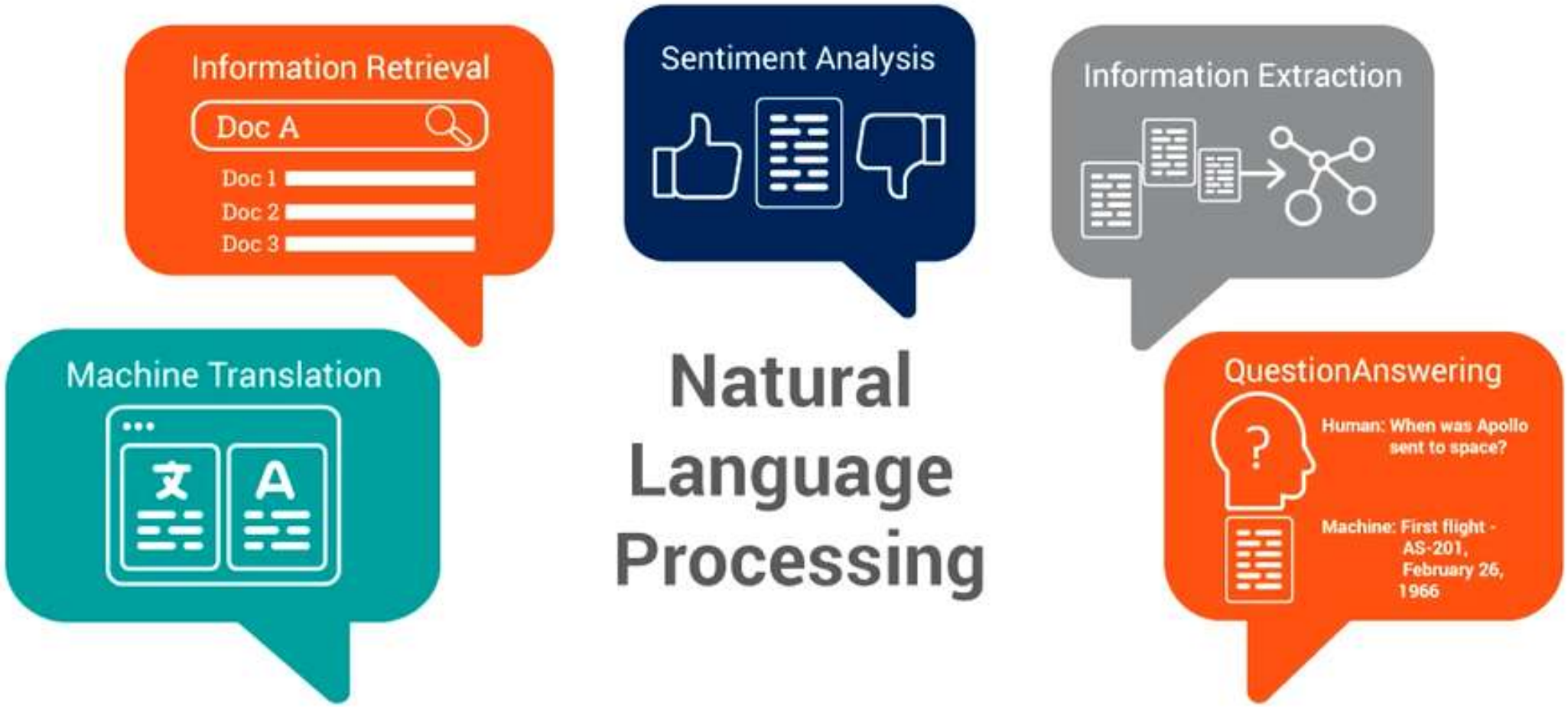
NLP:

- NLP is a field that focuses on software's ability to understand and process human languages

What is NLP?

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

NLP Subfields



NLP Techniques & Tasks

Text Example:

Voilà! In view, a humble vaudevillian veteran, cast vicariously as both victim and villain by the vicissitudes of Fate. This visage, no mere veneer of vanity, is a vestige of the vox populi, now vacant, vanished. However, this valorous visitation of a by-gone vexation, stands vivified and has vowed to vanquish these venal and virulent vermin vanguarding vice and vouchsafing the violently vicious and voracious violation of volition.The only verdict is vengeance; a vendetta, held as a votive, not in vain, for the value and veracity of such shall one day vindicate the vigilant and the virtuous. Verily, this vichyssoise of verbiage veers most verbose, so let me simply add that it's my very good honor to meet you and you may call me V.

-V for Vendetta

NLP Techniques

Cleaning:

Voilà In view a humble vaudevillian veteran cast vicariously as both victim and villain by the vicissitudes of Fate This visage no mere veneer of vanity is a vestige of the vox populi now vacant vanished However this valorous visitation of a by gone vexation stands vivified and has vowed to vanquish these venal and virulent vermin vanguarding vice and vouchsafing the violently vicious and voracious violation of volition The only verdict is vengeance a vendetta held as a votive not in vain for the value and veracity of such shall one day vindicate the vigilant and the virtuous Verily this vichyssoise of verbiage veers most verbose so let me simply add that its my very good honor to meet you and you may call me V

NLP Techniques

Tokenization:

- Process of breaking up text into smaller pieces (tokens)
- Token: Can be a word or a sentence

Voilà	In	view	a	humble	vaudevillian	veteran	cast	vicariously	as	both	victim	and	villain	by
the	vicissitudes	of	Fate	This	visage	no	mere	veneer	of	vanity	is	a	vestige	of
the	vox	populi	now	vacant	vanished	However	this	valorous	visitation	of	a	bygone	vexation	stands
vivified	and	has	vowed	to	vanquish	these	venal	and	virulent	vermin	vanguarding	vice	and	vouchsafing
the	violently	vicious	and	voracious	violation	of	volition The	only	verdict	is	vengeance	a	vendetta	held
as	a	votive	not	in	vain	for	the	value	and	veracity	of	such	shall	one
day	vindicate	the	vigilant	and	the	virtuous	Verily	this	vichyssoise	of	verbiage	veers	most	verbose
so	let	me	simply	add	that	its	my	very	good	honor	to	meet	you	and
you	may	call	me	V										

NLP Techniques

Stop words:

- Words like ‘then’, ‘before’, ‘between’, ‘the’, ‘is’, ‘are’, and so on
- Words that are filtered out before processing natural text

Voilà view humble vaudevillian veteran cast vicariously victim villain vicissitudes Fate visage mere veneer vanity vestige vox populi vacant vanished valorous visitation bygone vexation stands vivified vowed vanquish venal virulent vermin vanguarding vice vouchsafing violently vicious voracious violation volition verdict vengeance vendetta held votive vain value veracity one day vindicate vigilant virtuous Verily vichyssoise verbiage veers verbose let simply add good honor meet call V

NLP Techniques

Stemming:

Process of reducing or taking the stem or root of a word

Voilà view humble vaudevillian veteran cast **vicariously** victim villain **vicissitudes** Fate visage mere veneer vanity vestige **vox populi** vacant **vanished** **valorous** visitation bygone vexation **stands vivified vowed** vanquish venal virulent vermin **vanguarding** vice **vouchsafing violently** vicious voracious **violation** volition verdict vengeance vendetta held votive vain value veracity one day vindicate vigilant **virtuous** Verily vichyssoise verbiage veers verbose let simply add good honor meet call V

NLP Techniques

Parts of Speech Tagging:

- Process of assigning part of speech tags to tokens (words)
- Tags include noun, verb, adjective, and so on

Voilà![**interjection**] In[**preposition**] view[**noun**], a[**article**] humble[**adjective**] vaudevillian[**adjective**] veteran[**noun**], cast[**verb**] vicariously[**adverb**] as both victim and villain by the vicissitudes of Fate.

NLP Techniques

Named Entity Recognition

NER is used to recognize entities in a text like people, organizations, places, and so on

India[*country*] was elected **Wednesday**[*time*] to the **UN Security Council**[*organization*] as non-permanent member with 184 of the 192 votes for a two-year term starting **January 1, 2021**[*time*].

Introducing a Chatbot

Why need Chatbot?

Cost and Time Effective

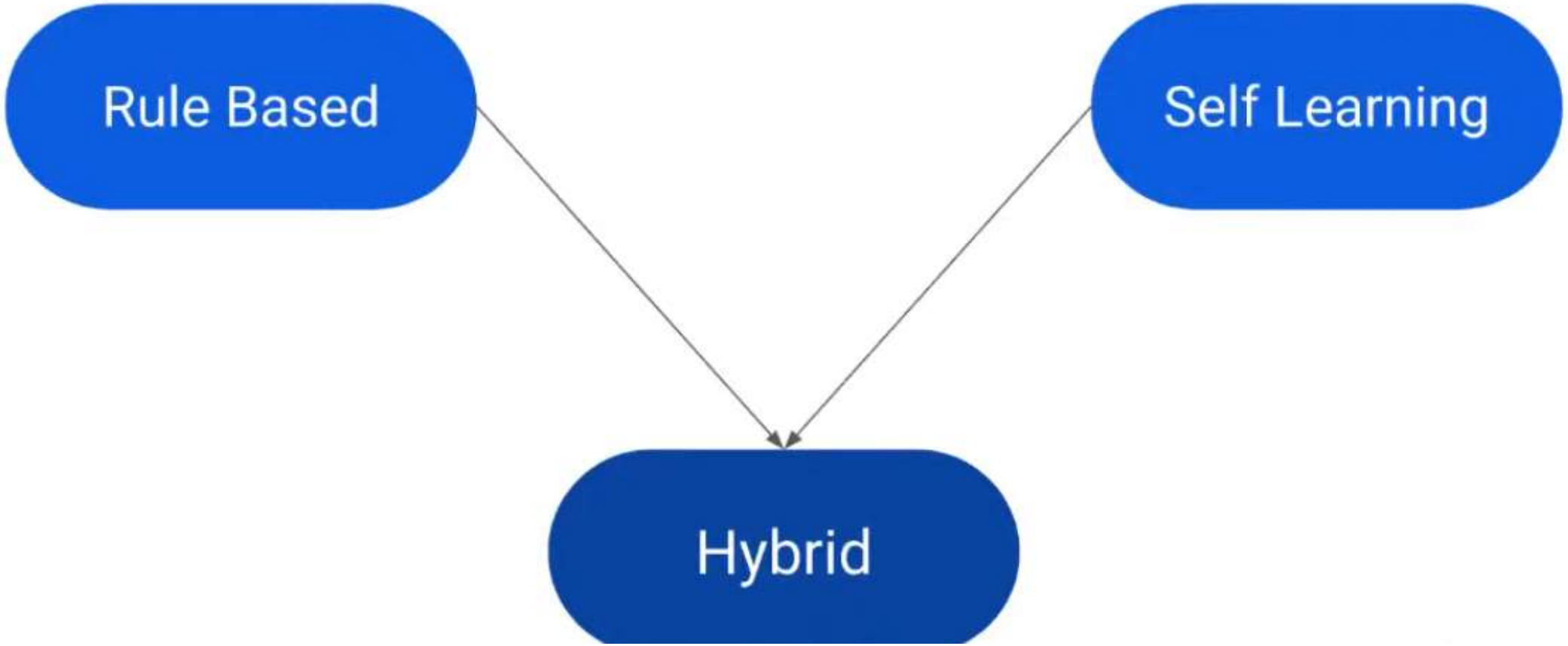
Cheap Development Cost

Why need Chatbot?

Human Resource

Business Branding

Types of Chatbot



Rule based

We can ask Ann to memorize what to say and in any situation, she just has to refer to her memory to give a reply.

Self Learning

Or we can have Ann behave intelligently and reply by not just referring to memory but by actively “thinking”.

Hybrid

We can have Ann memorize what to say in certain situations and also give "intelligent" reactions in other situations.

Chatbot examples

- https://www.chatbot.com/ (Rule based Chatbot)
- https://uae.sharafdg.com/ (Rule based Chatbot)
- https://www.snaps.io/ (ML Chatbot)

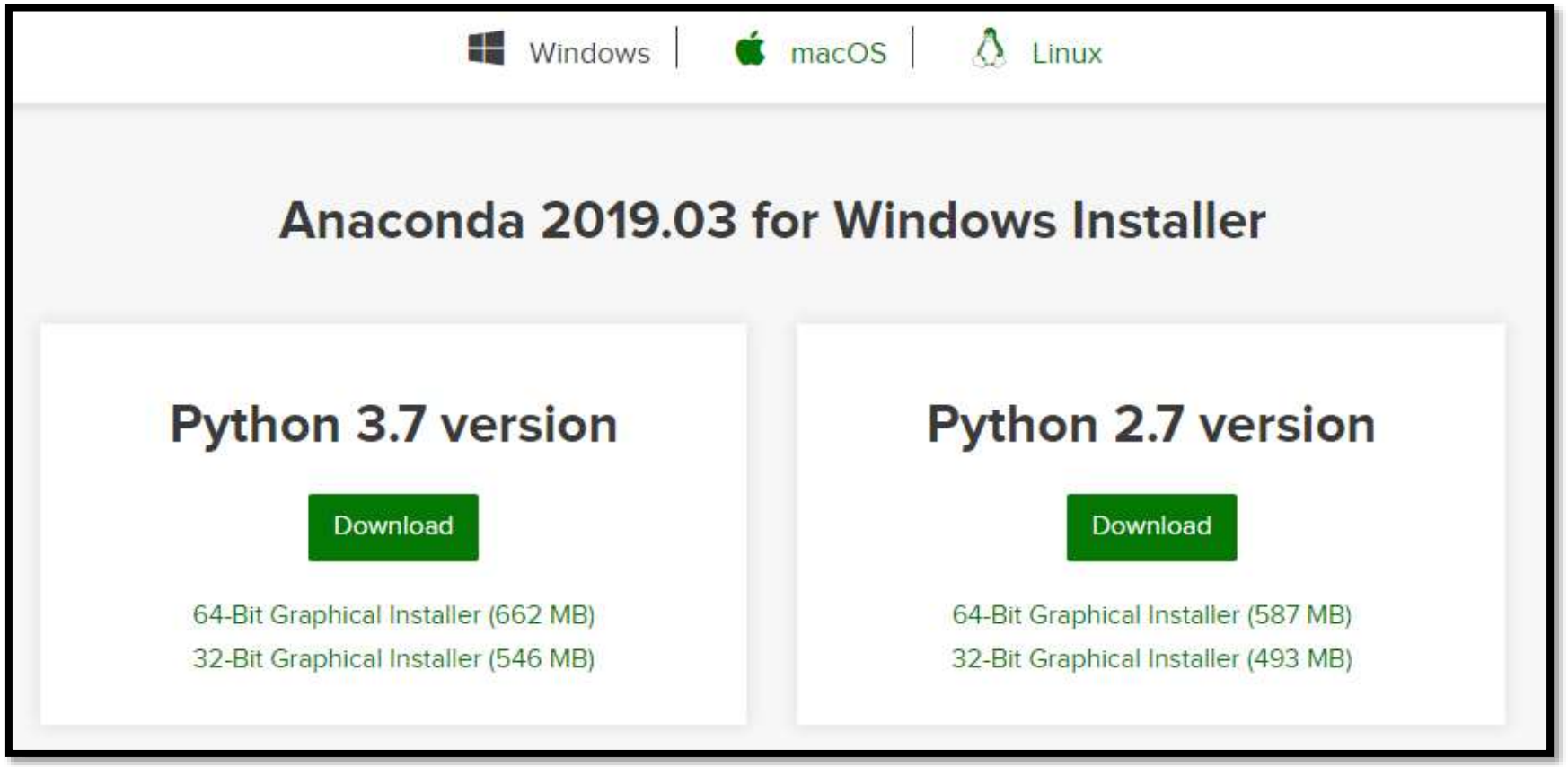
Installing Anaconda

Anaconda Distribution

The open-source Anaconda Distribution is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 11 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

1. Quickly download **1,500+ Python/R data science packages**
2. Manage libraries, dependencies, and environments with **Conda**
3. Develop and train machine learning and deep learning models with **scikit-learn, TensorFlow, and Theano**
4. Analyze data with scalability and performance with **NumPy& pandas.**
5. Visualize results with **Matplotlib, Bokeh, Datashader, and Holoviews**

Anaconda Installation



Or (https://docs.anaconda.com/anaconda/install/windows/)

Anaconda Installation

Download the Anaconda installer.

Double click the installer to launch.

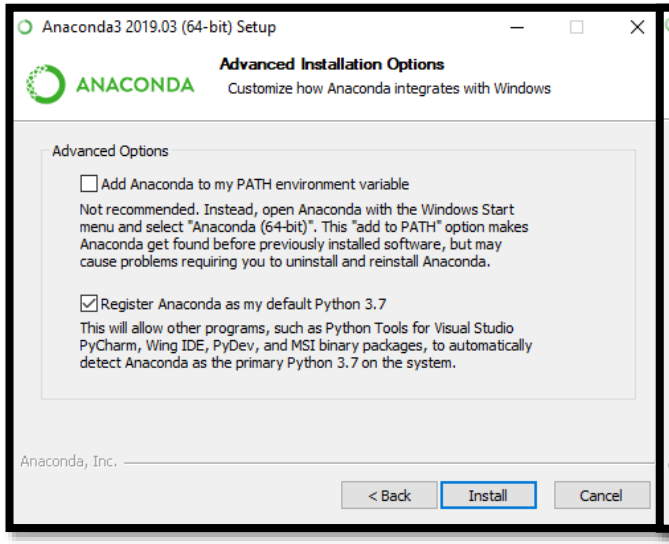
Click Next.

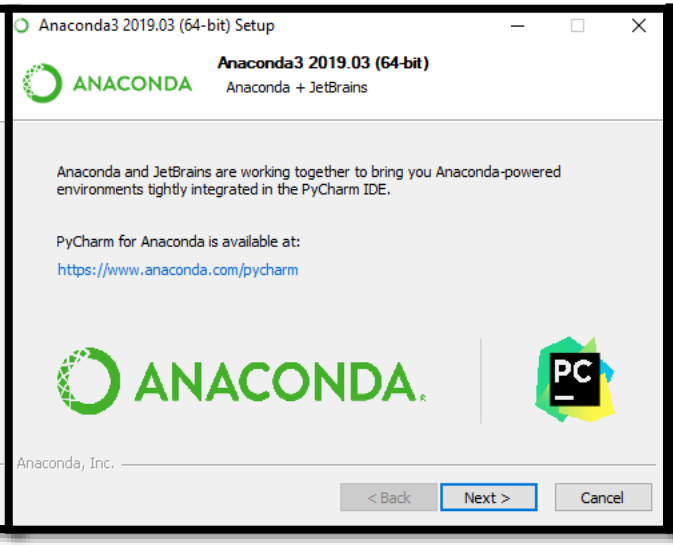
Read the licensing terms and click “I Agree”.

Select an install for “Just Me”

Select a destination folder to install Anaconda and click the Next button.

Anaconda Installation





Jupyter Overview

NLP Tasks Using Jupyter

NLP Tasks Practical Session

- Tokenization
- Stop words
- Stemming
- Lemmatization
- Part of speech tagging (POS)

Creating a Chatbot using Chatterbot

What is Chatterbot?

ChatterBot is a Python library that makes it easy to generate automated responses to a user's input. ChatterBot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users. For more details about the ideas and concepts behind ChatterBot see the *process flow diagram*.

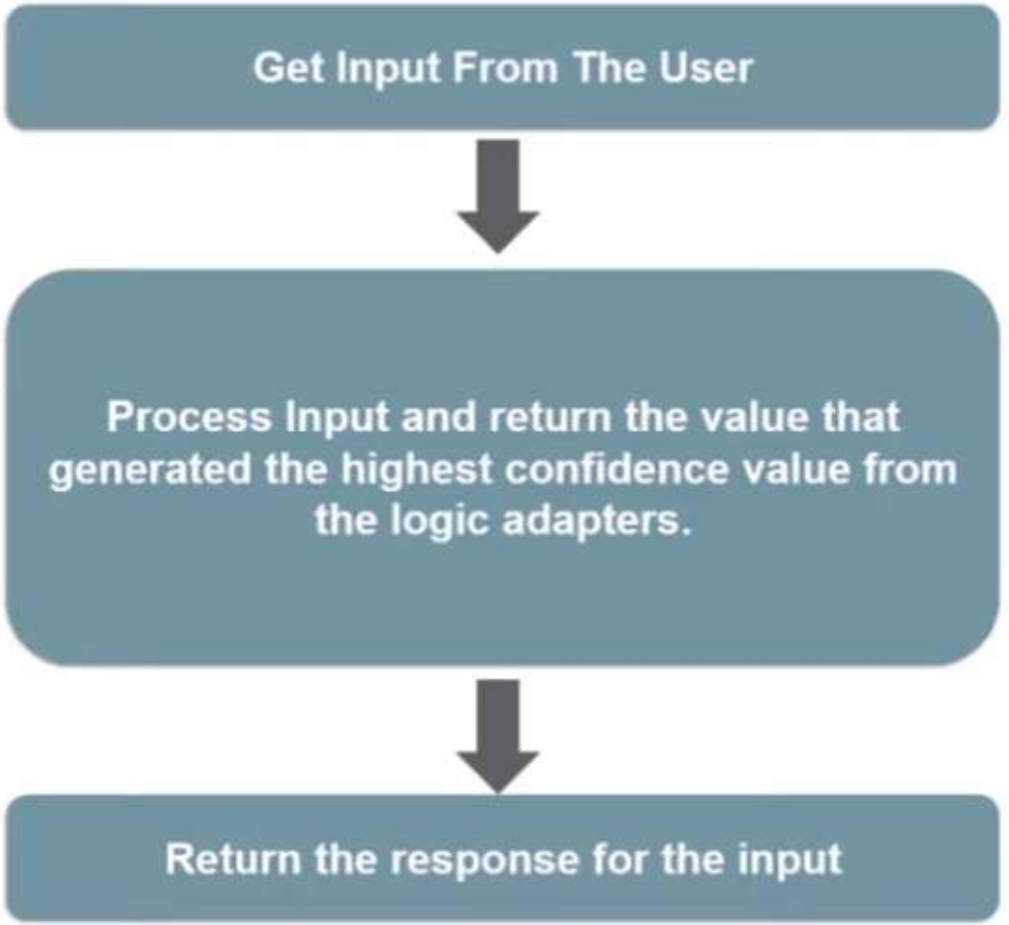
An example of typical input would be something like this:

```
user: Good morning! How are you doing?
bot:  I am doing very well, thank you for asking.
user: You're welcome.
bot:  Do you like hats?
```

How Chatterbot Works?

Every time a chatbot gets the input from the user, it saves the input and the response which helps the chatbot with no initial knowledge to evolve using the collected responses.

With increased responses, the accuracy of the chatbot also increases. The program selects the closest matching response from the closest matching statement that matches the input, it then chooses the response from the known selection of statements for that response.



Installation

The recommended method for installing ChatterBot is by using pip.

4.1.1 Installing from PyPi

If you are just getting started with ChatterBot, it is recommended that you start by installing the latest version from the Python Package Index (PyPi). To install ChatterBot from PyPi using pip run the following command in your terminal.

```
pip install chatterbot
```

4.1.2 Installing from GitHub

You can install the latest **development** version of ChatterBot directly from GitHub using `pip`.

```
pip install git+git://github.com/gunthercox/ChatterBot.git@master
```

Quick Start

The first thing you'll need to do to get started is install ChatterBot.

```
pip install chatterbot
```

See *Installation* for options for alternative installation methods.

4.2.1 Create a new chat bot

```
from chatterbot import ChatBot
chatbot = ChatBot("Ron Obvious")
```

Note: The only required parameter for the *ChatBot* is a name. This can be anything you want.

Training your chatbot

After creating a new ChatterBot instance it is also possible to train the bot. Training is a good way to ensure that the bot starts off with knowledge about specific responses. The current training method takes a list of statements that represent a conversation. Additional notes on training can be found in the *Training* documentation.

Note: Training is not required but it is recommended.

```
from chatterbot.trainers import ListTrainer

conversation = [
    "Hello",
    "Hi there!",
    "How are you doing?",
    "I'm doing great.",
    "That is good to hear",
    "Thank you.",
    "You're welcome."
]

trainer = ListTrainer(chatbot)

trainer.train(conversation)
```

Get a Response

4.2.3 Get a response

```
response = chatbot.get_response("Good morning!")
print(response)
```

Chatterbot Tutorial

This tutorial will guide you through the process of creating a simple command-line chat bot using ChatterBot.

4.3.2 Installing ChatterBot

You can install ChatterBot on your system using Python's pip command.

```
pip install chatterbot
```

4.3.3 Creating your first chat bot

Create a new file named *chatbot.py*. Then open *chatbot.py* in your editor of choice.

Before we do anything else, ChatterBot needs to be imported. The import for ChatterBot should look like the following line.

```
from chatterbot import ChatBot
```

Create a new instance of the `ChatBot` class.

```
bot = ChatBot('Norman')
```

This line of code has created a new chat bot named *Norman*. There is a few more parameters that we will want to specify before we run our program for the first time.

Setting Storage Adapter

ChatterBot comes with built in adapter classes that allow it to connect to different types of databases. In this tutorial, we will be using the `SQLStorageAdapter` which allows the chat bot to connect to SQL databases. By default, this adapter will create a SQLite database.

The `database` parameter is used to specify the path to the database that the chat bot will use. For this example we will call the database *sqlite:///database.sqlite3*. this file will be created automatically if it doesn't already exist.

```
bot = ChatBot(
    'Norman',
    storage_adapter='chatterbot.storage.SQLStorageAdapter',
    database_uri='sqlite:///database.sqlite3'
)
```

Note: The SQLStorageAdapter is ChatterBot's default adapter. If you do not specify an adapter in your constructor, the SQLStorageAdapter adapter will be used automatically.

Specify logic adapters

The *logic_adapters* parameter is a list of logic adapters. In ChatterBot, a logic adapter is a class that takes an input statement and returns a response to that statement.

You can choose to use as many logic adapters as you would like. In this example we will use two logic adapters. The TimeLogicAdapter returns the current time when the input statement asks for it. The MathematicalEvaluation adapter solves math problems that use basic operations.

```
bot = ChatBot(
    'Norman',
    storage_adapter='chatterbot.storage.SQLStorageAdapter',
    logic_adapters=[
        'chatterbot.logic.MathematicalEvaluation',
        'chatterbot.logic.TimeLogicAdapter'
    ],
    database_uri='sqlite:///database.sqlite3'
)
```

Getting Response

Next, you will want to create a while loop for your chat bot to run in. By breaking out of the loop when specific exceptions are triggered, we can exit the loop and stop the program when a user enters *ctrl+c*.

```
while True:
    try:
        bot_input = bot.get_response(input())
        print(bot_input)

    except(KeyboardInterrupt, EOFError, SystemExit):
        break
```

Train your chatbot

At this point your chat bot, Norman will learn to communicate as you talk to him. You can speed up this process by training him with examples of existing conversations.

```
from chatterbot.trainers import ListTrainer

trainer = ListTrainer(bot)

trainer.train([
    'How are you?',
    'I am good.',
    'That is good to hear.',
    'Thank you',
    'You are welcome.',
])
```

You can run the training process multiple times to reinforce preferred responses to particular input statements. You can also run the train command on a number of different example dialogs to increase the breadth of inputs that your chat bot can respond to.

Examples

4.4.1 Simple Example

```
from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer

# Create a new chat bot named Charlie
chatbot = ChatBot('Charlie')

trainer = ListTrainer(chatbot)
```

```
trainer.train([
    "Hi, can I help you?",
    "Sure, I'd like to book a flight to Iceland.",
    "Your flight has been booked."
])

# Get a response to the input text 'I would like to book a flight.'
response = chatbot.get_response('I would like to book a flight.')

print(response)
```

Terminal example

This example program shows how to create a simple terminal client that allows you to communicate with your chat bot by typing into your terminal.

```
from chatterbot import ChatBot


# Uncomment the following lines to enable verbose logging
# import logging
# logging.basicConfig(level=logging.INFO)

# Create a new instance of a ChatBot
bot = ChatBot(
    'Terminal',
    storage_adapter='chatterbot.storage.SQLStorageAdapter',
    logic_adapters=[
        'chatterbot.logic.MathematicalEvaluation',
        'chatterbot.logic.TimeLogicAdapter',
        'chatterbot.logic.BestMatch'
    ],
    database_uri='sqlite:///database.db'
)

print('Type something to begin...')
```

Terminal example-Cont

```
# The following loop will execute each time the user enters input
while True:
    try:
        user_input = input()

        bot_response = bot.get_response(user_input)

        print(bot_response)

    # Press ctrl-c or ctrl-d on the keyboard to exit
    except (KeyboardInterrupt, EOFError, SystemExit):
        break
```

Enhancing the chatbot

Time & Mathematics Example

ChatterBot has natural language evaluation capabilities that allow it to process and evaluate mathematical and time-based inputs.

```
from chatterbot import ChatBot


bot = ChatBot(
    'Math & Time Bot',
    logic_adapters=[
        'chatterbot.logic.MathematicalEvaluation',
        'chatterbot.logic.TimeLogicAdapter'
    ]
)

# Print an example of getting one math based response
```

Time & Mathematics Example

```
response = bot.get_response('What is 4 + 9?')
print(response)

# Print an example of getting one time based response
response = bot.get_response('What time is it?')
print(response)
```

Training the Chatbot

Training With Corpus Data

4.5.2.2 Training with corpus data

`chatterbot.trainers.`**`ChatterBotCorpusTrainer`**(*chatbot*, ***kwargs*)

Allows the chat bot to be trained using data from the ChatterBot dialog corpus.

ChatterBot comes with a corpus data and utility module that makes it easy to quickly train your bot to communicate. To do so, simply specify the corpus data modules you want to use.

Listing 4: chatbot.py

```
chatbot = ChatBot('Training Example')
```

Listing 5: train.py

```
from chatbot import chatbot
from chatterbot.trainers import ChatterBotCorpusTrainer

trainer = ChatterBotCorpusTrainer(chatbot)

trainer.train(
    "chatterbot.corpus.english"
)
```

Specifying corpus scope

Specifying corpus scope

It is also possible to import individual subsets of ChatterBot's corpus at once. For example, if you only wish to train based on the english greetings and conversations corpora then you would simply specify them.

Listing 6: train.py

```
trainer.train(
    "chatterbot.corpus.english.greetings",
    "chatterbot.corpus.english.conversations"
)
```

Quiz

Natural Language is any language that has evolved naturally through use and repetition without conscious planning or premeditation.

- ❑ True
- ❑ False

NLP is a science that focuses on _____.

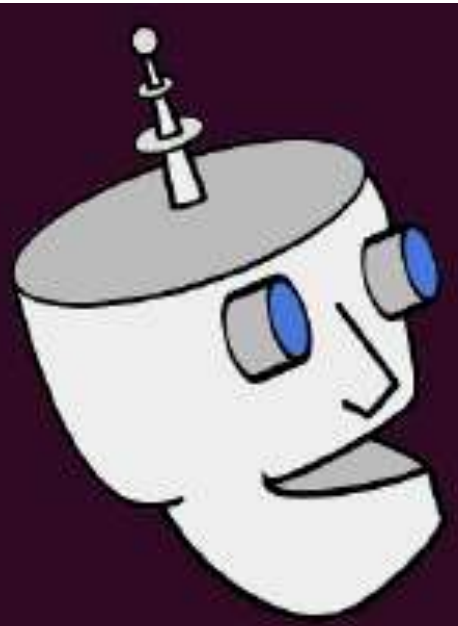
- ❑ Grammar
- ❑ Translation
- ❑ Speech recognition
- ❑ Software ability to understand and process human language.

Tokenizing a sentence is assigning ids to each word.

- ❑ True
- ❑ False

POS tagging is the process of assigning tags to tokens (words) like nouns, verbs, etc.

- ❑ True
- ❑ False



- Day2 -

ChatterBot

Machine learning, conversational dialog engine.

Done By: Eng. Mohammed Marwan Shahin
Technical Training Specialist
Technology Academy

Day.2

Outlines

Day 2:

- What is a Virtual Assistant?
- Speech Recognition & text to Speech
- How to create virtual assistant?
- Using API to empower our Chatbot and Assistant
- Deploying & Integrating a Chabot with Telegram.

Speech Recognition Library

```
pip install speechrecognition
pip install pyttsx3
```

Let us use Speech Recognition library

Let us convert text to speech using pyttsx3 Library

What is Virtual Assistant

A virtual assistant is an application that can understand **voice commands** and complete tasks for a user.

Google's assistant and Amazon's Alexa are good examples of virtual assistants.

Creating Virtual Assistant

https://www.youtube.com/watch?v=cUVLvRii4fk

How to create Virtual Assistant

- Get all required equipment's as mentioned below.
- Install the required Python Library (Chatterbot , Speech Recognition ..).
- Write python program in Raspberry Pi.
- Integrate your Assistant with many API as per the requirements.

Speaker

Raspberry Pi

Microphone

Let create our first simple Assistant

Let’s Create Virtual Assistant using Python Libraries

We are going to make use of the following programs that we have previously created:

- Chatbot
- Speech Recognition
- Text to Speech

Enhancing our Virtual Assistant

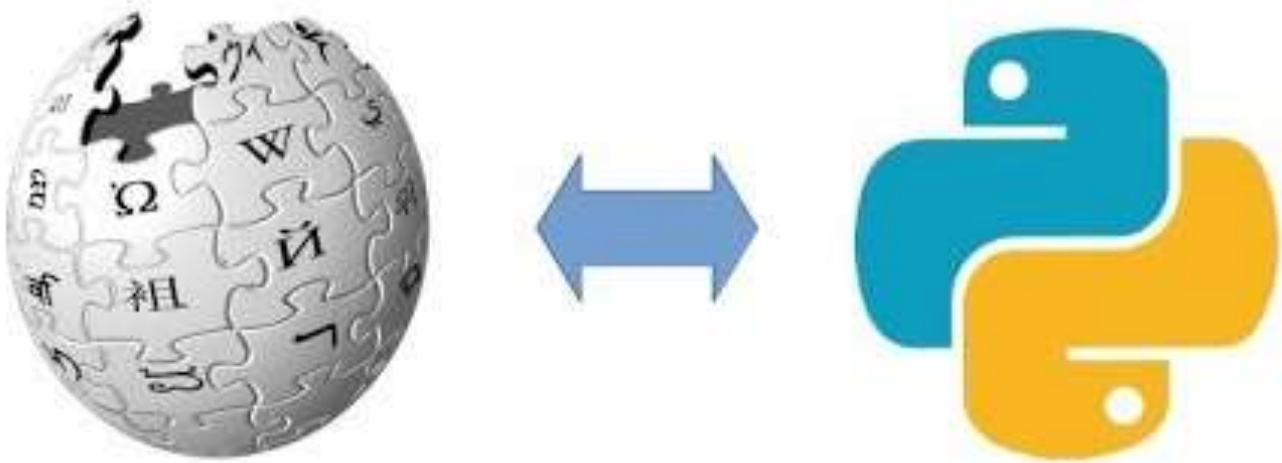
In addition to the above we can use some API's other libraries to increase from our Chabot capabilities:

- wikipedia (API) # to get access to Wikipedia database
- webbrowser # to run a website using a browser
- os # to open a local file from our hard drive

Let us Empower our assistant using API’s

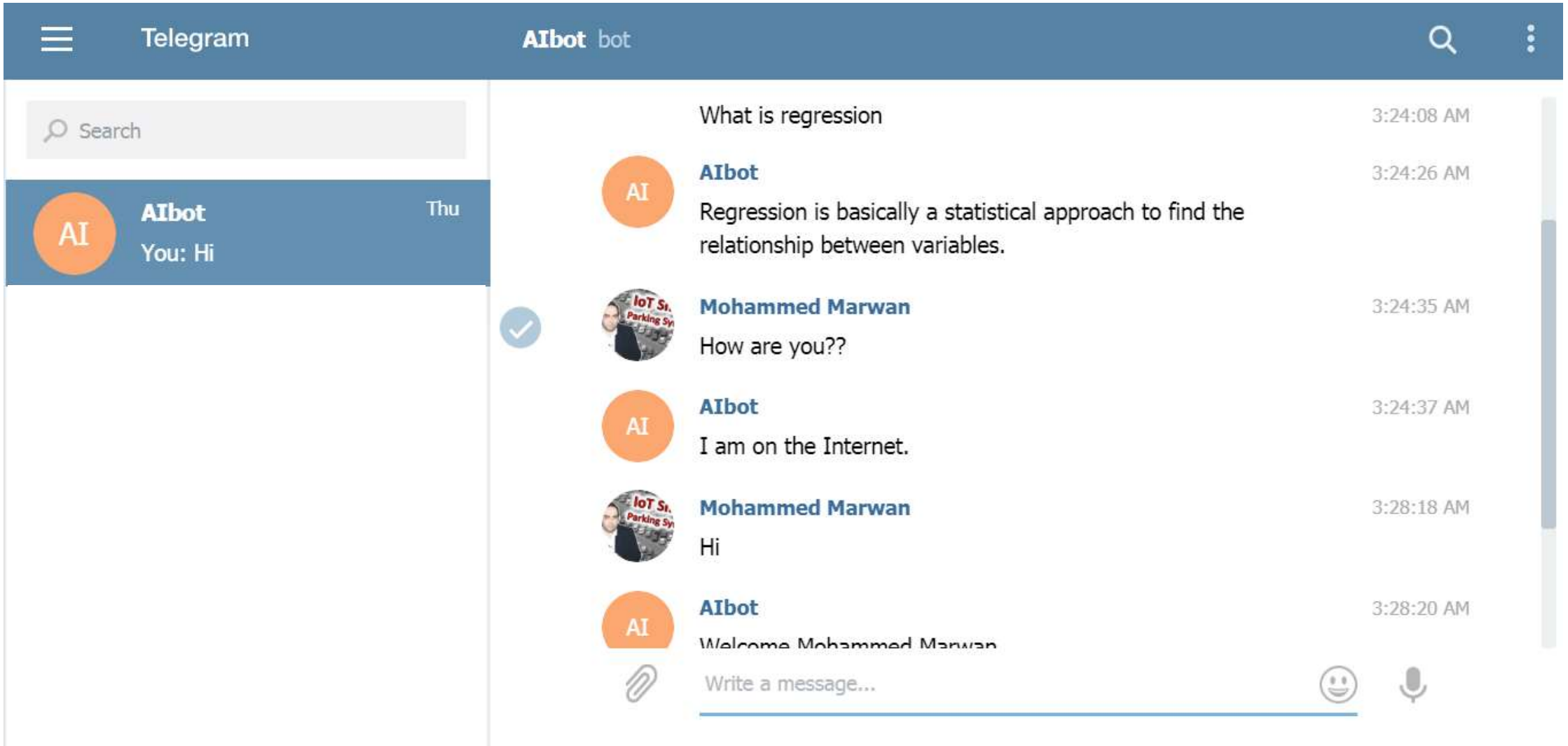
Wikipedia API:

This provides developers code-level access to the entire Wikipedia reference. The goal of this API is to provide direct, high-level access to the data contained in the Wikipedia databases.



Integrating Chatbot to Telegram

What you will learn by the end of this course?



Integrating Chatbot with Telegram

Creating a new bot

Use the **/newbot** command to create a new bot. The BotFather will ask you for a name and username, then generate an authorization token for your new bot.
The **name** of your bot is displayed in contact details and elsewhere.
The **Username** is a short name, to be used in mentions and t.me links. Usernames are 5-32 characters long and are case insensitive, but may only include Latin characters, numbers, and underscores. Your bot's username **must** end in ‘bot’, e.g. ‘tetris_bot’ or ‘TetrisBot’.

The **token** is a string along the lines of 110201543:AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw that is required to authorize the bot and send requests to the Bot API. Keep your token secure and store it safely, it can be used by anyone to control your bot.

Note: After creating you chatbot account in Telegram you can share it with people using the username for example : (moh198699bot)

Congratulations !!
Your Chatbot is finally ready & Accessible by Telegram

Thank you !!