Code:

```
import numpy as np

import pandas as pd

import warnings

warnings.filterwarnings('ignore')

import json


with open('intents.json', 'r') as f:

    data = json.load(f)


df = pd.DataFrame(data['intents'])

df

dic = {"tag":[], "patterns":[], "responses":[]}

for i in range(len(df)):

    ptrns = df[df.index == i]['patterns'].values[0]

    rspns = df[df.index == i]['responses'].values[0]

    tag = df[df.index == i]['tag'].values[0]

    for j in range(len(ptrns)):

        dic['tag'].append(tag)

        dic['patterns'].append(ptrns[j])

        dic['responses'].append(rspns)


df = pd.DataFrame.from_dict(dic)

df
```

```
df['tag'].unique()

from tensorflow.keras.preprocessing.text import Tokenizer


tokenizer = Tokenizer(lower=True, split=' ')

tokenizer.fit_on_texts(df['patterns'])

tokenizer.get_config()

vacab_size = len(tokenizer.word_index)

print('number of unique words = ', vacab_size)

from tensorflow.keras.preprocessing.sequence import pad_sequences

from sklearn.preprocessing import LabelEncoder


ptrn2seq = tokenizer.texts_to_sequences(df['patterns'])

X = pad_sequences(ptrn2seq, padding='post')

print('X shape = ', X.shape)


lbl_enc = LabelEncoder()

y = lbl_enc.fit_transform(df['tag'])

print('y shape = ', y.shape)

print('num of classes = ', len(np.unique(y)))

import tensorflow

from tensorflow import keras

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Input, Embedding, LSTM, LayerNormalization, Dense,
Dropout

from tensorflow.keras.utils import plot_model
```

```
model = Sequential()

model.add(Input(shape=(X.shape[1])))

model.add(Embedding(input_dim=vacab_size+1, output_dim=100, mask_zero=True))

model.add(LSTM(32, return_sequences=True))

model.add(LayerNormalization())

model.add(LSTM(32, return_sequences=True))

model.add(LayerNormalization())

model.add(LSTM(32))

model.add(LayerNormalization())

model.add(Dense(128, activation="relu"))

model.add(LayerNormalization())

model.add(Dropout(0.2))

model.add(Dense(128, activation="relu"))

model.add(LayerNormalization())

model.add(Dropout(0.2))

model.add(Dense(len(np.unique(y)), activation="softmax"))

model.compile(optimizer='adam', loss="sparse_categorical_crossentropy",
metrics=['accuracy'])


model.summary()

plot_model(model, show_shapes=True)

model_history = model.fit(x=X,

                  y=y,

                  batch_size=10,
                    callbacks=[tensorflow.keras.callbacks.EarlyStopping(monitor='accuracy',
patience=3)],
                    epochs=50)
import re
import random


def generate_answer(pattern):
    text = []
    txt = re.sub('[^a-zA-Z\']', ' ', pattern)
    txt = txt.lower()
    txt = txt.split()
    txt = " ".join(txt)
    text.append(txt)


    x_test = tokenizer.texts_to_sequences(text)
    x_test = np.array(x_test).squeeze()
    x_test = pad_sequences([x_test], padding='post', maxlen=X.shape[1])
    y_pred = model.predict(x_test)
    y_pred = y_pred.argmax()
    tag = lbl_enc.inverse_transform([y_pred])[0]
    responses = df[df['tag'] == tag]['responses'].values[0]


    print("you: {}".format(pattern))
    print("model: {}".format(random.choice(responses)))
generate_answer('Hi! How are you?')
generate_answer('Maybe I just didn\'t want to be born :)')
generate_answer('help me:')
generate_answer(':')
def chatbot():
    print("Chatbot: Hi! I'm your friendly chatbot. How can I assist you today?")


    while True:
        user_input = input("You: ")
        if user_input.lower() in ['quit', 'exit', 'q', 'bye']:
            print("Chatbot: Goodbye!")
            break


        generate_answer(user_input)


if __name__ == "__main__":
    chatbot()
```