To get started with your chatbot project, create and activate a [virtual environment](#), then install `chatterbot` and `pytz`:

Here are the 4 steps to create a chatbot in Python from scratch:

1. *Import and load the data file*
2. *Preprocess data*
3. *Build the mode*
4. *Predict the response*

1.Import and load the data file:

```
PS> python -m venv venv
PS> venv\Scripts\activate
(venv) PS> python -m pip install chatterbot==1.0.4 pytz
```

- *First, make a file name as train_chatbot.py. We import the necessary packages for our chatbot and initialize the variables we will use in our Python*

project. 8. The data file is in JSON format so we used the json package to parse the JSON file into Python.

- Running these commands in your terminal application installs ChatterBot and its dependencies into a new Python virtual environment.

- After the installation is complete, running python -m pip freeze should bring up list of installed dependencies that's similar to what you can find in the provided sample code's requirements.txt file:

2.preprocess data:
- When working with text data, we need to perform various preprocessing on the data before we make a machine learning or a deep learning model. Based on the requirements we need to apply various operations to preprocess the data. Tokenizing is the most basic and first thing you can do on text data. Tokenizing is the process of breaking the whole text into small parts like words. Here we iterate through the patterns and tokenize the sentence using nltk.word_tokenize() function and append each word in the words list.

- *We also create a list of classes for our tags. Now we will lemmatize each word and remove duplicate words from the list. Lemmatizing is the process of converting a word into its lemma form and then creating a pickle file to store the Python objects which we will use while predicting.*

```
# bot.py

From chatterbot import ChatBot

Chatbot = ChatBot("Chatpot")

Exit_conditions = (":q", "quit", "exit")
While True:
    Query = input("> ")
    If query in exit_conditions:
        Break
    Else:
        Print(f"🤖 {chatbot.get_response(query)}")
```

3.building a model:

- *We have our training data ready, now we will build a deep neural network that has 3 layers. We use the Keras sequential API for this. After training the model for 200 epochs, we achieved 100% accuracy on our model. Let us save the model as 'chatbot_model.h5'.*

- *After the language models are set up, you'll see the greater than sign (>) that you defined in bot.py as your input prompt. You can now start to interact with your chatty pot:*

> *Hello*

? *hello*

> *Are you a plant?*

? *hello*

> *Can you chat, pot?*

? *hello*

4. Predict the response (Graphical User Interface) :

- *To predict the sentences and get a response from the user to let us create a new file 'chatapp.py'.*
- *We will load the trained model and then use a graphical user interface that will predict the response from the bot.*
- *The model will only tell us the class it belongs to, so we will implement some functions which will identify the class and then retrieve us a random response from the list of responses. Again we import the necessary packages and load the 'words.pkl' and 'classes.pkl' pickle files which we have created when we trained our model: To predict the class, we will need to provide input in the same way as we did while training.*
- *So we will create some functions that will perform text preprocessing and then predict the class. After predicting the class, we will get a random response from the list of intents.*

5.Run the chatbot :

To run the chatbot, we have two main files; train_chatbot.py and chatapp.py. First, we train the model using the command in the terminal: python

train_chatbot.py If we don't see any error during training, we have successfully created the model. Then to run the app, we run the second file. Python chatgui.py

Team members:

R.karthika Devi

M.vasantha kumari

S.Babitha

T.Abinaya lakshmi

A.Indhuja