DEVOPS:

Prerequisites:

Import the necessary packages in this case it is the following

```
import nltk

from nltk.stem.lancaster import LancasterStemmer
stemmer = LancasterStemmer()

import numpy
import tflearn
import tensorflow
import random
import json
import pickle
```

Note: U might get an error while importing nltk "Unable to load the resource nltk.punkt"
In such case the above stated package would have not been installed in your system to can install this package by including this statement : nltk.download("punkt") or else if u wish to install all the nltk packages(recommended) use this statement : nltk.download()

Open the intents.json file and load the file contents into variable data .

Data Preprocessing:

In this step we will be doing all the data segmentation and cleaning actitivies.

STEP 1: We need to find all the possible words to construct the bag of words model. Fo this purpose we use work tokenizer function. It basically returns the words splitted by space and also extracts apostrophe like 's ,'nt,'ld etc and treat it as special words.

Words -> contains all the possible words in the patterns or questions
Dimension : 1 X no of words

doc_x -> contains sentence/ question wise words
Dimension:22 rows -> each row containing words corresponding to that pattern/question

doc_y_> contains labels or tags for each sentence
Dimension: 1X22

Labels -> It contains all the possible labels. Dimension : 1X no_of_labels

STEP 2: In this step we will be stemming the words.

Stemming the words means removing words with same meaning that is basically avoiding redundant data.

Eg : I am currently riding car
The car is ride by me .

In this case both the words riding and ride convey the same meaning so we treat the both word as ride.
Here i have used LancasterStemmer refer the below link to understand its working.

Link: https://www.nltk.org/_modules/nltk/stem/lancaster.html

All the question marks are removed during stemming.

STEP 3: Then stemmed words then sorted and duplicate is avoided using sets.
So words has the final set of words and labels has the possible tag names.

Training the model:

As neural networks understand only no we need to convert these string into some form of nos
THIS IS DONE USING BAG OF WORDS MODEL

enumerate(doc_x) simply return a tuple in the following format:

(index,list of words)

x-> index
doc-> list od words

Example :
0 ['Hi']
1 ['How', 'are', 'you']
2 ['Is', 'anyone', 'there', '?']
3 ['Hello']
4 ['Good', 'day']
5 ['Bye']
6 ['See', 'you', 'later']
7 ['Goodbye']
8 ['Thanks']
9 ['Thank', 'you']

Now we take each element of the enumurated type and again stem them to form sequence of unredudant words (this is because only the words list has the stemmed words but doc_x simply contains list of words question wise as stated above).

These stemmed words are then tested against the list of possible words (**words list**) and bag of words is constructed for that sentence and then appended to training list and the corresponding tag is appended to output list.

Training : contains bag of words for each sentence
Dimension : No of sentence X Size(words list) --------->input data to input laer

Output : contains tag for each sentence
Dimension : 1X No of sentence ---------->input data for output layer

Model architecture:

Input layer : BATCH SIZE X BAG_OF_WORDS
Fully Connected Layer (Hidden) : 8 nodes
Fully Connected Layer (Hidden) : 8 nodes
Output layer : NO OF POSSIBLE TAGS

Save and restore :
In order to avoid data preprocessing and model train each time we run it we make use of try-except to check weather the training and data preprocessing is already done.

```
Try:
 with open("data.pickle", "rb") as f:
      words, labels, training, output = pickle.load(f)
Except:
  Data preprocessing part

try :
   model.load("model.tflearn")
except:
   model = tflearn.DNN(net)
   model.fit(training, output, n_epoch=1000, batch_size=8, show_metric=True)
   model.save("model.tflearn")
```

FINALLY THE CHIITIBOT IS READY:
The user finally input the sentence and the bag of words for that input is computed using **bag_of_words function** it is then passed to NN and output result is obtained.