# Chatbot In Python

#### Project: House Price Prediction



Creating a chatbot in Python involves multiple components, and abstraction and modularization are crucial for building a maintainable and scalable chatbot application. Here's a high-level overview of how you can approach abstraction and module organization for a Python chatbot:

**Chatbot's Purpose:**

Start by defining the purpose and functionality of your chatbot. What tasks or conversations should it handle? This will help you determine the modules and abstractions you need.

**Natural Language Processing** (NLP):

Abstraction is essential for handling natural language understanding and generation. You can use popular NLP libraries like spaCy, NLTK, or Hugging Face Transformers. Create a separate module or class for NLP-related tasks. For example:

**Program:**

# nlp\_module.py

import spacy

class NLPProcessor:

def \_\_init\_\_(self):

self.nlp = spacy.load("en\_core\_web\_sm")

def process\_text(self, text):

# Implement NLP processing here

Pass

**Conversational Logic:**

Create a module or class that handles the conversational logic of your chatbot. This should include functions for managing conversations, generating responses, and maintaining context. For example:

**Program:**

# chatbot\_logic.py

class ChatbotLogic:

def \_\_init\_\_(self):

self.context = {} # Store conversation context

def generate\_response(self, user\_input):

# Implement chatbot logic here

pass

def update\_context(self, user\_input, bot\_response):

# Update conversation context

Pass

**External Services:**

If your chatbot needs to interact with external services or APIs, create separate modules or classes to handle these interactions. This promotes modularity and maintainability. For example:

**Program:**

# external\_services.py

import requests

class ExternalServiceClient:

def \_\_init\_\_(self, api\_key):

self.api\_key = api\_key

def call\_api(self, endpoint, params):

# Implement API calls here

pass

**Main Application Module:**

Finally, create a main module that ties everything together. This module should instantiate the NLP processor, chatbot logic, and any other necessary components, and orchestrate the conversation flow. This is where you can use a loop to continuously accept user input and generate responses.

Program:

# main.py

from nlp\_module import NLPProcessor

from chatbot\_logic import ChatbotLogic

nlp\_processor = NLPProcessor()

chatbot = ChatbotLogic()

while True:

user\_input = input("User: ")

nlp\_output = nlp\_processor.process\_text(user\_input)

bot\_response = chatbot.generate\_response(nlp\_output)

chatbot.update\_context(user\_input, bot\_response)

print(f"Bot: {bot\_response}")

**Configuration and Settings:**

You may also want to create a separate module for configuration and settings, allowing you to easily modify parameters like API keys, conversation thresholds, and more without changing the core logic.

**Testing and Debugging Modules:**

Consider creating separate modules or scripts for testing and debugging to ensure your chatbot works correctly.

**Source Code:**

from nltk.chat.util import Chat, reflections

pairs = [

[

r"my name is (.\*)",

["Hello %1, How are you today ?",]

],

[

r"what is your name ?",

["My name is Chatty and I'm a chatbot ?",]

],

[

r"how are you ?",

["I'm doing good\nHow about You ?",]

],

[

r"sorry (.\*)",

["Its alright","Its OK, never mind",]

],

[

r"i'm (.\*) doing good",

["Nice to hear that","Alright :)",]

],

[

r"hi|hey|hello",

["Hello", "Hey there",]

],

[

r"(.\*) age?",

["I'm a computer program dude\nSeriously you are asking me this?",]

],

[

r"what (.\*) want ?",

["Make me an offer I can't refuse",]

],

[

r"(.\*) created ?",

["Nagesh created me using Python's NLTK library ","top secret ;)",]

],

[

r"(.\*) (location|city) ?",

['Chennai, Tamil Nadu',]

],

[

r"how is weather in (.\*)?",

["Weather in %1 is awesome like always","Too hot man here in %1","Too cold man here in %1","Never

even heard about %1"]

],

[

r"i work in (.\*)?",

["%1 is an Amazing company, I have heard about it. But they are in huge loss these days.",]

],

[

r"(.\*)raining in (.\*)",

["No rain since last week here in %2","Damn its raining too much here in %2"]

],

[

r"how (.\*) health(.\*)",

["I'm a computer program, so I'm always healthy ",]

],

[

r"(.\*) (sports|game) ?",

["I'm a very big fan of Football",]

],

[

r"who (.\*) sportsperson ?",

["Messy","Ronaldo","Roony"]

],

[

r"who (.\*) (moviestar|actor)?",

["Brad Pitt"]

],

[

r"quit",

["BBye take care. See you soon :) ","It was nice talking to you. See you soon :)"]

],

]

def chatty():

print("Hi, I'm Chatty and I chat alot ;)\nPlease type lowercase English language to start a

conversation. Type quit to leave ") #default message at the start

chat = Chat(pairs, reflections)

chat.converse()

if \_\_name\_\_ == "\_\_main\_\_":

chatty()

**Trigger the conversation:**

chat.converse()

**conversation :**

In [27]: runfile("D:/KJ/Phase2/Downloads/chatbot/chatbot\_1.py', wdir="D:/KJ/Nagesh/Downloads/chatbot')

Hi, I'm Chatty and I chat alot ;)

Please type lowercase English language to start a conversation. Type quit to leave

>hi

Hey there

>how are you?

I'm doing good

How about You ?

>i'm also doing good

Nice to hear that

>what is your name ?

My name is Chatty and I'm a chatbot

>what is your age?

I'm a computer program dude

Seriously you are asking me this?

>sorry Chatty

Its OK, never mind

>what is your location?

Chennai, Tamil Nadu

>how is weather in Chennai?

Too cold man here in Chennai

>i work in abc company

abc company is an amazing company, I have heard about it. But they are in huge loss these days.

>is it raining in Chennai

Damn its raining too much here in Chennai

>what is your fav game?

I'm a very big fan of football

>who is your fav moviestar?

Brad Pitt

>quit

Bye take care. See you soon :)

**Conclusion:**

In conclusion, building a chatbot in Python is a dynamic and evolving process that requires careful planning, development, and ongoing refinement. Chatbots have become valuable tools for businesses and organizations to enhance customer engagement, automate tasks, and provide assistance.