



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

CHATBOT IN PYTHON

A Report for the Evaluation 3 of Project 2

Submitted by

VAIBHAV SRIVASTAVA
(1613101806/16SCSE101685)

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

JANARTHANAN.S
Professor

APRIL / MAY- 2020

CONTENT

	Page No
CHAPTER 1: ABSTRCT	4
CHAPTER 2: INTRODUCTION	5-12
1) Chatbot and Machine Learning	
2) Artificial Intelligence	
3) AI application	
CHAPTER 3: EXISTING MODEL	13
CHAPTER 4: PROPOSED MODEL	14-18
1) AIML Scripting	
2) Creating a Startup File	
3) Creating Interface	
4) Speeding Up Brain Load	
5) Loading Brain	
CHAPTER 5: IMPLEMENTATION	19-20
CHAPTER 6: RESULT	21-22
CHAPTER 7: CONCLUSION	23
REFERENCES	24

ABSTRACT

A chatbot is a computer software program that conducts a conversation via auditory or textual methods. This software is used to perform tasks such as quickly responding to users, informing them, helping to purchase products and providing better service to customers. Chatbots are programs that work on Artificial Intelligence (AI) & Machine Learning Platform. Chatbot has become more popular in business groups right now as it can reduce customer service costs and handles multiple users at a time. But yet to accomplish many tasks there is a need to make chatbots as efficient as possible. In this project, we provide the design of a chatbot, which provides a genuine and accurate answer for any query using Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA) with python platform

INTRODUCTION

A chatbot is a computer program that can converse with humans using artificial intelligence in messaging platforms. The goal of the project is to add a chatbot feature and API for Yioop. discussion groups, blogs, wikis etc. Yioop provides all the basic features of web search portal. It has its own account management system with the ability to set up groups that have discussions boards. Groups are collections of users that have access to a group feed. The user who creates a group is set as the initial group owner. Posts are grouped by thread in a group containing the most recent activity at the top. The chatbot API for Yioop will allow developers to create new chatbots, powered by rules or artificial intelligence, that can interact like a human with users in a groups feed page. Example chatbots that can be developed with this API is weather chatbots or book flight chatbots. Over past few years, messaging applications have become more popular than Social networking sites. People are using messaging applications these days such as Facebook Messenger, Skype, Viber, Telegram, Slack etc. This is making other businesses available on messaging platforms leads to proactive interaction with users about their products. To interact on such messaging platforms with many users, the businesses can write a computer program that can converse like a human which is called a chatbot.

Chatbots come in two kinds:

- Limited set of rules
- Machine learning

Chatbot that uses limited set of rules

This kind of bots are very limited to set of texts or commands. They have ability to respond only to those texts or commands. If user asks something different or other than the set of texts or commands which are defined to the bot, it would not respond as desired since it does not understand or it has not trained what user asked. These bots are not very smart when compared to other kind of bots.

Chatbot and Machine learning

Machine learning chatbots works using artificial intelligence. User need not to be more specific while talking with a bot because it can understand the natural language, not only commands. This kind of bots get continuously better or smarter as it learns from past conversations it had

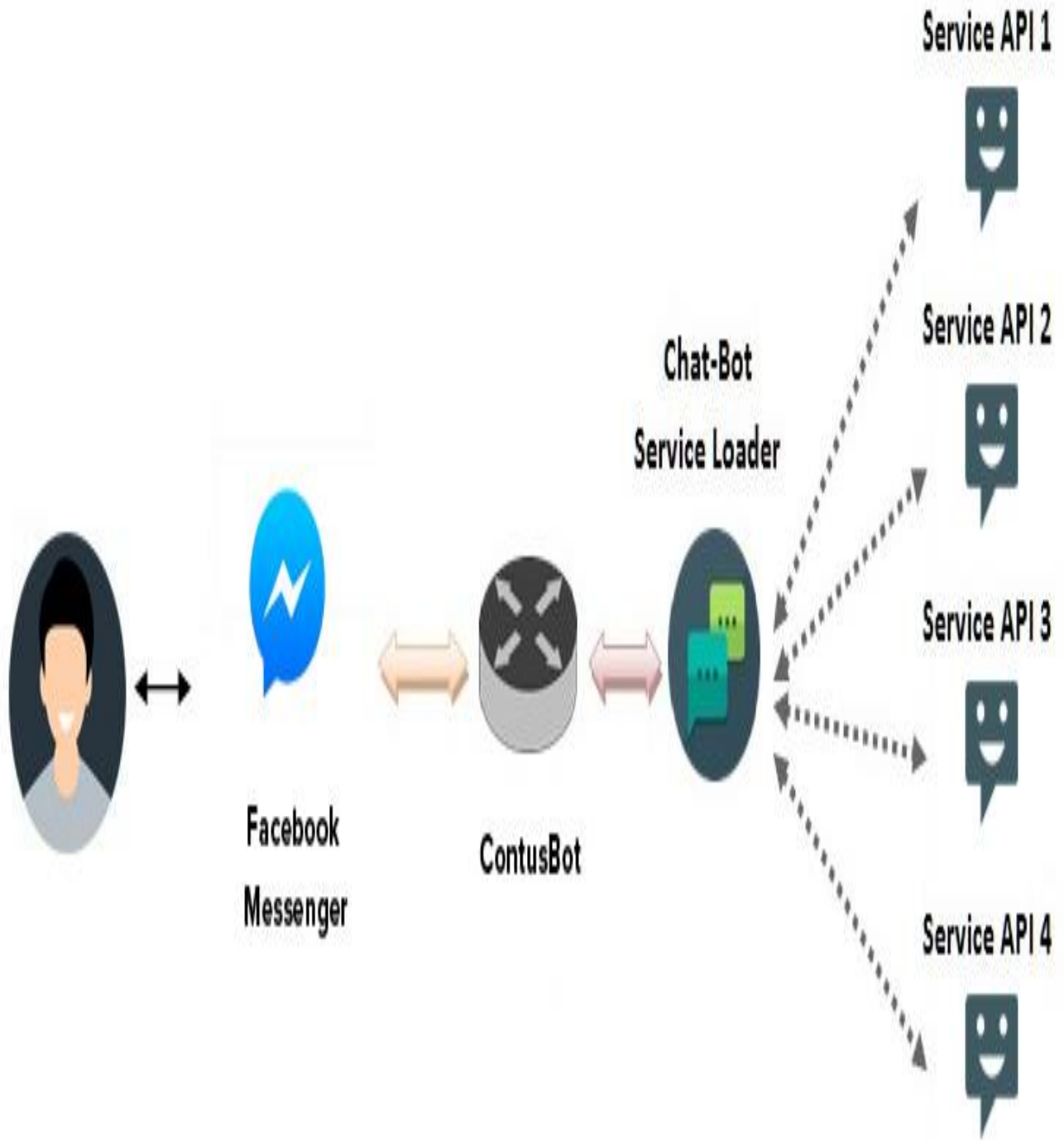
with people.

Here is a simple example which illustrate how they work. The following is a conversation between a human and a chatbot: Human: "I need a flight from San Jose to New York." Bot: "Sure! When would you like to travel?" Human: "From Dec 20, 2016 to Jan 28, 2017." Bot: "Great! Looking for flights."

In order to achieve the ultimate goal, I have taken an iterative approach and divided my work into four major deliverables. These deliverables not only helped me in understanding the code structure of Yioop but also enhances Yioop's functionality. In the rest of the report, I will be discussing about the four deliverables. To understand more on chatbot service, I had implemented a Facebook Messenger Weather Bot in deliverable 1, which is discussed in next section. The purpose of deliverable 2 is to introduce chatbots to the Yioop. I have added Bot Configuration settings which is used to add bot users in Yioop. In the next deliverable, I have added a functionality where the user will be able to call bots in a group thread. Activation of bots will happen by calling respective callback URL which is already configured that helps bots to have a conversation with users. More details on this is discussed in deliverable 3 section. As a deliverable 4, I have created a weather bot i.e, a web application in php that calls yahoo API to get weather information. The last section of the report contains the conclusion and future work.

I have implemented a Facebook Messenger Bot to get an overview of how chatbot is build. During this implementation, I understood the flow of control for a chatbot service with other services which is explained below.

In order to create a Facebook Messenger Bot, a developer needs to be authenticated and approved by Facebook to converse with the public and the web server for security reasons. For a Facebook Messenger Bot, I have created a simple web application using Node.js by installing the necessary dependencies using npm. I ran this locally. I also downloaded and installed ngrok and started it - `npm run ngrok`. This launched a Forwarding URL to the local running server, that means any requests to Forwarding URL will hit the locally running server. This url is used as a Callback URL in Facebook App which will be explained further. To set up the Facebook App, I have created a Facebook Page and Facebook App using my Facebook account. While setting up a Webhook in the app settings, I have given the Forwarding URL as Callback URL and added code for verification. The access token in page settings is stored as environment variable as it will be used in integration. In order to make webhook to receive messages from this page, the app is subscribed to the page created. To set up the bot to handle the POST calls at webhook, I have created a webhook endpoint in the sample application.



I have implemented a Facebook Messenger Bot to get an overview of how chatbot is build. During this implementation, I understood the flow of control for a chatbot service with other services which is explained below.

In order to create a Facebook Messenger Bot, a developer needs to be authenticated and approved by Facebook to converse with the public and the web server for security reasons. For a Facebook Messenger Bot, I have created a simple web application using Node.js by installing the necessary dependencies using npm. I ran this locally. I also downloaded and installed ngrok and started it - `npm run ngrok`. This launched a Forwarding URL to the local running server, that means any requests to Forwarding URL will hit the locally running server. This url is used as a Callback URL in Facebook App which will be explained further. To set up the Facebook App, I have created a Facebook Page and Facebook App using my Facebook account. While setting up a Webhook in the app settings, I have given the Forwarding URL as Callback URL and added code for verification. The access token in page settings is stored as environment variable as it will be used in integration. In order to make webhook to receive messages from this page, the app is subscribed to the page created. To set up the bot to handle the POST calls at webhook, I have created a webhook endpoint in the sample application.

Artificial Intelligence

AI was coined by John McCarthy, an American computer scientist, in 1956 at The Dartmouth Conference where the discipline was born. Today, it is an umbrella term that encompasses everything from robotic process automation to actual robotics. It has gained prominence recently due, in part, to big data, or the increase in speed, size and variety of data businesses are now collecting. AI can perform tasks such as identifying patterns in the data more efficiently than humans, enabling businesses to gain more insight out of their data.

AI can be categorized in any number of ways, but here are two examples.

The first classifies AI systems as either weak AI or strong AI. Weak AI, also known as narrow AI, is an AI system that is designed and trained for a particular task. Virtual personal assistants, such as Apple's Siri, are a form of weak AI.

Strong AI, also known as artificial general intelligence, is an AI system with generalized human cognitive abilities so that when presented with an unfamiliar task, it has enough intelligence to find a solution. The Turing Test, developed by mathematician Alan Turing in 1950, is a method used to determine if a computer can actually think like a human, although the method is

controversial.

The second example is from Arend Hintze, an assistant professor of integrative biology and computer science and engineering at Michigan State University. He categorizes AI into four types, from the kind of AI systems that exist today to sentient systems, which do not yet exist. His categories are as follows:

Types of artificial intelligence

AI can be categorized in any number of ways, but here are two examples.

The first classifies AI systems as either weak AI or strong AI. Weak AI, also known as narrow AI, is an AI system that is designed and trained for a particular task. Virtual personal assistants, such as Apple's Siri, are a form of weak AI.

Strong AI, also known as artificial general intelligence, is an AI system with generalized human cognitive abilities so that when presented with an unfamiliar task, it has enough intelligence to find a solution. The Turing Test, developed by mathematician Alan Turing in 1950, is a method used to determine if a computer can actually think like a human, although the method is controversial.

The second example is from Arend Hintze, an assistant professor of integrative biology and computer science and engineering at Michigan State University. He categorizes AI into four types, from the kind of AI systems that exist today to sentient systems, which do not yet exist. His categories are as follows:

Type 1: Reactive machines.

An example is Deep Blue, the IBM chess program that beat Garry Kasparov in the 1990s. Deep Blue can identify pieces on the chess board and make predictions, but it has no memory and cannot use past experiences to inform future ones. It analyzes possible moves -- its own and its opponent -- and chooses the most strategic move. Deep Blue and Google's AlphaGO were designed for narrow purposes and cannot easily be applied to another situation.

Type 2: Limited memory.

These AI systems can use past experiences to inform future decisions. Some of the decision-making functions in autonomous vehicles have been designed this way. Observations used to inform actions happening in the not-so-distant future, such as a car that has changed lanes. These observations are not stored permanently.

Type 3: Theory of mind.

This is a psychology term. It refers to the understanding that others have their own beliefs, desires and intentions that impact the decisions they make. This kind of AI does not yet exist.

Type 4: Self-awareness. In this category,

AI systems have a sense of self, have consciousness. Machines with self-awareness understand their current state and can use the information to infer what others are feeling. This type of AI does not yet exist.

Examples of AI technology

Automation is the process of making a system or process function automatically. Robotic process automation, for example, can be programmed to perform high-volume, repeatable tasks normally performed by humans. RPA is different from IT automation in that it can adapt to changing circumstances.

Machine learning is the science of getting a computer to act without programming. Deep learning is a subset of machine learning that, in very simple terms, can be thought of as the automation of predictive analytics. There are three types of machine learning algorithms: supervised learning, in which data sets are labeled so that patterns can be detected and used to label new data sets; unsupervised learning, in which data sets aren't labeled and are sorted according to similarities or differences; and reinforcement learning, in which data sets aren't labeled but, after performing an action or several actions, the AI system is given feedback

Machine vision is the science of making computers see. Machine vision captures and analyzes visual information using a camera, analog-to-digital conversion and digital signal processing. It is often compared to human eyesight, but machine vision isn't bound by biology and can be programmed to see through walls, for example. It is used in a range of applications from signature identification to medical image analysis. Computer vision, which is focused on machine-based image processing, is often conflated with machine vision.

Natural language processing (NLP) is the processing of human -- and not computer -- language by a computer program. One of the older and best known examples of NLP is spam detection, which looks at the subject line and the text of an email and decides if it's junk. Current approaches to NLP are based on machine learning. NLP tasks include text translation, sentiment analysis and speech recognition.

Pattern recognition is a branch of machine learning that focuses on identifying patterns in data. The term, today, is dated.

Robotics is a field of engineering focused on the design and manufacturing of robots. Robots are often used to perform tasks that are difficult for humans to perform or perform consistently. They are used in assembly lines for car production or by NASA to move large objects in space. More recently, researchers are using machine learning to build robots that can interact in social settings.

AI applications

AI in healthcare. The biggest bets are on improving patient outcomes and reducing costs. Companies are applying machine learning to make better and faster diagnoses than humans. One of the best known healthcare technologies is IBM Watson. It understands natural language and is capable of responding to questions asked of it. The system mines patient data and other available data sources to form a hypothesis, which it then presents with a confidence scoring schema. Other AI applications include chatbots, a computer program used online to answer questions and assist customers, to help schedule follow-up appointments or aiding patients through the billing process, and virtual health assistants that provide basic medical feedback.

AI in business. Robotic process automation is being applied to highly repetitive tasks normally performed by humans. Machine learning algorithms are being integrated into analytics and CRM platforms to uncover information on how to better serve customers. Chatbots have been incorporated into websites to provide immediate service to customers. Automation of job positions has also become a talking point among academics and IT consultancies such as Gartner and Forrester.

AI in education. AI can automate grading, giving educators more time. AI can assess students and adapt to their needs, helping them work at their own pace. AI tutors can provide additional support to students, ensuring they stay on track. AI could change where and how students learn, perhaps even replacing some teachers.

AI in finance. AI applied to personal finance applications, such as Mint or Turbo Tax, is upending financial institutions. Applications such as these could collect personal data and provide financial advice. Other programs, IBM Watson being one, have been applied to the process of buying a home. Today, software performs much of the trading on Wall Street.

AI in law. The discovery process, sifting through of documents, in law is often overwhelming for humans. Automating this process is a better use of time and a more efficient process. Startups are

also building question-and-answer computer assistants that can sift programmed-to-answer questions by examining the taxonomy and ontology associated with a database.

AI in manufacturing. This is an area that has been at the forefront of incorporating robots into the workflow. Industrial robots used to perform single tasks and were separated from human workers, but as the technology advanced that changed.

EXISTING SYSTEM

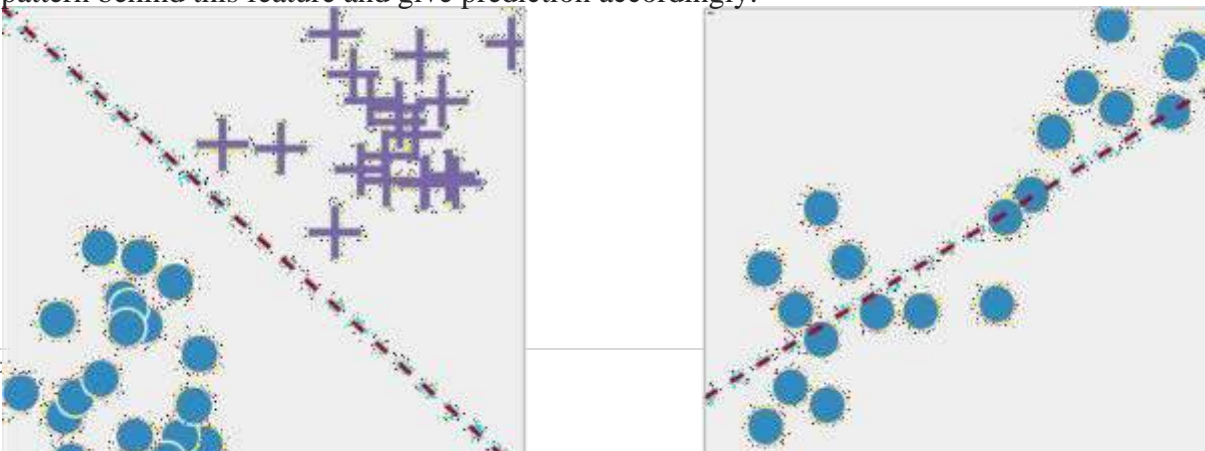
Natural Language Processing

Natural Language Processing (NLP) is the study of letting computers understand human languages[3]. Without NLP, human language sentences are just a series of meaningless symbols to computers. Computers don't recognize the words and don't understand the grammars. NLP can be regarded as a "translator", who will translate human languages to computer understandable information.

Traditionally, users need to follow well-defined procedures accurately, in order to interact with computers. For example, in Linux systems, all commands must be precise. A single replace of one character or even a space can have significant difference. However, the emergence of NLP is changing the way of interacting. Apple Siri and Microsoft Cortana have made it possible to give command in everyday languages and is changing the way of interacting.

Machine Learning.

Machine Learning (ML) is an area of computer science that "gives computers the ability to learn without being explicitly programmed". The parameter of the formulas is calculated from the data, rather than defined by the programmer. Two most common usage of ML is Classification and Regression. As shown in figure1[8], Classification means to categorize different types of data, while Regression means to find a way to describe the data. Basic ML program will have two stages, *fitting* and *predicting*. In the fitting stage, the program will be given a large set (at least thousands) of data. The program will try to adjust its parameter based on some statistical models, in order to make it "fit" the input data best. In the predicting stage, the program will give a prediction for a new input based on the parameters it just calculated out. For example, the famous Iris flower dataset [9] contains the measurement of several features of three different species of flowers, such as the length of sepals and petals. A well-defined ML program can learn the pattern behind this feature and give prediction accordingly.



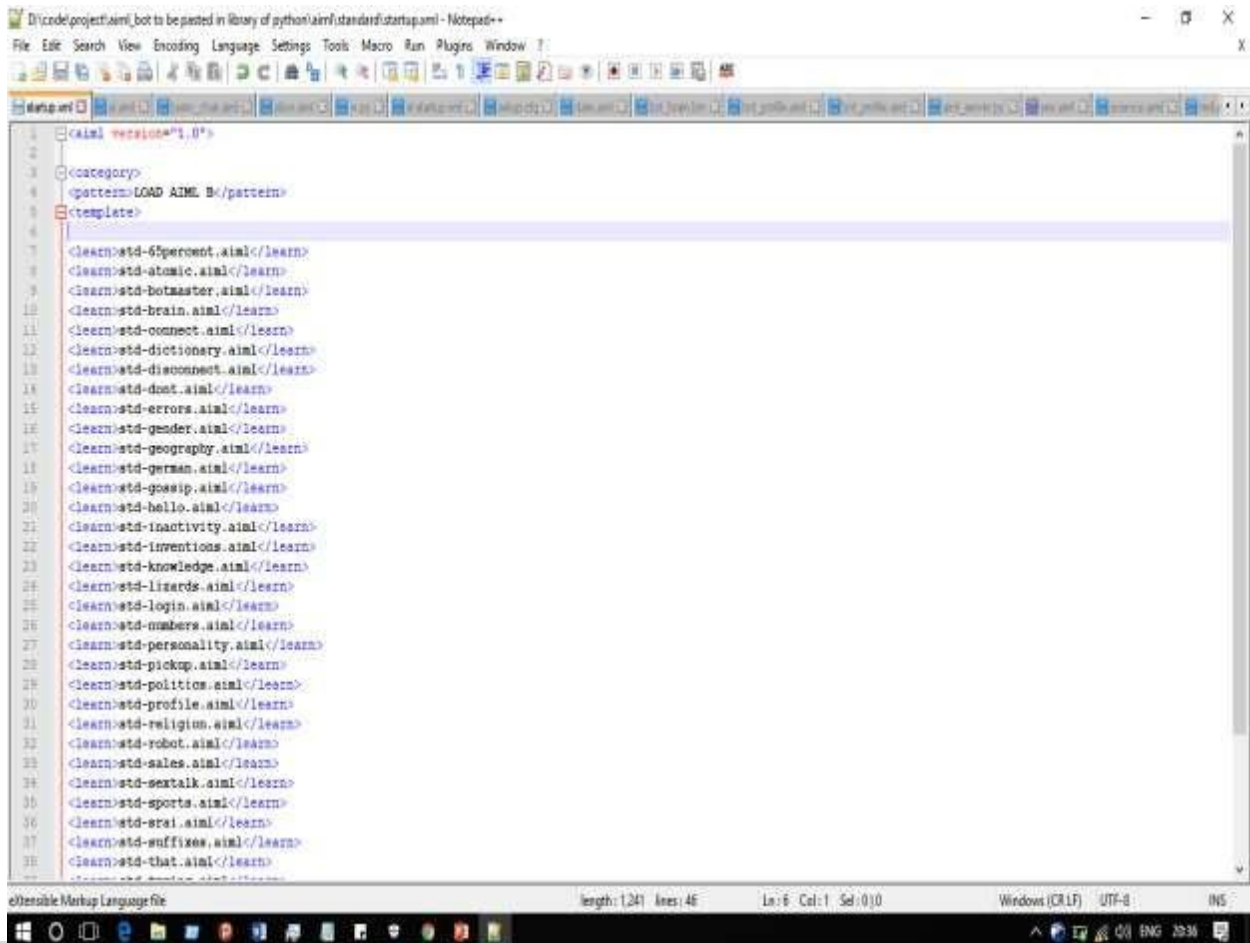
PROPOSED SYSTEM

AIML Scripting

We created the AIML file that only handles one pattern, load aiml b. When we enter that command to the bot, it will try to load basic_chat.aiml. It won't work unless we actually create it. Here is what you can put inside basic_chat.aiml. We will match two basic patterns and respond.

Creating a Startup File-

It is standard to create a startup file called std-startup.xml as the main entry point for loading AIML files. In this case we will create a basic file that matches one pattern and takes one action. We want to match the pattern load aiml b, and have it load our aiml brain in response. We will create the basic_chat.aiml file in a minute.



```
1 <aiml version="1.0">
2
3 <category>
4   <pattern>LOAD AIML B</pattern>
5   <template>
6
7     <learn>std-6percent.aiml</learn>
8     <learn>std-atomic.aiml</learn>
9     <learn>std-botmaster.aiml</learn>
10    <learn>std-brain.aiml</learn>
11    <learn>std-connect.aiml</learn>
12    <learn>std-dictionary.aiml</learn>
13    <learn>std-disconnect.aiml</learn>
14    <learn>std-dot.aiml</learn>
15    <learn>std-errors.aiml</learn>
16    <learn>std-gender.aiml</learn>
17    <learn>std-geography.aiml</learn>
18    <learn>std-german.aiml</learn>
19    <learn>std-gossip.aiml</learn>
20    <learn>std-hello.aiml</learn>
21    <learn>std-inactivity.aiml</learn>
22    <learn>std-inventions.aiml</learn>
23    <learn>std-knowledge.aiml</learn>
24    <learn>std-lizards.aiml</learn>
25    <learn>std-login.aiml</learn>
26    <learn>std-numbers.aiml</learn>
27    <learn>std-personality.aiml</learn>
28    <learn>std-pickup.aiml</learn>
29    <learn>std-politics.aiml</learn>
30    <learn>std-profile.aiml</learn>
31    <learn>std-religion.aiml</learn>
32    <learn>std-robot.aiml</learn>
33    <learn>std-sales.aiml</learn>
34    <learn>std-sextalk.aiml</learn>
35    <learn>std-sports.aiml</learn>
36    <learn>std-srai.aiml</learn>
37    <learn>std-suffixes.aiml</learn>
38    <learn>std-that.aiml</learn>
```

Creating Interface-

As shown in the figure, there are six components in our project. The *interfaces* are the front end chat box for user to talk to the bot, which can be the Bot Portal, Skype, Facebook, etc. The *connector* works as a common gateway for all the interfaces. The outbound side calls different APIs to different front end, but the inbound APIs kept the same for our bot to connect. Fortunately, this connector has already been implemented by the bot framework SDK, we only need to rightly configure them.

The *botpart* contains the main flow control of our project. It is responsible for redirect the input to different models, parse the return values, and determines what to do next. It is also connected to the database to retrieve and update values.



```
wpy - C:\Users\ganit\Desktop\project\wpy (2.7.12)
File Edit Format Run Options Window Help
#:/usr/bin/python

from Tkinter import *
import ai2l

k = ai2l.Kernel()

k.bootstrap(learnFile="st-startup.xml", commands = "load ai2l b")
k.saveBrain("bot_brain.krn")

top = Tk()

w = Entry(top)
w.pack()

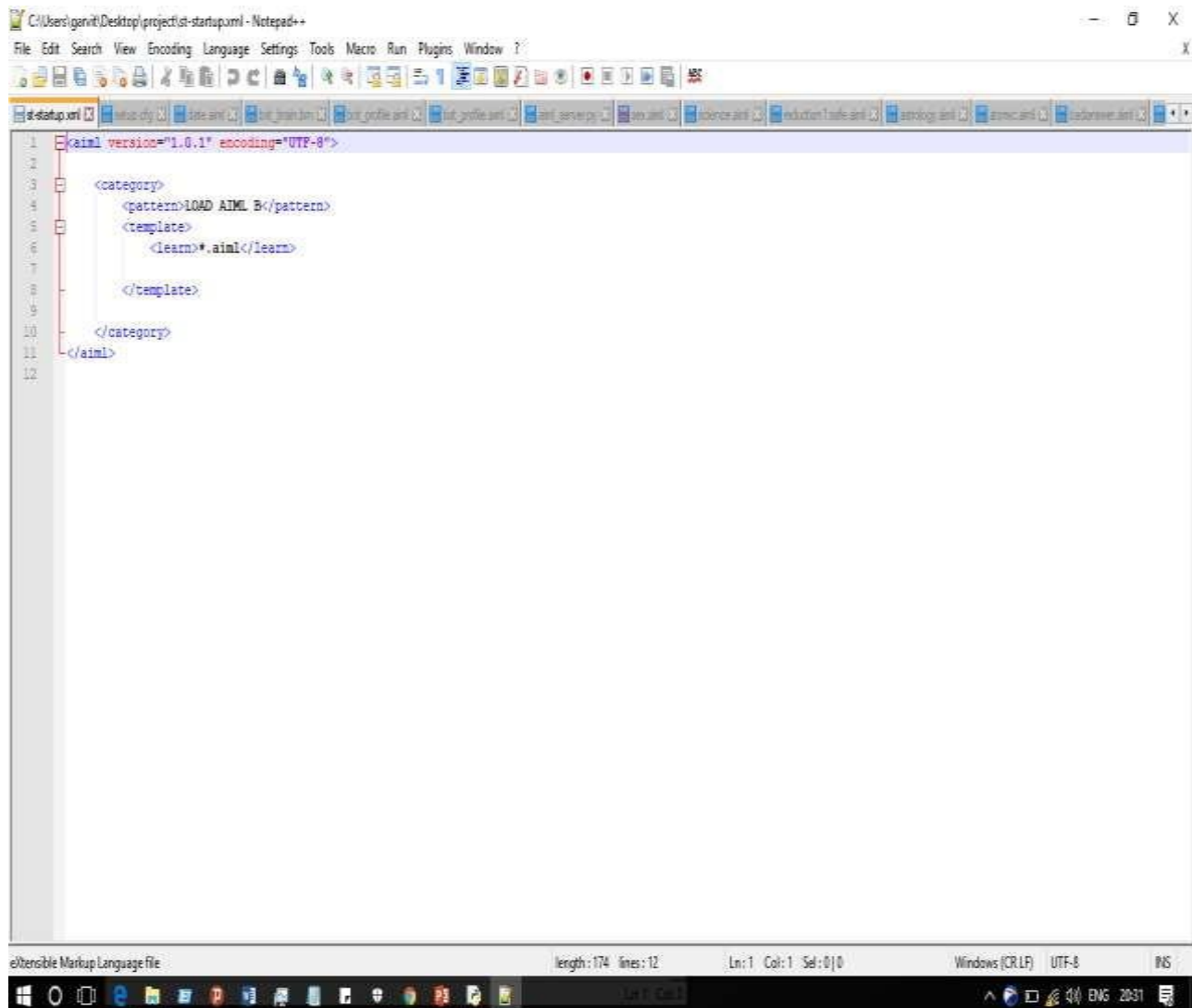
l = Text(top)
l.insert(END, '')
l.pack()

def procc():
    tx = w.get()
    w.delete(0, END)
    w.pack()
    response = k.respond(tx)
    l.insert(END, k+' '+tx)
    l.insert(END, response+'\n')
    l.pack()

b = Button(top, text="chat", command = procc)
b.pack()
top.mainloop()
```

Speeding up Brain Load

When you start to have a lot of AIML files, it can take a long time to learn. This is where brain files come in. After the bot learns all the AIML files it can save its brain directly to a file which will drastically speed up load times on subsequent runs.

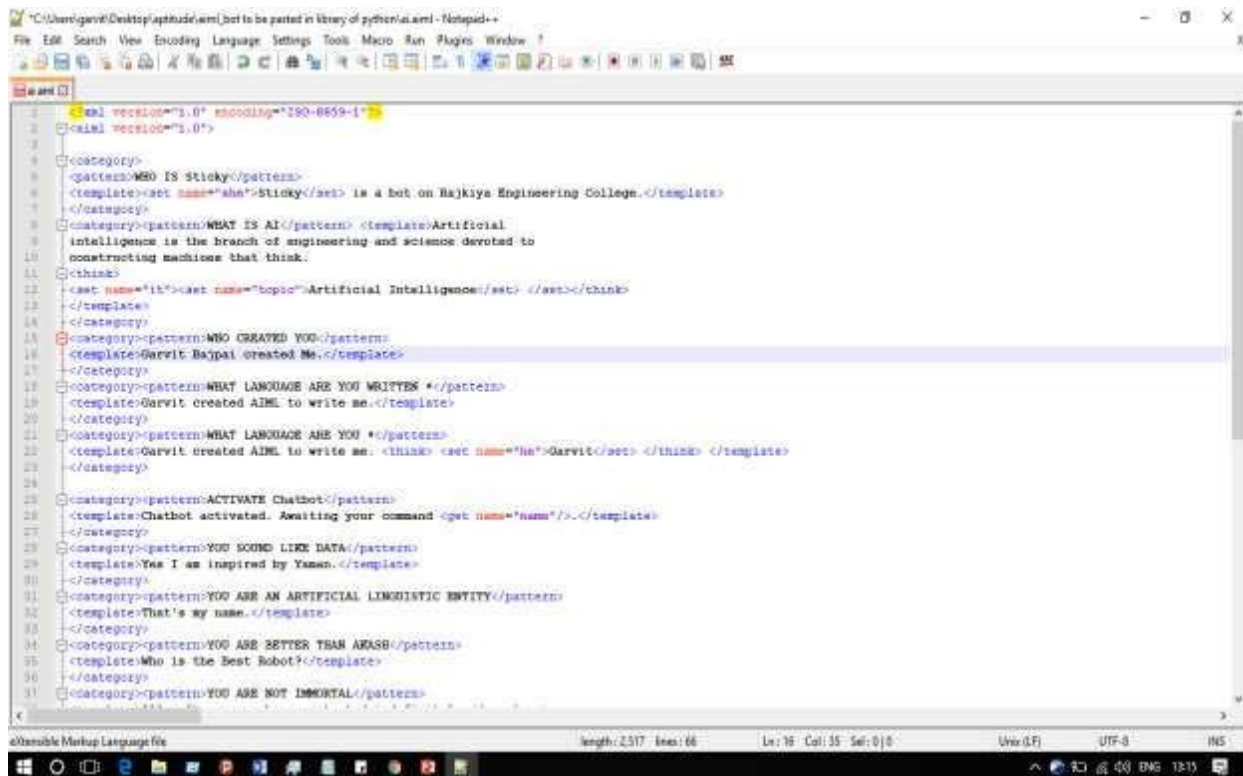


```
1 <aiml version="1.0.1" encoding="UTF-8">
2
3   <category>
4     <pattern>LOAD AIML B</pattern>
5     <template>
6       <learn>*.aiml</learn>
7
8     </template>
9
10  </category>
11 </aiml>
12
```

The screenshot shows a Notepad++ window with the following text: `<aiml version="1.0.1" encoding="UTF-8">`, `<category>`, `<pattern>LOAD AIML B</pattern>`, `<template>`, `<learn>*.aiml</learn>`, `</template>`, `</category>`, and `</aiml>`. The status bar at the bottom indicates the file is an eXtensible Markup Language file, with a length of 174 bytes and 12 lines. The current line is 1, column is 1, and selection is 0 characters.

Loading Brain

This is the simplest program we can start with. It creates the aiml object, learns the startup file, and then loads the rest of the aiml_files. After that, it is ready to chat, and we enter an infinite loop that will continue to prompt the _user for a message. You will need



```
1 <!--!xml version="1.0" encoding="ISO-8859-1" -->
2 <!--!xml version="1.0" -->
3
4 <category>
5 <pattern>WHO IS Sticky</pattern>
6 <template><set name="sh">Sticky</set> is a bot on Rajkiya Engineering College.</template>
7 </category>
8 <category><pattern>WHAT IS AI</pattern> <template>Artificial
9 intelligence is the branch of engineering and science devoted to
10 constructing machines that think.
11 </template>
12 <think>
13 <set name="it"><set name="topic">Artificial Intelligence</set> </set></think>
14 </category>
15 <category><pattern>WHO CREATED YOU</pattern>
16 <template>Garvit Baipal created Me.</template>
17 </category>
18 <category><pattern>WHAT LANGUAGE ARE YOU WRITING *</pattern>
19 <template>Garvit created AIML to write me.</template>
20 </category>
21 <category><pattern>WHAT LANGUAGE ARE YOU *</pattern>
22 <template>Garvit created AIML to write me. <think> <set name="he">Garvit</set> </think> </template>
23 </category>
24
25 <category><pattern>ACTIVATE Chatbot</pattern>
26 <template>Chatbot activated. Awaiting your command <get name="name"/>.</template>
27 </category>
28 <category><pattern>YOU SOUND LIKE DATA</pattern>
29 <template>Yes I am inspired by Yaman.</template>
30 </category>
31 <category><pattern>YOU ARE AN ARTIFICIAL LINGUISTIC ENTITY</pattern>
32 <template>That's my name.</template>
33 </category>
34 <category><pattern>YOU ARE BETTER THAN ARASH</pattern>
35 <template>Who is the Best Robot?</template>
36 </category>
37 <category><pattern>YOU ARE NOT IMMORTAL</pattern>
```

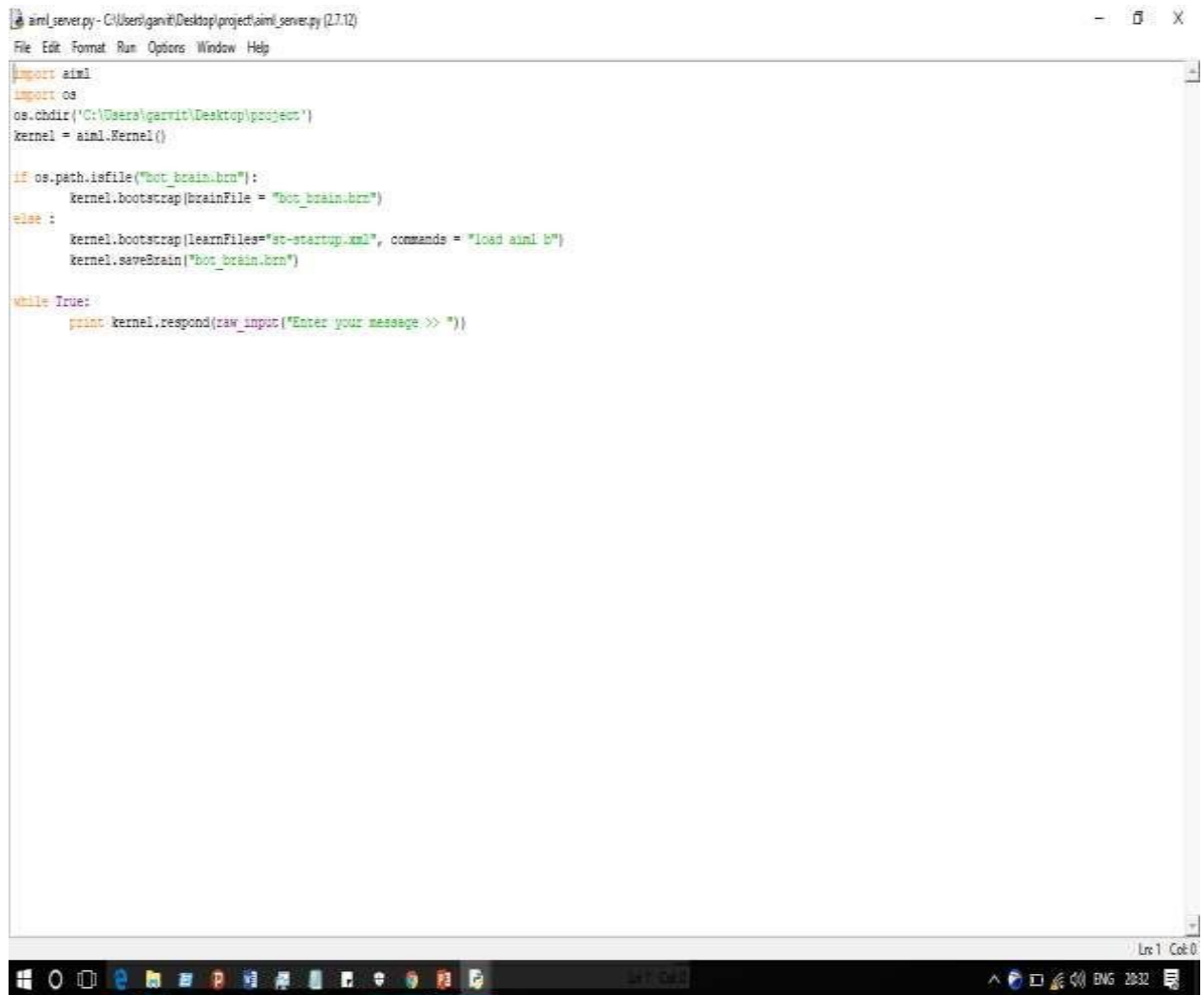
to enter a pattern the bot recognizes. The patterns recognized depend on what AIML files you loaded. We create the startup file as a separate entity so that we can add more aiml files to the bot later without having to modify any of the programs source code. We can just add more files to learn in the startup xml file.

```
ami_server.py - C:\Users\garvit\Desktop\project\ami_server.py (2.7.12)
File Edit Format Run Options Window Help
import ai1
import os
os.chdir('C:\Users\garvit\Desktop\project')
kernel = ai1.Kernel()

if os.path.isfile("bot_brain.brn"):
    kernel.bootstrap(brainFile = "bot_brain.brn")
else :
    kernel.bootstrap(learnFiles="st-startup.xml", commands = "load ai1 b")
    kernel.saveBrain("bot_brain.brn")

while True:
    print kernel.respond(raw_input("Enter your message >> "))
```

Ln 1 Col 0

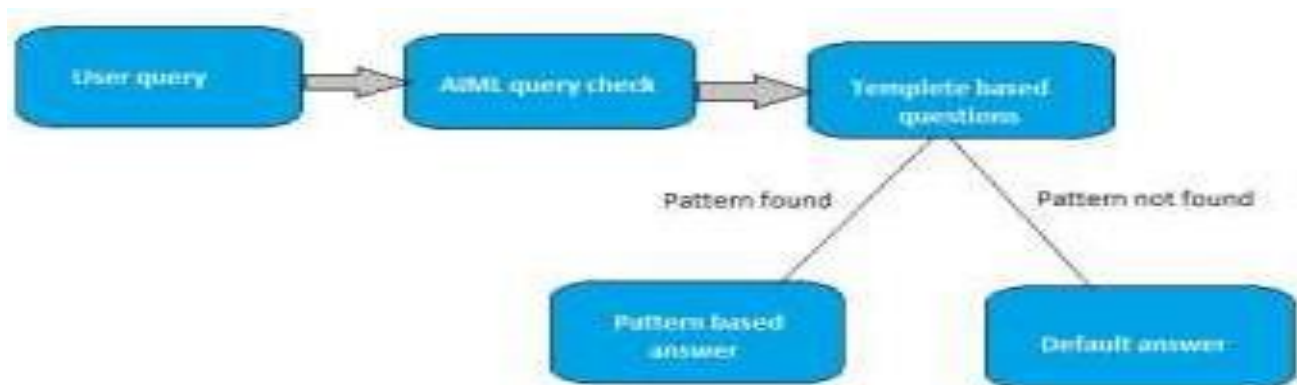


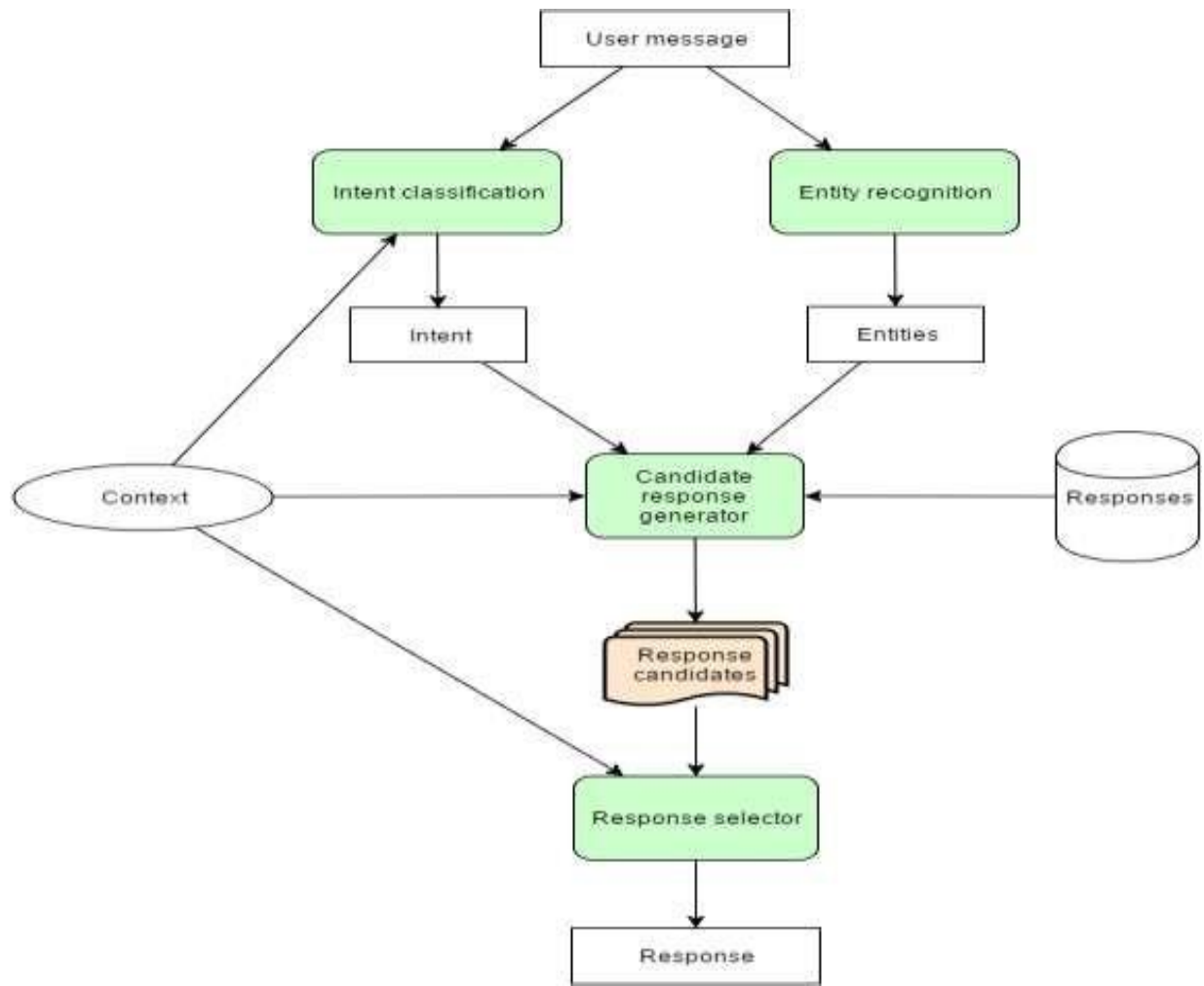
IMPLEMENTATION

This section covers the design and implementation of a different module of the bot, which contains the design of the PYTHON module, the Translator API and the AIML module of domain. Using APIs like Weather, Sports, News and Government Services, the chatbot will be able to answer the questions outside of its dataset and which are currently happening in the real world.

The next step towards building chatbots involves helping people to facilitate their work and interact with computers using natural language or using their set of rules. Future Such chatbots, backed by machine-learning technology, will be able to remember past conversations and learn from them to answer new ones. The challenge would be conversing with the various multiple bot users and multiple users.

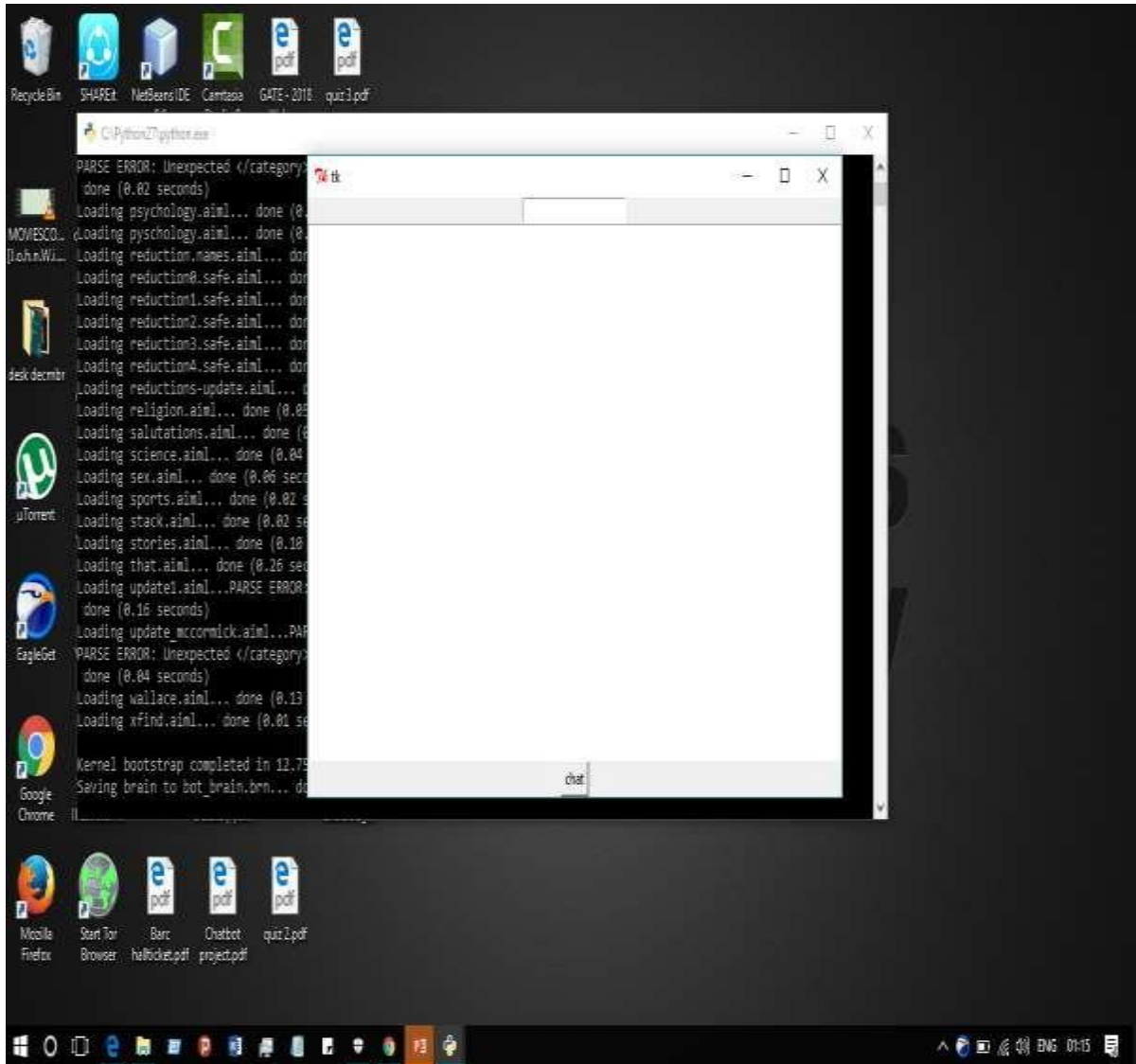
As future work, we can make a chatbot that is based on AIML and LSA. This technology will enable a client to interact with a chatbot in a more natural fashion. We can enhance the discussion by including and changing patterns and templates for general client queries using AIML and the right response are given more often than LSA.





RESULT

Result of the implementation of different modules in python using machine learning and artificial Intelligence.



The screenshot shows a Windows desktop environment. A window titled "Python2\python.exe" is open, displaying the output of a script. The output consists of a list of AIML modules being loaded, each followed by "done" and a time duration. The modules include: psychology.aiml, reduction.names.aiml, reduction0.safe.aiml, reduction1.safe.aiml, reduction2.safe.aiml, reduction3.safe.aiml, reduction4.safe.aiml, reductions-update.aiml, religion.aiml, salutations.aiml, science.aiml, sex.aiml, sports.aiml, stack.aiml, stories.aiml, that.aiml, update1.aiml, update_mccormick.aiml, wallace.aiml, and xfind.aiml. There are also "PARSE ERROR: Unexpected </category>" messages interspersed among the loading logs. At the bottom of the window, it says "Kernel bootstrap completed in 12.75" and "Saving brain to bot_brain.brn...". The desktop background is dark, and the taskbar at the bottom shows various application icons and the system tray with the date and time "01:15".

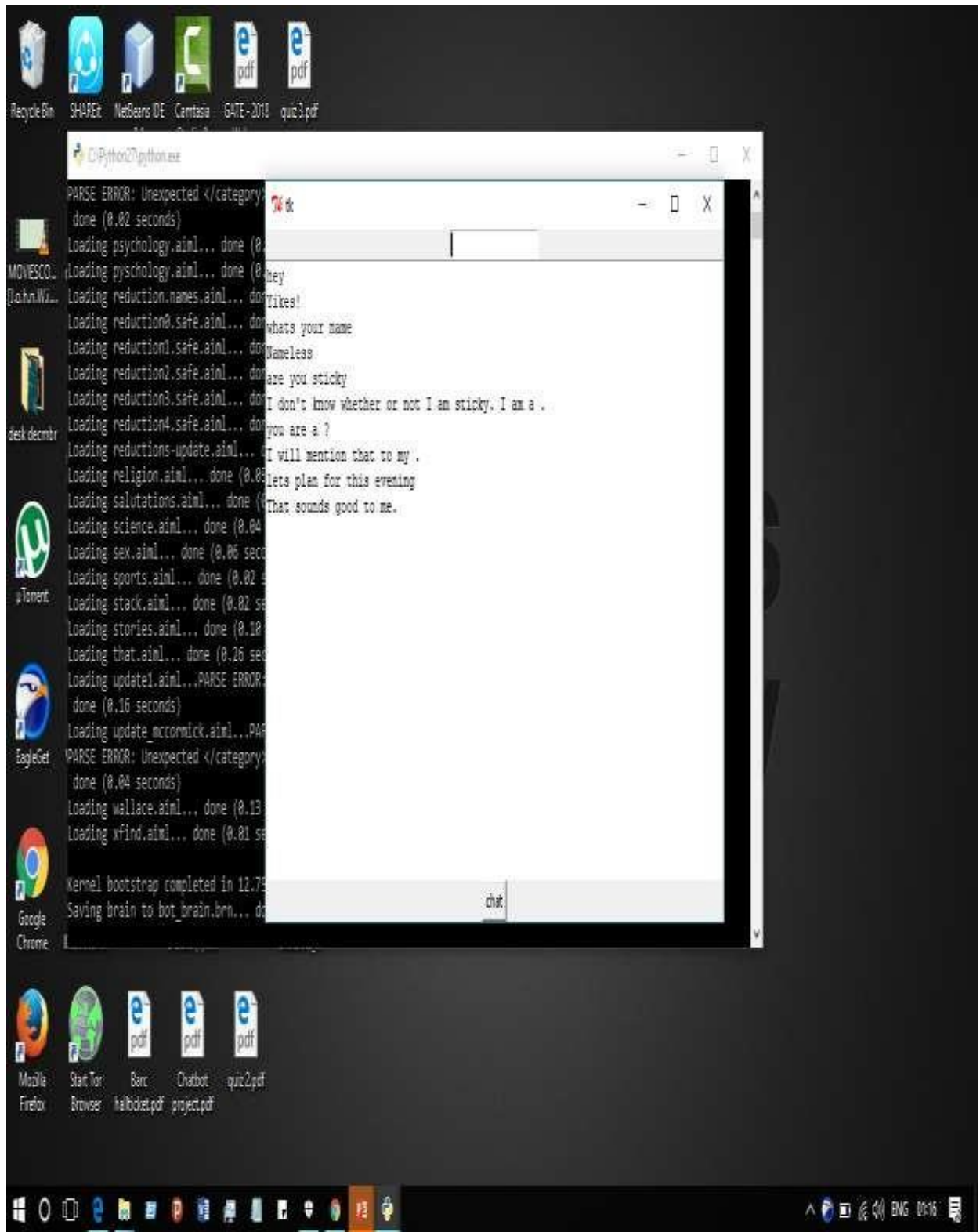


Fig 6.4 Chatbot Conversation

CONCLUSION

Chatbots are the new Apps! As we have discussed in the above deliverables, this project brings the power of chatbots to Yioop and enriches its usability. Chatbots in Yioop can give a human like touch to some aspects and make it an enjoyable conversation. And they are focused entirely on providing information and completing tasks for the humans they interact with. The above mentioned functionality in all the deliverables is implemented and pushed in to Yioop code. By implementing the above mentioned deliverables I was able to add a basic chatbot functionality in to the Yioop. I.e., configuring and creating accounts for bot users with bot settings which is mentioned in deliverable 2, activating a bot whenever a user asks for it via post in a thread which is discussed in deliverable 3 and as I discussed in deliverable 4, I have implemented a simple weather chatbot that gives weather information whenever a user ask and Fig. 3 tells that I was also able to converse with the bot in Yioop. I intend to enhance the system developed so far in CS298. Next step towards building chatbots involve helping people to facilitate their work and interact with computers using natural language or using set of rules. Future Yioop chatbots, backed by machine-learning technology, will be able to remember past conversations and learn from them to answer new ones. The challenge would be conversing with multiple bot users and multiple user

REFERENCES

1. Bayan Abu Shawar and Eric Atwell, 2007 "Chatbots: Are they Really Useful?"
2. LDV Forum - GLDV Journal for Computational Linguistics and Language Technology.
3. http://www.ldv-forum.org/2007_Heft1/Bayan_Abu-Shawar_and_Eric_Atwell.pdf
4. Bringing chatbots into education: Towards natural language negotiation of open learner models. Know.-Based Syst. 20, 2 (Mar. 2007), 177-185.
5. Intelligent Tutoring Systems: Prospects for Guided Practice and Efficient Learning. Whitepaper for the Army's Science of Learning Workshop, Hampton, VA. Aug 1-3, 2006.
6. <http://en.wikipedia.org/wiki/Chatterbot>
ALICE. 2002. *A.L.I.C.E AI* Foundation, <http://www.alicebot.org/>

