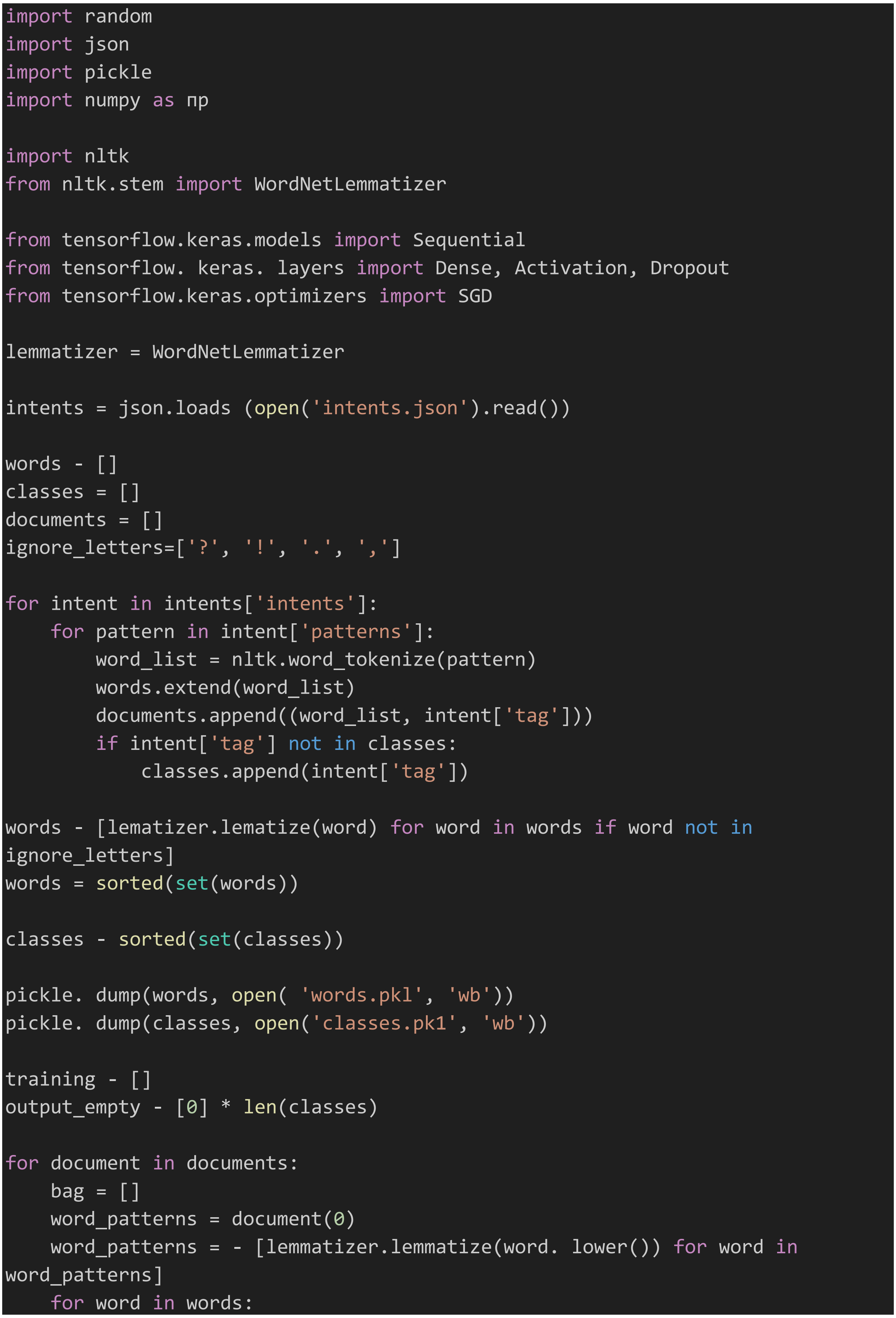
# Create a Chatbot in Python

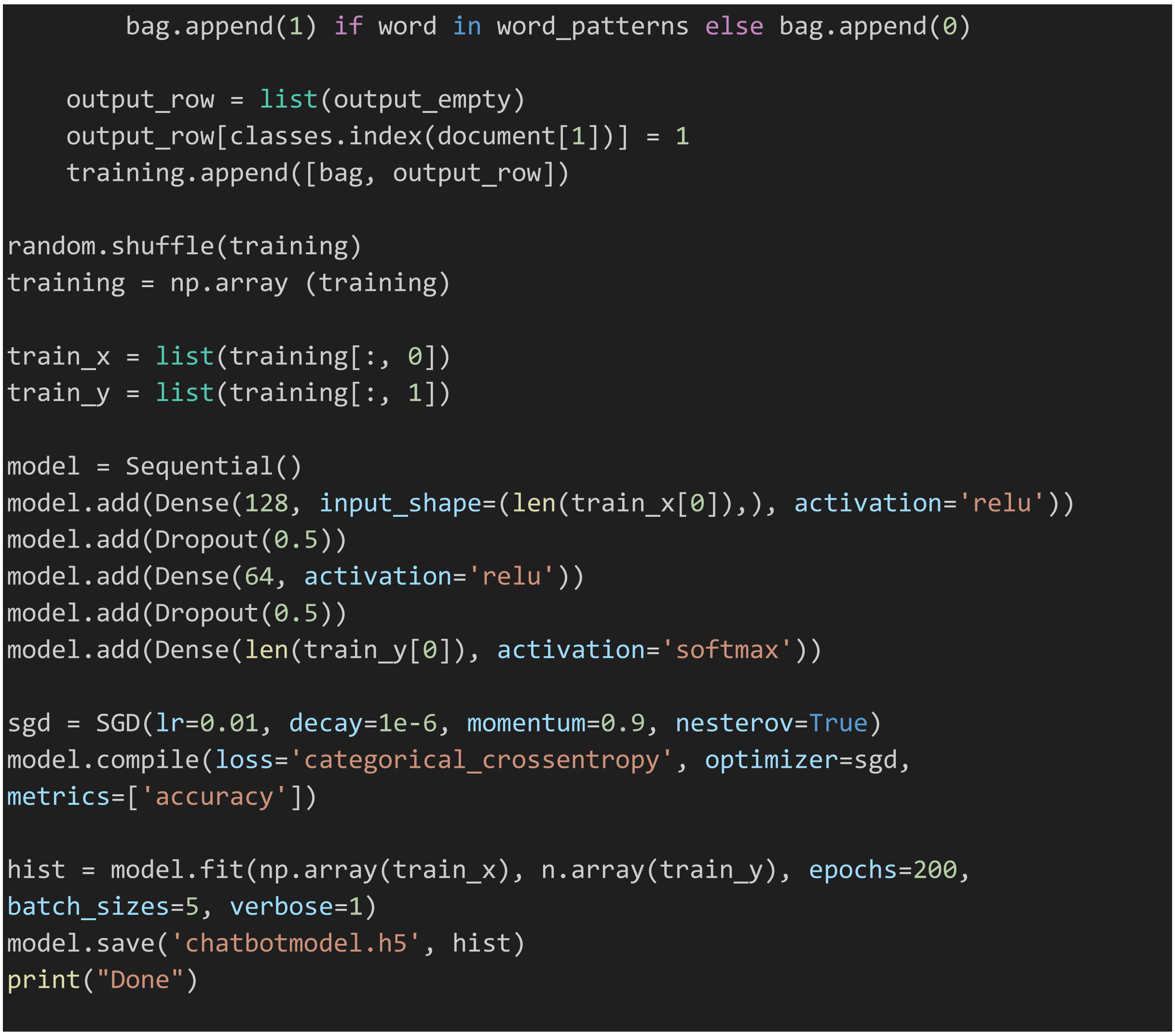
## Save “intents.json”:

```
{"intents":  [
    {"tag": "greetings",
    "pattern": ["hello","hey","hi","good day","Greetings","What's up?","how is
it going?"],
    "responses": ["Hello!","Hey!","what can I do for you?"]
},
    {"tag": "goodbye",
    "patterns": ["cya", "See you later", "Goodbye", "I am Leaving", "Have a
Good day", "bye", "cao", "see ya"],
    "responses": ["Sad to see you go :(", "Talk to you later", "Goodbye!"]
},
     {"tag": "age",
     "patterns": ["how old", "how old is florian", "what is your age", "how
old are you", "age?"],
     "responses": ["My owner Florian is 21 years old!", "21 years!"]
},
     {"tag": "name",
     "patterns": ["what is your name", "what should I call you","whats your
name?","who are you","Can you tell me"],
     "responses": ["You can call me Neural!.", "I'm Neural!", "I'm Neural the
assistant of Florian!"]
},
    {"tag": "shop",
    "patterns": ["Id like to buy something", "what are your products?", "what
do you recommend?", "What are you selling"],
    "responses": ["NeuralNine has a seven part Python book series and a lot of
free content on YouTube and the Blog a website"]
},
    {"tag": "hours",
    "patterns": ["when are you guys open", "what are your hours", "hours of
operation"],
    "responses": ["24/7"]
},
    {"tag": "stocks",
    "patterns": ["what stocks do I own?", "how are my shares?", "what
companies am I investing in?", "what am I doing"],
    "responses": ["You own THE following shares: ABBV, AAPL, FB, NVDA and an
ETF of the S&P 500 Index!"]
}
]}
```

Save “training.py”:



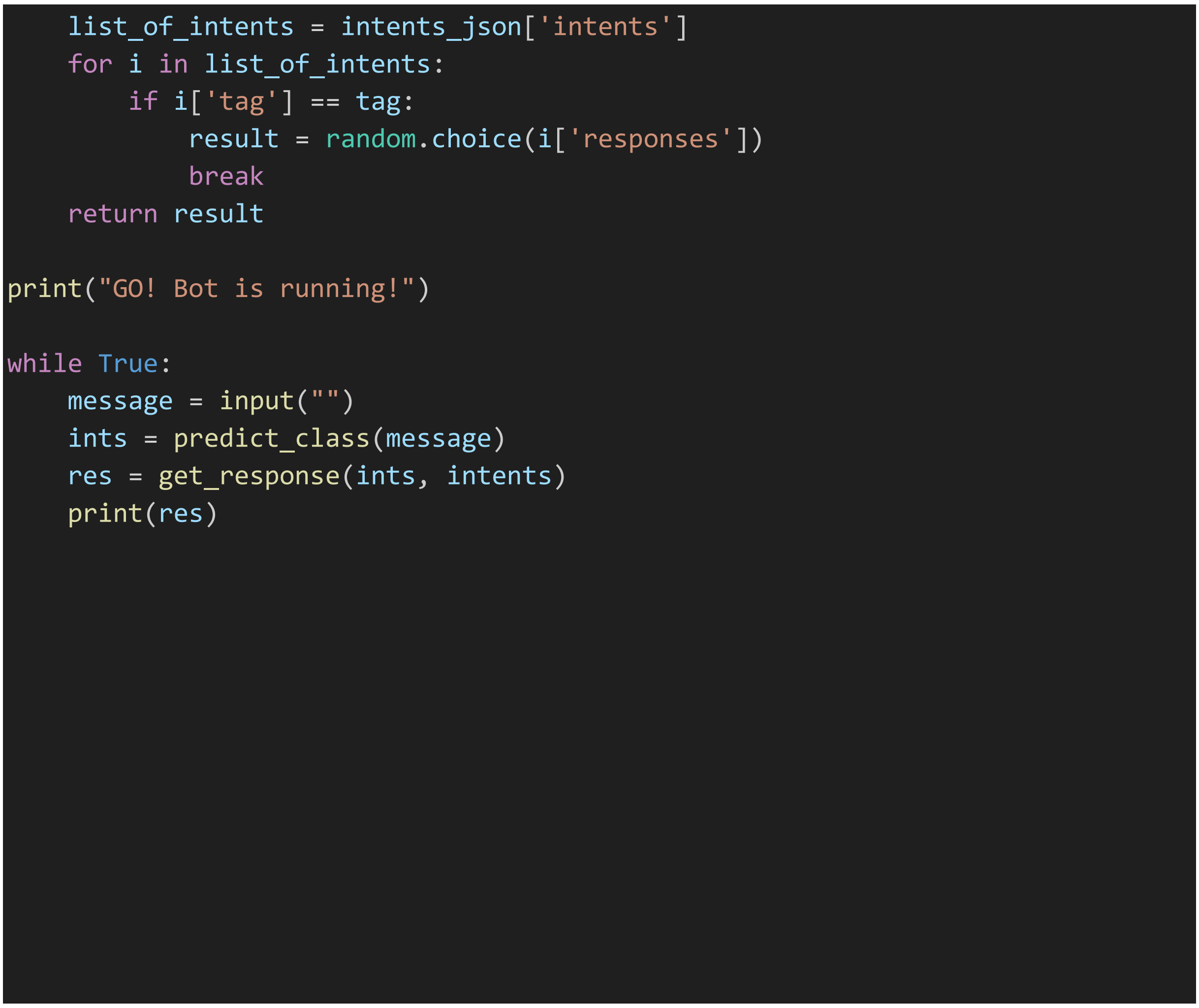
```
import random
import json
import pickle
import numpy as пр

import nltk
from nltk.stem import WordNetLemmatizer

from tensorflow.keras.models import Sequential
from tensorflow. keras. layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers import SGD

lemmatizer = WordNetLemmatizer

intents = json.loads (open('intents.json').read())

words - []
classes = []
documents = []
ignore_letters=['?', '!', '.', ',']

for intent in intents['intents']:
    for pattern in intent['patterns']:
        word_list = nltk.word_tokenize(pattern)
        words.extend(word_list)
        documents.append((word_list, intent['tag']))
        if intent['tag'] not in classes:
            classes.append(intent['tag'])

words - [lematizer.lematize(word) for word in words if word not in
ignore_letters]
words = sorted(set(words))

classes - sorted(set(classes))

pickle. dump(words, open( 'words.pkl', 'wb'))
pickle. dump(classes, open('classes.pk1', 'wb'))

training - []
output_empty - [0] * len(classes)

for document in documents:
    bag = []
    word_patterns = document(0)
    word_patterns = - [lemmatizer.lemmatize(word. lower()) for word in
word_patterns]
    for word in words:
```



```
        bag.append(1) if word in word_patterns else bag.append(0)

    output_row = list(output_empty)
    output_row[classes.index(document[1])] = 1
    training.append([bag, output_row])

random.shuffle(training)
training = np.array (training)

train_x = list(training[:, 0])
train_y = list(training[:, 1])

model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd,
metrics=['accuracy'])

hist = model.fit(np.array(train_x), n.array(train_y), epochs=200,
batch_sizes=5, verbose=1)
model.save('chatbotmodel.h5', hist)
print("Done")
```

Save “Chatbot.py”:

```
import random
import json
import pickle
import numpy as пp

import nltk
from nltk.stem import WordNetLemmatizer

from tensorflow.keras.models import load_model

lemmatizer = WordNetLemmatizer()
intents = json. loads(open(' intents. json').read())

word = pickle.load(open('words.pk1', 'rb'))
classes = pickle.load(open('classes.pk1', 'rb'))
model = load_model('chatbot_model.model')

def clean_up_sentence(sentence):
    sentence_words = nltk.word_tokenize(sentence)
    sentence_words = [lenmatizer.lemmatize(word) for word in sentence_words]
    return sentence_words

def bag_of_words(sentence):
    sentence_words = clean_up_sentence(sentence)
    bag = [0] * len(words)
    for w in sentence_words:
        for i, word in enumerate(words):
            if word == w:
                bag[i] = 1
    return np.array(bag)


def predict_class(sentence):
    bow = bag_of_words(sentence)
    res = model.predict(np.array([bow]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i, r] for i, r in enumerate(res) if r > ERROR_THRESHOLD]

    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append({'intent': classes[r[0]], 'probability':
str(r[1])})
    return return_list

def get_response(intents_list, intents_json):
    tag = intents_list[0]['intent']
```



```python
    list_of_intents = intents_json['intents']
    for i in list_of_intents:
        if i['tag'] == tag:
            result = random.choice(i['responses'])
            break
    return result

print("GO! Bot is running!")

while True:
    message = input("")
    ints = predict_class(message)
    res = get_response(ints, intents)
    print(res)
```