MINI PROJECT

(2020-2021)

INTELLIGENT CHATBOT

REPORT

GLA University, Mathura

Department of Computer Science and Engineering

Submitted in partial fulfilment of the requirements for the Degree of Bachelor Technology

Team Members

MANISH SHARMA (181500368)

SHIVAM YADAV (181500676)

RAKESH YADAV (181500551)

Supervised By

Mr. Mandeep Singh

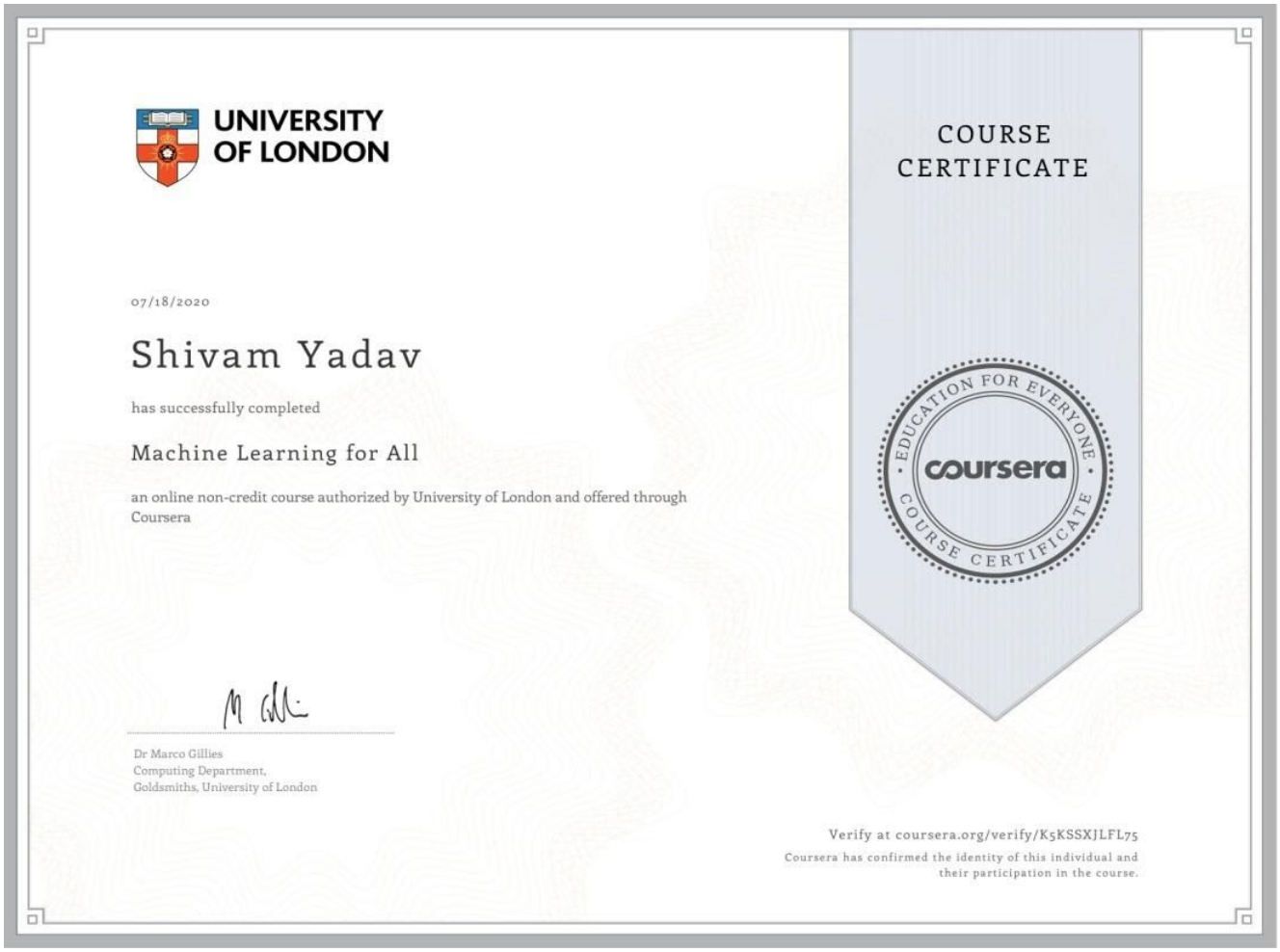
(Assistant Professor)

DECLARATION

We hereby declare that we carried out the work reported in this project under the department of Computer Science and Engineering, GLA University, Mathura under the supervision of Mr. Mandeep Singh (Assistant Professor).We solemnly declare that to best our knowledge, no part of this report has been submitted here or elsewhere in a previous application for a third year project. All sources of knowledge used have been duly acknowledged.

……..	……..	……...
Manish Sharma	Shivam Yadav	Rakesh Yadav
1815000368	181500676	181500551
B.TECH (CSE)	B.TECH (CSE)	B.TECH (CSE)
3rd year (V SEM.)	3rd year (V SEM.)	3rd year (V SEM.)

CERTIFICATEs



UNIVERSITY OF LONDON

COURSE CERTIFICATE

07/18/2020

Shivam Yadav

has successfully completed

Machine Learning for All

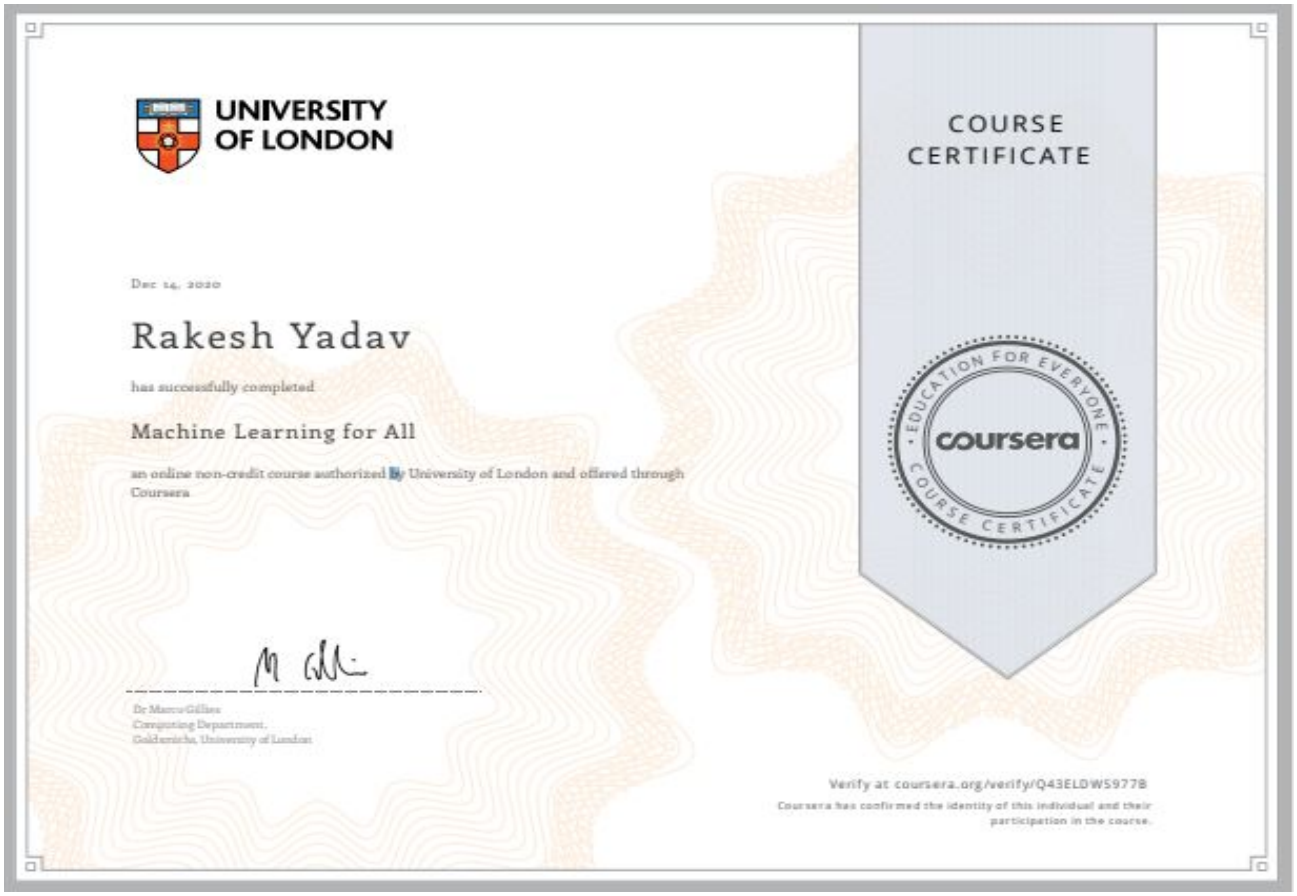
an online non-credit course authorized by University of London and offered through Coursera

EDUCATION FOR EVERYONE · coursera · COURSE CERTIFICATE

Dr Marco Gillies
Computing Department,
Goldsmiths, University of London

Verify at coursera.org/verify/K5KSSXJLFL75

Coursera has confirmed the identity of this individual and their participation in the course.


UNIVERSITY OF LONDON
COURSE CERTIFICATE
Dec 14, 2020
Rakesh Yadav
has successfully completed
Machine Learning for All
an online non-credit course authorized by University of London and offered through Coursera
Dr Marco Gillies
Computing Department,
Goldsmiths, University of London
EDUCATION FOR EVERYONE
coursera
COURSE CERTIFICATE
Verify at coursera.org/verify/Q43ELDWS977B
Coursera has confirmed the identity of this individual and their participation in the course.


Certificate of Completion
This is to certify that Manish Sharma successfully completed 22 total hours of 2021 Data Science & Machine Learning with R from A-Z Course online course on Nov. 26, 2020
Juan E. Galvan, Instructor
Ismail Tigrek, Instructor
&
Udemy
Certificate no: UC-4a183e0c-4fc9-4fd8-b788-27fbf1327461
Certificate url: ude.my/UC-4a183e0c-4fc9-4fd8-b788-27fbf1327461
Version 3
#BeAble

ACKNOWLEDGEMENT

This project itself is acknowledgement for all those people who have given us their heartfelt co-operation in making this project a grand success. We would like to express our special thanks of gratitude to our teacher Mr. Mandeep Singh (Assistant Professor) of GLA UNIVERSITY, MATHURA for providing valuable guidance at every stage of this project work.

We would also like to thank to all teachers of the department of Computer Science And Engineering who gave us the golden opportunity to do this wonderful project on the topic Intelligent Chatbot using Python and Machine Learning which will also help us in doing a lot of research and we will come to know about so many new things.

We would like to express our deep sense of gratitude and earnest thanks giving to our dear parents for their moral support and heartfelt cooperation in doing the main project. Finally, we would like to thank all those friends who are involved directly and indirectly in completion of our project.

ABSTRACT

Chatbot is widely popular now-a-days and catching speed as an application of computer communication. Some programs respond intelligently like human. This type of program is called a Chatbot. This paper addresses the design and implementation of a Chatbot system. We will also study another application where Chatbots could be useful and techniques used while designing a Chatbot. A chatbot is a conversational agent where a computer program is designed to simulate an intelligent conversation. It can take user input in many formats like text, voice, sentiments, etc. For this purpose, many open source platforms are available. Artificial Intelligence Markup Language (AIML) is derived from Extensible Markup Language (XML) which is used to build up a conversational agent (chatbot) artificially.

The goal of the project is to make a fully functional chatbot. Chatbots, powered by rules or artificial intelligence/machine learning, that can interact like a human with users in real life. Weather chatbots or book flight chatbots are the real life examples that are developed with this language and machine learning.

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, 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 as a new information, communication and transaction channel enable businesses to reach their target audience through messenger apps like Facebook, WhatsApp or WeChat. Compared to traditional chats, chatbots are not handled by human persons, but software is leading through conversations. Latest chatbots developments in customer services and sales are remarkable. However, in the field of public transport, little research has been published on chatbots so far. With chatbots, passengers find out timetables, buy tickets and have a personal, digital travel advisor providing real-time and context-relevant information about trips. Chatbots collect and provide different data about users and their journey in public transportation systems.

TABLE OF CONTENT

CHAPTER - 1

INTRODUCTION

1.1 Overview

As its title suggests Chatbot is software designed to interact with humans in their natural language through messaging applications, websites, mobile apps. The chatbot is often considered as the most promising and advanced method for interaction between humans and machines. These chatbots use audio as well as textual ways to converse with people or to communicate with humans in a human-like manner.

Chatbots work as a virtual assistant. It is predicted that by 2020, chatbots will handle nearly 85% of the customer-brand interaction. The Chatbots answer the base of the question on the set of predefined rules and instructions. We use Machine Learning to train chatbot and make chatbot more flexible so that it can handle simple as well as complex conversations.The aim of this project is that everyone gets to understand the real time software development environment.

This mini-project helps in understanding the core concept of automations through implementation of ML concept and using knowledge of python to solve real-time problems.

Chatbots come in two kinds:

• Limited set of rules
• Machine learning

1. Chatbot that uses a limited set of rules.This kind of bots are very limited to a set of texts or commands. They have the ability to respond only to those texts or commands. If a 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 the user asked. These bots are not very smart when compared to other kinds of bots.

2. Chatbot and Machine learning Machine learning chatbots works using artificial intelligence. Users 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.

We name this intelligent chatbot as **Technobot** which will do work for personal assistants like Google Assistant, Siri etc.

1.2 Significance and Motivation

The current interest in chatbots is spurred by recent developments in artificial intelligence (AI) and machine learning. Major Internet companies such as Google, Facebook, and Microsoft see chatbots as the next popular technology; Microsoft CEO Satya Nadella said, "Chatbots are the new apps". In Spring 2016, Facebook and Microsoft provided resources for creating chatbots to be integrated into their respective messaging platforms, Messenger and Skype. One year later, more than 30,000 chatbots have been launched on Facebook Messenger. Other messaging platforms have also seen a substantial increase in chatbots, including Slack, Kik, and Viber. Chatbots are seen as a means for direct user or customer engagement through text messaging for customers.

However, it is not simple to transition from established user interfaces, such as web pages and apps, to chatbots as a common means of interacting with data and services.

For example, there is a lack of knowledge regarding how customers react to the substitution of human customer service personnel with chatbots or how the presence of chatbots in online social networks affects multi-party conversations and the spread of information.

Since the initial optimism regarding the launch of chatbots by Microsoft and Facebook, a number of commentators have noted that users' adoption of available chatbots is less substantial than hoped. This could be explained by the fact that most available chatbots fail to fill users' needs due to unclear purposes, nonsensical responses, or insufficient usability.

1.3 Aim and Objective

2 A project objective describes the desired results of a project, which often includes a tangible item. An objective is specific and measurable, and must meet time, budget, and quality constraints.

Objectives can be used in project planning for business, government, nonprofit organizations, and even for personal use (for example, in resumes to describe the exact position a job-seeker wants). A project may have one objective, many parallel objectives, or several objectives that must be achieved sequentially. To produce the most benefit, objectives must be defined early in the project life cycle, in phase two, the planning phase.

In the future, Chatbots will play a vital role in every field whether it will be IT or corporate world.Some of the benefits of chatbots are-

• As we know Chatbots streamlines interactions between people and services, So it can work as enhancing the customer experiences.

• It also offers companies new opportunities to improve the customer interaction process.

• It can also reduce customer service costs.

• Provide an easier approach to global markets.

• Make customer service available 24/7.

• Think about Siri, Alexa, Google Assistant. Aren't these just wonderful?

1.4 Project Scope

Chatbots seem somehow humane now.

Intelligent enough to understand the patterns and put across answers that are appropriate and relevant, chatbots have come a long way. With efficient chatbot development practices, they can be made capable of literally engulfing and processing whatever information comes their way. They learn and develop a predictive analytical capability just like humans.

Simple chatbots were capable of matching a text string and offering an answer only when the exact match is found. When we said chatbots have come a long way, we actually meant it. The advanced chatbots today have a learning curve powered by artificial intelligence and are leading them to be of great significance.

Role of Chatbots in modern era

- **24x7 time availability:-** The availability of chatbots 24/7 with the immense knowledge they can hold is all set to outperform humans. With speed and accuracy, they are offering support to enterprises, they will soon augment human capabilities.

  Users love to interact with chatbots as it saves them time and in most cases offers them clear and concrete answers. They may not be perfect but they are scary close to be perfect.

- **Chatbots are replacing apps:-** The world has seen almost 6.5 million apps developed. It has been identified that 23% of the users uninstall the apps after a few weeks of use. Chatbots come across as a potential way to engage with the audience then. A messaging app brings along a chatbot that is easy to download and users get engaged with the campaigns quickly.
- **Healthcare**:- Many times patients need a quick piece of advice. They would not wish to take the pain of taking an appointment and wait for days to get in touch with the healthcare professional.

  Chatbots could make things easier. Personalized chatbots would have the patient history stored and would understand the queries of the patient. The queries would be answered as per the history and current situation.

1.5 Methodology

Chatbots are intelligent agents with which users can hold conversations, usually via text or voice. In recent years, chatbots have become popular in businesses focused on client service. Despite an increasing interest for chatbots in education, clear information on how to design them as intelligent tutors has been scarce. This paper presents a formal methodology for designing and implementing a chatbot as an intelligent tutor for a university level course.

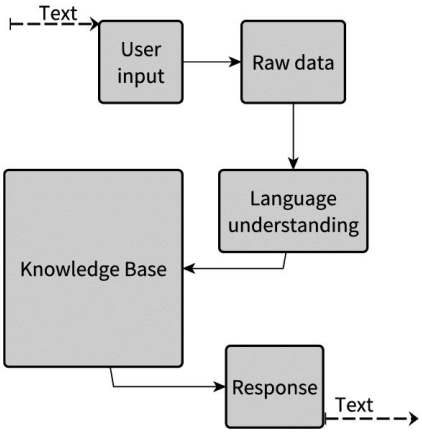


Figure 1: Proposed Methodology

The methodology is built upon first-order logic predicates which can be used in different commercially available tools, and focuses on two phases: knowledge abstraction and modeling, and conversation flow. As the main result of this research, we propose mathematical definitions to model conversation elements, reasoning processes and conflict resolution to formalize the methodology and make it framework-independent.

CHAPTER – 2

LITERATURE REVIEW

The chatbots used by the telecom and marketing sectors for customer service are scripted types of chatbots. They help the customers on some predefined customer care questions. Research is being carried out in making the conventional monotonous chatbots to be communicative, responsive and carry out the communication in a natural (conversational) language. This requires the inclusion of NLP and ML techniques in the system. There are a number of ways to do so. Selection of an appropriate method is based on the domain of the chatbot, the functionalities it intends to provide, the language of communication, the end user, etc. All the above-mentioned issues need to be considered while implementing a chatbot system.

A review on chat interface

This unit is the front end of the system. It is responsible for collecting the user queries from the user which are the input to the system. It is also responsible for displaying the system generated results to the user. Therefore, it can be said that the chat interface is the face of the system through which the entire communication takes place. It is the mediator of conversation between the system and the user. The query that the user fires on the chat interface is passed on to the chatting backend which acts as a message delivering system between the Chat interface and the Machine Learning Layer. This interface can be accessed either as a website or as a smart phone app. The type of interface depends on the requirements of the user that are to be satisfied by the system. If the system is accessed from a smartphone, the interface will be in the form of an app and if the system is accessed from a website, then the interface will be in the form of a website. For building apps on the smartphone, it will require to use android for android phones or Swift for iOS. In this case, only the interfacing platform will be programmed on android and the complete backend processing of the system will take place on a server on which the system will be deployed.

A review on word tokenization

Segmentation, also referred to as tokenization, is the process of splitting text into smaller and meaningful units. These units could be paragraphs, sentences, clauses, phrases, words or letters. The smallest unit are the letters. Word segmentation is the splitting of sentences into individual words separated by blank spaces. The tokenized units of the sentences are called as tokens. The tokenizers split the sentences into words and punctuations marks as independent units. The most commonly used tokenizer is of space type, i.e. it splits the sentences into words at the blank spaces. It is also required that the tokenizer should consider abbreviations, acronyms, dates, numbers in decimal formats, etc., which cannot split at punctuations and blank spaces, as they will lose their meaning if done so.

CHAPTER – 3

SYSTEM SPECIFICATIONS

3.1 Hardware Requirements

Computer system with minimum requirements (At Client Side):

- Processor: Any Processor x86 or x64 supportive to software required.
- Disk Space: .5- 1 GB
- RAM: 512 MB

3.2 Software Requirements

3.2.1 Pycharm

PyCharm is one of the most popular Python IDEs. There are a multitude of reasons for this, including the fact that it is developed by JetBrains, the developer behind the popular IntelliJ IDEA IDE that is one of the big 3 of Java IDEs and the "smartest JavaScript IDE" WebStorm. Having the support for web development by leveraging Django is yet another credible reason.

There are a galore of factors that make PyCharm one of the most complete and comprehensive integrated development environments for working with the Python programming language.

Before proceeding further into exploring the know-how of PyCharm i.e., features, installation, and pros & cons, let's first get a brief introduction to PyCharm. Available as a cross-platform application, PyCharm is compatible with Linux, macOS, and Windows platforms. Sitting gracefully among the best Python IDEs, PyCharm provides support for both Python 2 (2.7) and Python 3 (3.5 and above) versions.

PyCharm comes with a plethora of modules, packages, and tools to hasten Python development while cutting-down the effort required to do the same to a great extent, simultaneously. Further, PyCharm can be customized as per the development requirements, and personal preferences call for. It was released to the public for the very

first time back in February of 2010. In addition to offering code analysis, PyCharm features:

- A graphical debugger
- An integrated unit tester
- Integration support for version control systems (VCSs).

Usage

The main reason Pycharm for the creation of this IDE was for Python programming, and to operate across multiple platforms like Windows, Linux, and macOS. The IDE comprises code analysis tools, debugger, testing tools, and also version control options. It also assists developers in building Python plugins with the help of various APIs available. The IDE allows us to work with several databases directly without getting it integrated with other tools. Although it is specially designed for Python, HTML, CSS, and Javascript files can also be created with this IDE. It also comes with a beautiful user interface that can be customized according to the needs using plugins.

Features

1. Intelligent Code Editor

2. Availability of Integration Tools

3.Google App Engine

4. Integrated Debugging and Testing

5. Multi-technology Development

6. Project and Code Navigation

7. Refactoring

8. Remote Development

Other Features

- Code generation for generating language-specific code constructs.

- Code reference information for instantly accessing API documentation, hints on using various programming entities, etc
- File templates for creating scripts, stub classes, etc
- Import assistance for importing missing libraries
- Intention actions and quick fixes for optimizing code
- Language-specific tools for developing, running, testing, and deploying applications
- Language injections to work with supported languages inside attributes, tags, or string literals
- Live templates for expanding abbreviations into complicated code constructs

Installing and Setting Up PyCharm

Memory 4GB
Storage Space - 2.5GB (main) + 1GB (caches)
Resolution - 1024x768
OS - 64-bit version of macOS 10.11/Microsoft Windows 7 SP1/any Linux distribution supporting Gnome, KDE, or Unity DE

3.2.2 GIThub

GitHub is a Git repository hosting service that provides a web-based graphical interface. It is the world's largest coding community, and putting a code or a project out there brings increased, widespread exposure to your code. You can find source code in many different programming languages and keep track of all changes. Programmers use the command-line interface, Git, to make changes.

GitHub helps every team member work together on a project from any location while facilitating collaboration. You can also review previous versions at any previous point in time.

The Git version control system, as the name suggests, is a system that records all the modifications made to a file or set of data, so that a specific version may be called up

later if needed. The system makes sure that all the team members are working on the file's latest version, and everyone can work simultaneously on the same project.

Features

1. Easy project management

GitHub is a place where project managers and developers coordinate, track, and update their work so that projects are transparent and stay on schedule.

2. Increased safety with packages

Packages can be published privately, within the team, or publicly to the open-source community. The packages can be used or reused by downloading them from GitHub.

3. Effective team management

GitHub helps all the team members stay on the same page and organized. Moderation tools like Issue and Pull Request Locking help the team to focus on the code.

4. Improved code writing

Pull requests help the organizations to review, develop, and propose new code. Team members can discuss any implementations and proposals through these before changing the source code.

5. Increased code safety

GitHub uses tools to identify and analyze vulnerabilities to the code that other tools tend to miss. Development teams everywhere work together to secure the software supply chain, from start to finish.

6. Easy code hosting

All the code and documentation are located in one place. There are millions of repositories on GitHub, and each repository has its own tools to help you host and release code.

3.3 Language and Library Used

3.3.1 Python Programming Language

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming.

In Python, we don't need to declare the type of variable because it is a dynamically typed language.

Features in Python

There are many features in Python, some of which are discussed below –

1. Easy to code:

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in python language and anybody can learn python basics in a few hours or days. It is also a developer-friendly language.

2. Free and Open Source:

Since it is open-source, this means that source code is also available to the public. So you can download it as, use it as well as share it.

3. Object-Oriented Language:

One of the key features of python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, objects encapsulation, etc.

4. GUI Programming Support:

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in python.

PyQt5 is the most popular option for creating graphical apps with Python.

5. High-Level Language:

Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

6. Extensible feature:

Python is a **Extensible** language. We can write some Python code into C or C++ language and also we can compile that code in the C/C++ language.

7. Python is Portable language:

Python is also a portable language. For example, if we have python code for windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

8. Python is Integrated language:

Python is also an Integrated language because we can easily integrate python with other languages like c, c++, etc.

9. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. unlike other languages C, C++, Java, etc. there is no need to compile python code this makes it easier to debug our code. The source code of python is converted into an immediate form called **bytecode**.

10. Large Standard Library

Python has a large standard library which provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in python for such as regular expressions, unit-testing, web browsers, etc.

11. Dynamically Typed Language:

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don't need to specify the type of variable.

3.3.2 Chatterbot Library

ChatterBot is a Python library that makes it easy to generate automated responses to a user's input. ChatterBot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users. The language independent design of ChatterBot allows it to be trained to speak any language. Additionally, the machine-learning nature

of ChatterBot allows an agent instance to improve it's own knowledge of possible responses as it interacts with humans and other sources of informative data.

How does it work?

ChatterBot is a Python library designed to make it easy to create software that can engage in conversation.

An untrained instance of ChatterBot starts off with no knowledge of how to communicate. Each time a user enters a statement, the library saves the text that they entered and the text that the statement was in response to. As ChatterBot receives more input the number of responses that it can reply and the accuracy of each response in relation to the input statement increase.

The program selects the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

3.3.3 Tkinter

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

Button The Button widget is used to display the buttons in your application.	
2	Canvas The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.
3	Checkbutton The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time.
4	Entry The Entry widget is used to display a single-line text field for accepting values from a user.
5	Frame The Frame widget is used as a container widget to organize other widgets.
6	Label The Label widget is used to provide a single-line caption for other widgets. It can also contain images.
7	Listbox The Listbox widget is used to provide a list of options to a user.
8	Menubutton The Menu Button widget is used to display menus in your application.
9	Menu

	The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.
10	Message The Message widget is used to display multiline text fields for accepting values from a user.
11	Radiobutton The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.
12	Scale The Scale widget is used to provide a slider widget.
13	Scrollbar The Scrollbar widget is used to add scrolling capability to various widgets, such as list boxes.
14	Text The Text widget is used to display text in multiple lines.
15	Toplevel The Toplevel widget is used to provide a separate window container.
16	Spinbox The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values.
17	PanedWindow A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically.
18	LabelFrame

	A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.
19	tkMessageBox This module is used to display message boxes in your applications.

Standard Attributes

Let us look at how some of their common attributes, such as sizes, colors and fonts are specified.

- Dimensions
- Colors
- Fonts
- Anchors
- Relief styles
- Bitmaps
- Cursors

3.3.4 NLP AND NLTK

Natural Language Processing is manipulation or understanding text or speech by any software or machine. An analogy is that humans interact, understand each other's views, and respond with the appropriate answer. In NLP, this interaction, understanding, the response is made by a computer instead of a human.

Usage of natural language processing

- Email filters. Email filters are one of the most basic and initial applications of NLP online.
- Smart assistants.
- Search results.
- Predictive text.
- Language translation.

- Digital phone calls.
- Data analysis.
- Text analytics

NLTK stands for Natural Language Toolkit. This toolkit is one of the most powerful NLP libraries which contains packages to make machines understand human language and reply to it with an appropriate response. Tokenization, Stemming, Lemmatization, Punctuation, Character count, word count are some of these packages which will be discussed in this tutorial.

Chapter – 4

Software Design

4.1 Use Case Diagram

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system. Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities. They also help identify any internal or external factors that may influence the system and should be taken into consideration. They provide a good high level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

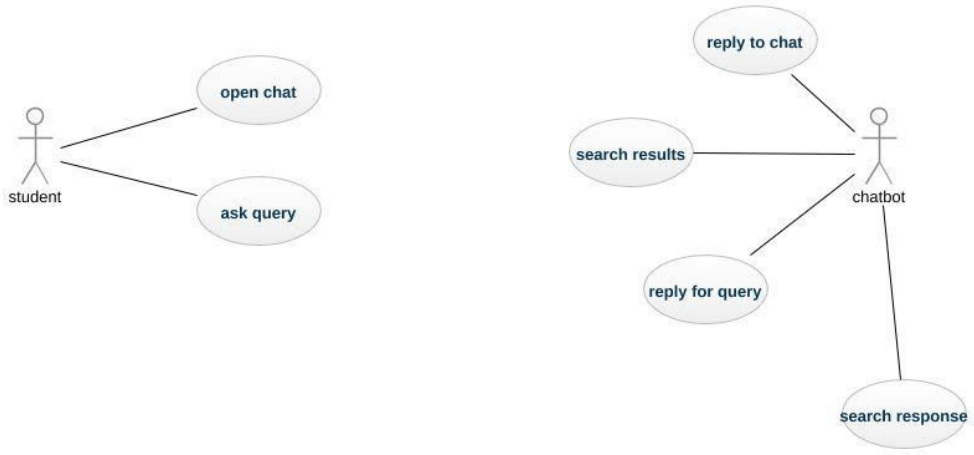


Figure 2: Use Case Diagram of Chatbot

4.2 Flow Diagram

A flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study and plan, improve and communicate often complex processes in clear, easy-to-understand diagrams.

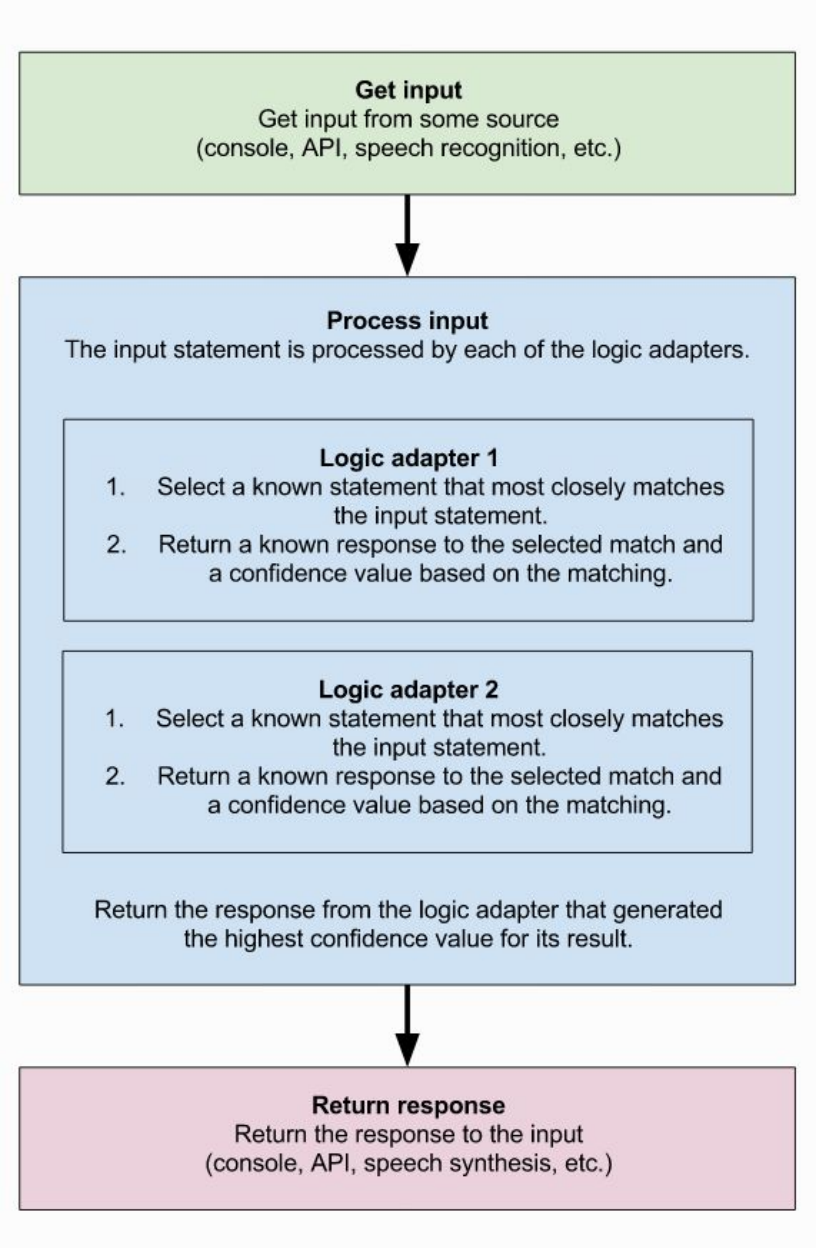


Figure 3: Flow Diagram of Chatbot

4.3 DFD (Data flow diagram)

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth,

multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyses an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That's why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

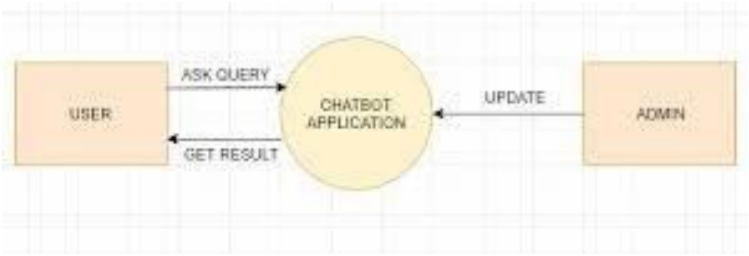


Figure 4: Data Flow Diagram for Chatbot

Chapter – 5

Implementation and User Interface

5.1 System Implementation

Systems implementation is the process of: defining how the information system should be built (i.e., physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e., quality assurance).

This section covers the design and implementation of different module of the bot, which contains the design of the Python module, the Translator API and the Machine Learning module.

5.1.1 Steps to create an intelligent Chatbot:-

Step 1. Import libraries and load the data

Step 2. Preprocessing the data.

Step 3. Create training and testing data.

Step 4. Training the model.

Step 5. Interacting with the chatbot.

Step 6. Running the chatbot.

5.1.2 Installation

We start the project of Technobot by installing the chatterbot library. For creating chatbot also need to install chatterbot corpus. Corpus - literal meaning is a collection of words. This contains a corpus of data that is included in the chatterbot module. Each corpus is nothing but a prototype of different input statements and their responses. These corpus are used by bots to train themselves. The most recommended method for installing chatterbot and chatterbot_corpus is by using pip.

- **pip install chatterbot**
- **pip install chatterbot_corpus**

then install nltk library and import Chatterbot library.

- **pip install nltk**
- **from chatterbot import Chatbot**

Before going deep into the above script let me remind you that we need to install a few libraries.

- Install pip and update pip
- Install chatterbot
- Install nltk
- Install scipy
- Install spacy
- Install Pint
- Install mathparse

5.1.3 Training the chatbot

Now the final step in making a chatbot is to train the chatbot using the modules available in chatterbot. Training a chatbot using chatterbot is as simple as providing a conversation into the chatbot database. As soon as the chatbot is given a dataset, it produces the essential entries in the chatbot's knowledge graph to represent the input and output in the right manner. Firstly, let's import the ListTrainer, create its object by passing the Chatbot object, and call the train() method by passing a list of sentences.

We will create a while loop for our chatbot to run in. When statements are passed in the loop, we will get an appropriate response for it, as we have already entered data into our database.

Chatbot Testing

The last step of this project is to test the chatterbot's conversational skills. For testing its responses, we will call the `get_responses()` method of `Chatbot` instance.

5.2 User Interface

The user interface (UI), in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.

Generally, the goal of user interface design is to produce a user interface which makes it easy, efficient, and enjoyable (user-friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human.

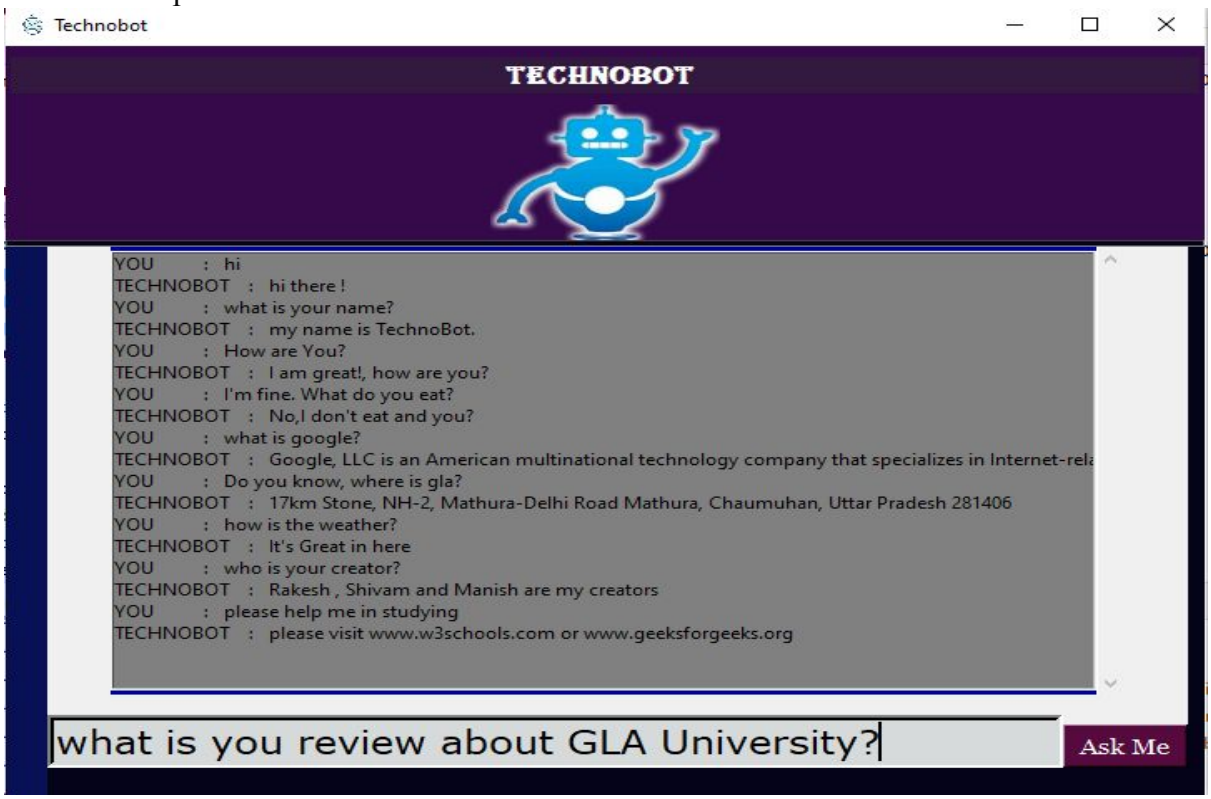


Figure 5: User Interface of Chatbot

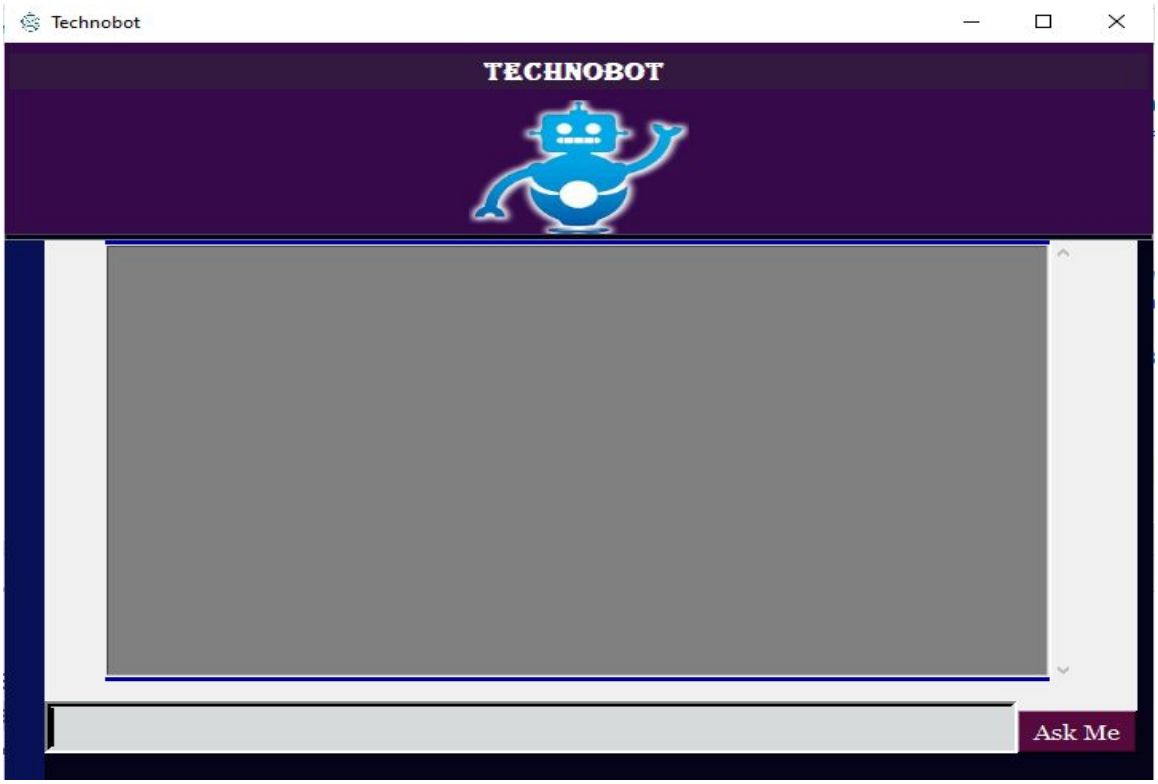


Figure 6: Initial Image of Chatbot interface

CHAPTER – 6

CONCLUSION

From my perspective, chatbots or smart assistants with artificial intelligence are dramatically changing businesses. There is a wide range of chatbot building platforms that are available for various enterprises, such as e-commerce, retail, banking, leisure, travel, healthcare, and so on.

Chatbots can reach out to a large audience on messaging apps and be more effective than humans. They may develop into a capable information-gathering tool in the near future.

These days, consumers expect to be able to find the information they're looking for online quickly and easily. And when a business can't provide that type of experience, they become frustrated. Chatbots are poised to ease these frustrations by providing the real-time, on-demand approach that consumers are seeking out.

The top three potential benefits of chatbots that consumers reported in our survey:

1. 24-hour service (64%)
2. Instant responses (55%)
3. Answers to simple questions (55%)

And that's true across all age groups. It's not just Millennials who see the potential benefits of chatbots. In fact, Baby Boomers were 24% more likely to to expect benefits from chatbots in five of the nine categories we looked at compared to their Millennial counterparts.

However, chatbots — like all technologies — aren't without their limitations: 43% of consumers said they prefer dealing with an actual person (that was the number one potential barrier to using chatbots). That being said, 34% of consumers also predicted that they would use chatbots for getting connected with a human. So it doesn't have to be either/or. As a business, you can use chatbots to supplement your human workforce (not replace them).

Compared to other business communication channels, chatbots scored the second-highest when it came to consumers expecting instant responses, only losing out to online chat. But by using chatbots in combination with online chat, businesses can deliver a level of real-time service that they'd be unable to achieve using either technology on its own.

CHAPTER – 7

REFERENCES

[1]- International Journal of Innovative Research in Computer Science & Technology (IJIRCST) ISSN: 2347-5552, Volume-6, Issue-3,

[2]- AIML Based Voice Enabled Artificial Intelligent Chatterbot", International Journal of u- and e- Service, Science and Technology Volume 8 - No. 2, 2015.

[3]- www.Researchgate.net

CHAPTER – 8

APPENDICES

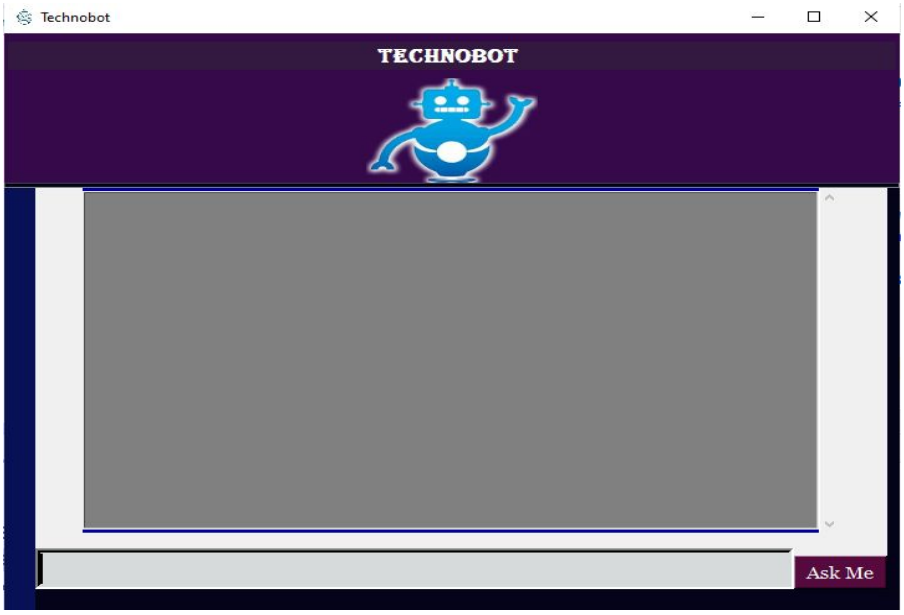


Figure 7: User Interface of Chatbot

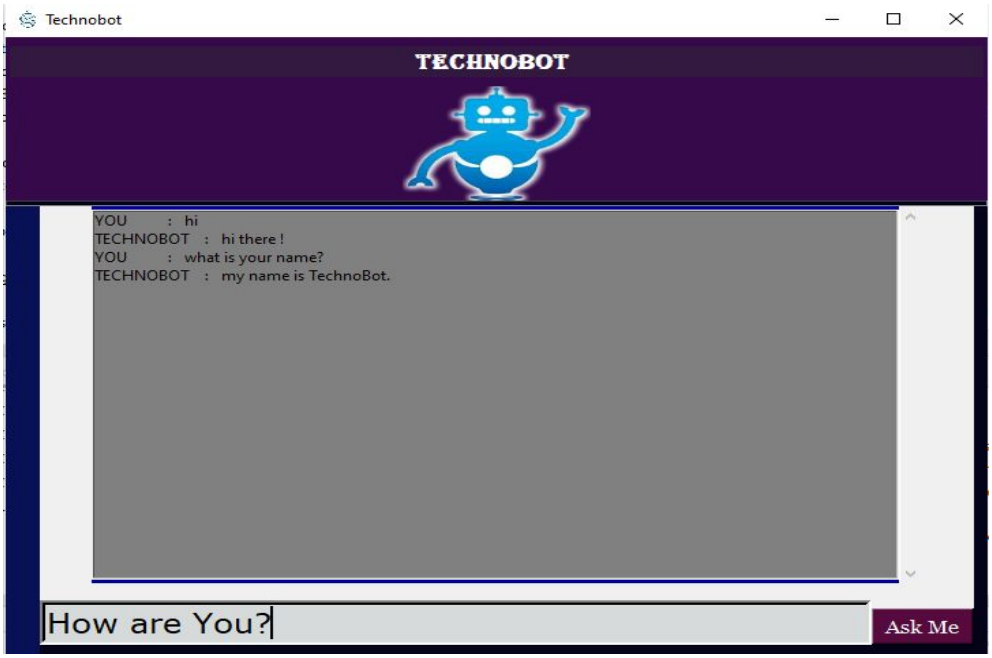


Figure 8

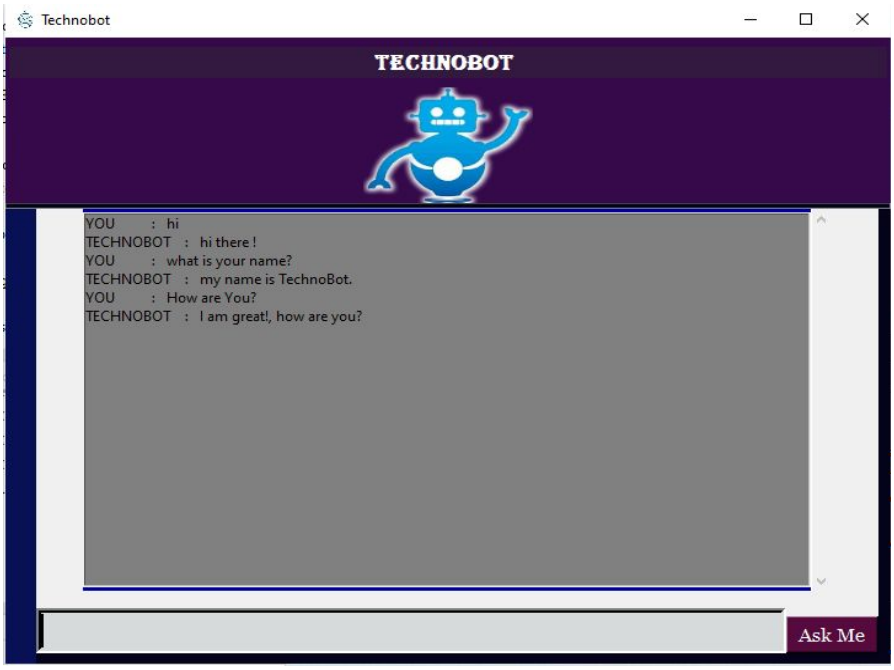


Figure 9

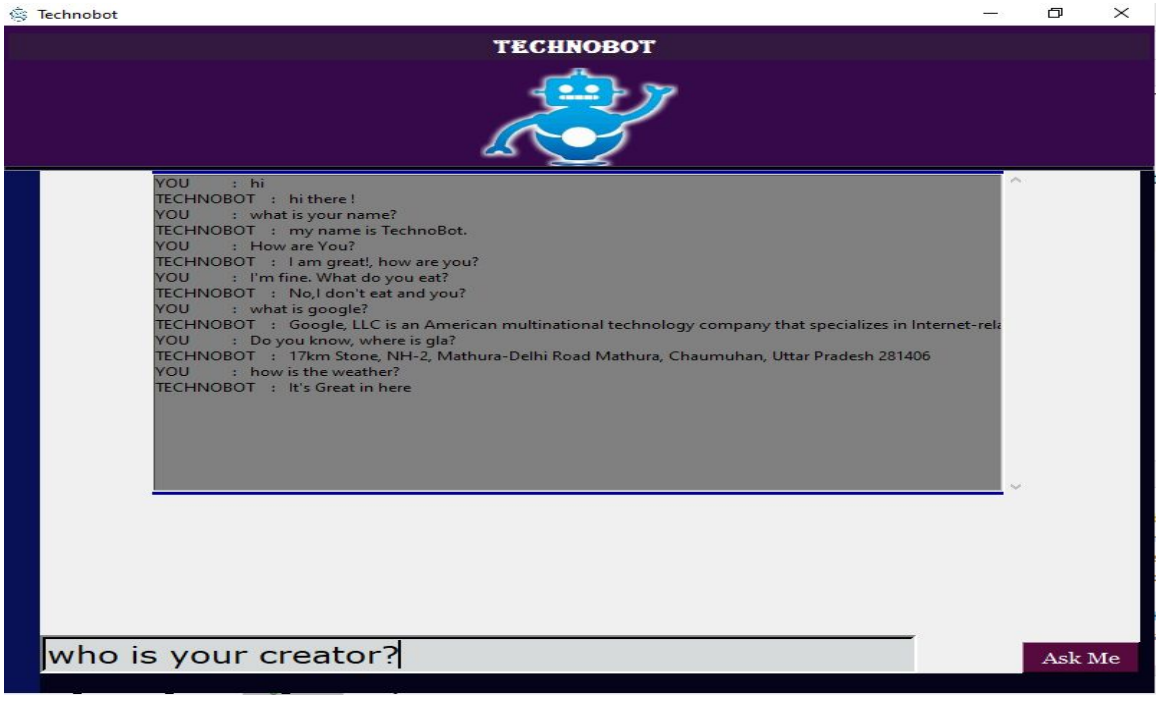


Figure 10

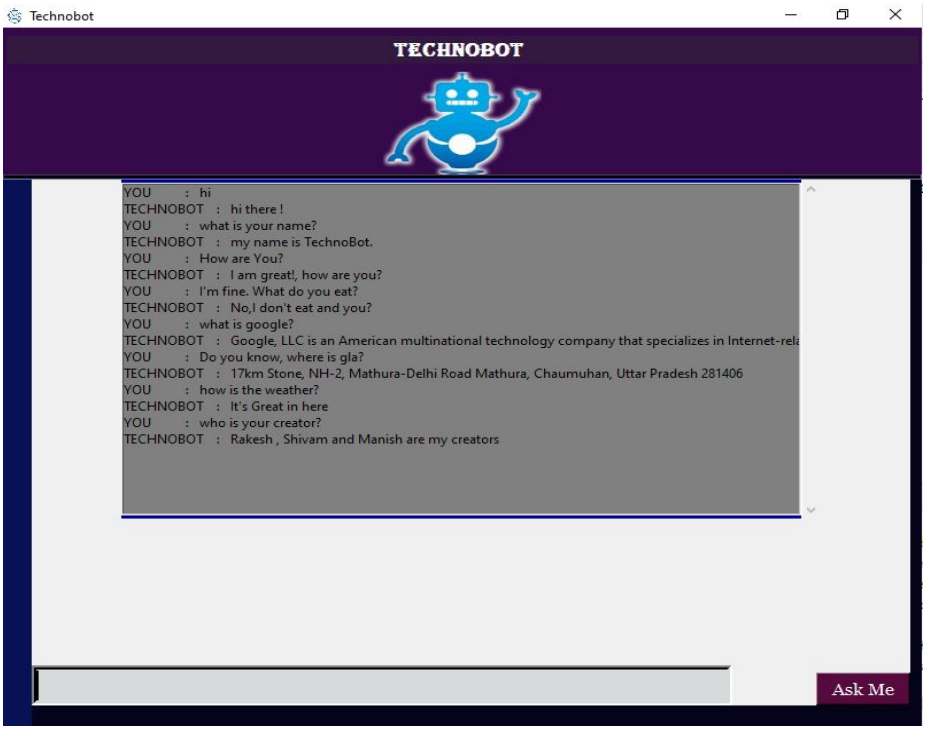


Figure 11

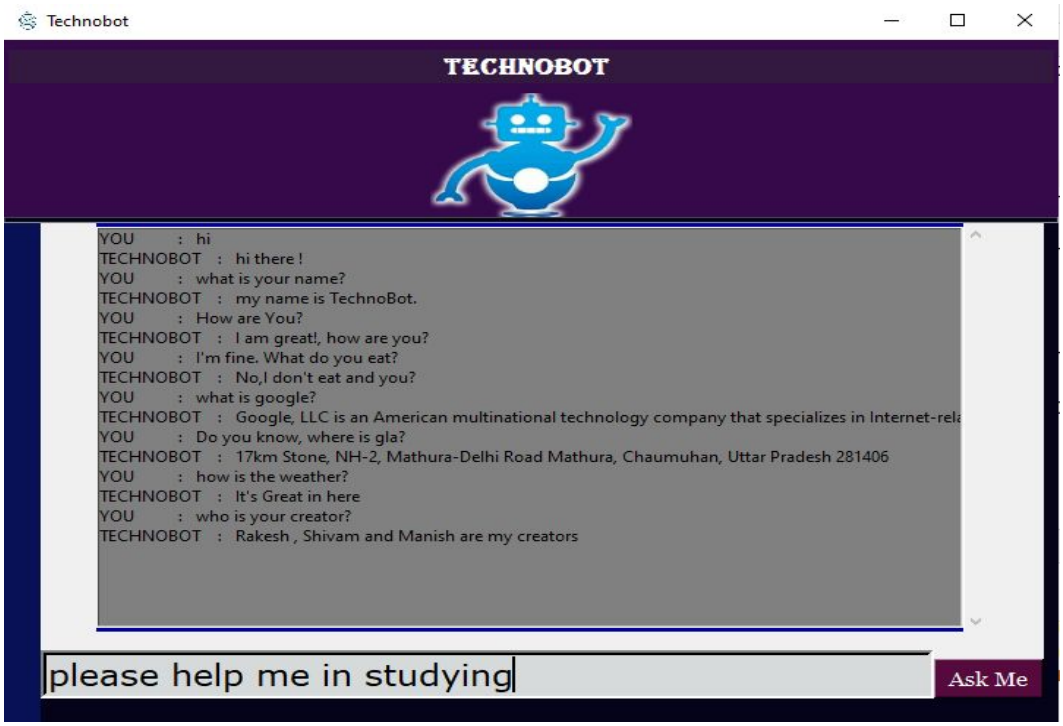


Figure 12

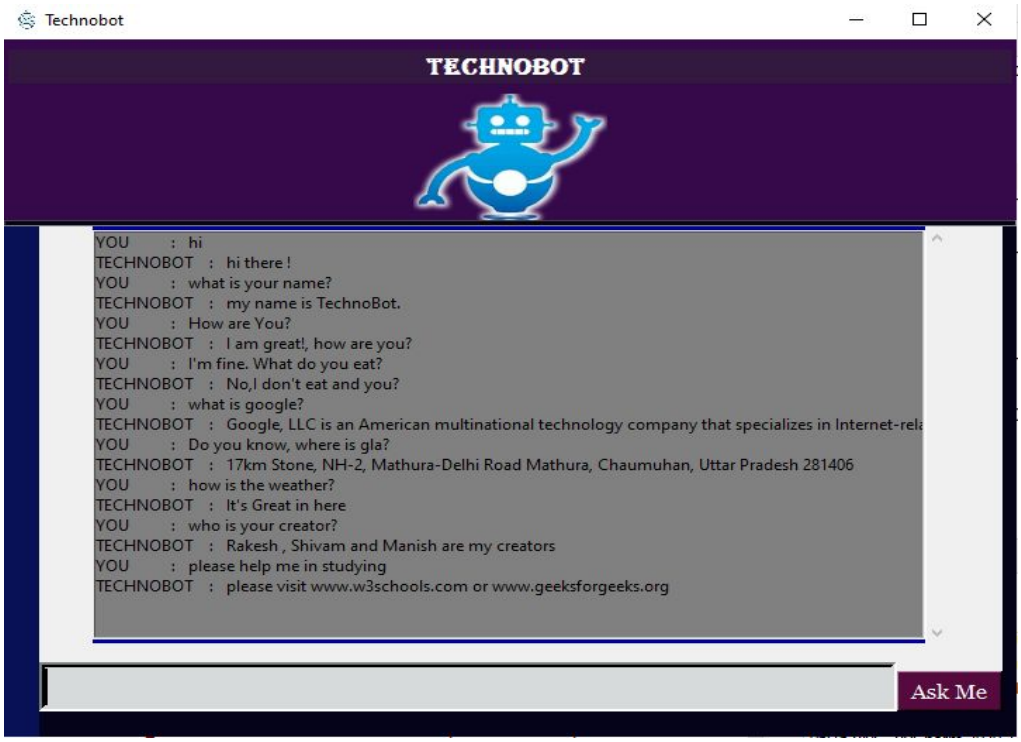


Figure 13

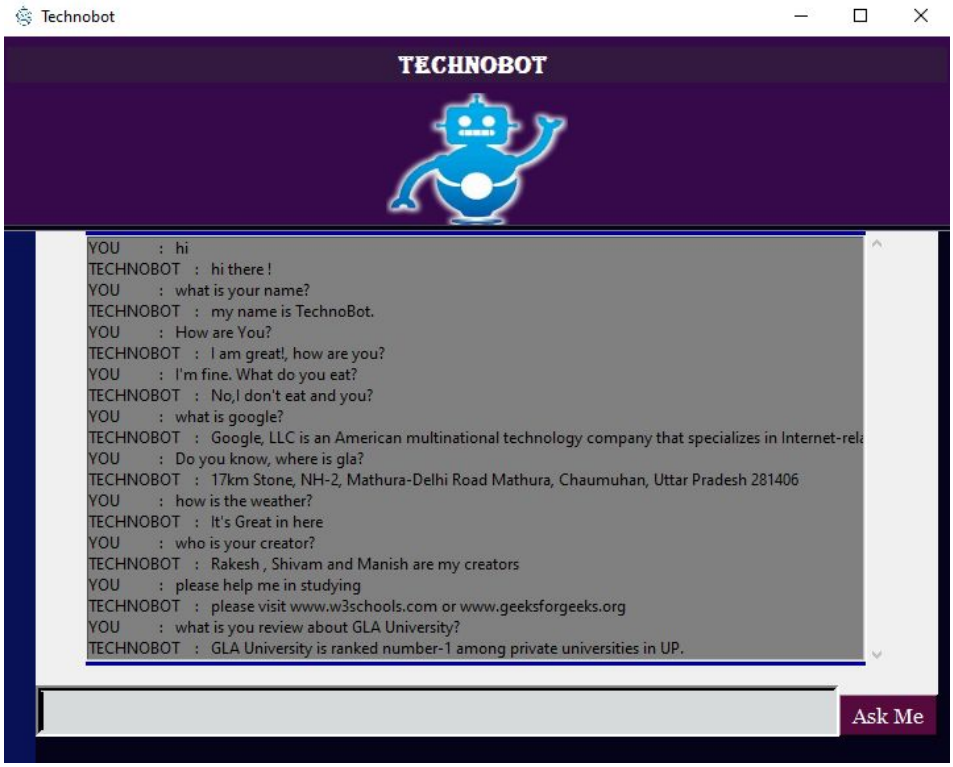


Figure 14

Figure 15: thank you reply from technobot

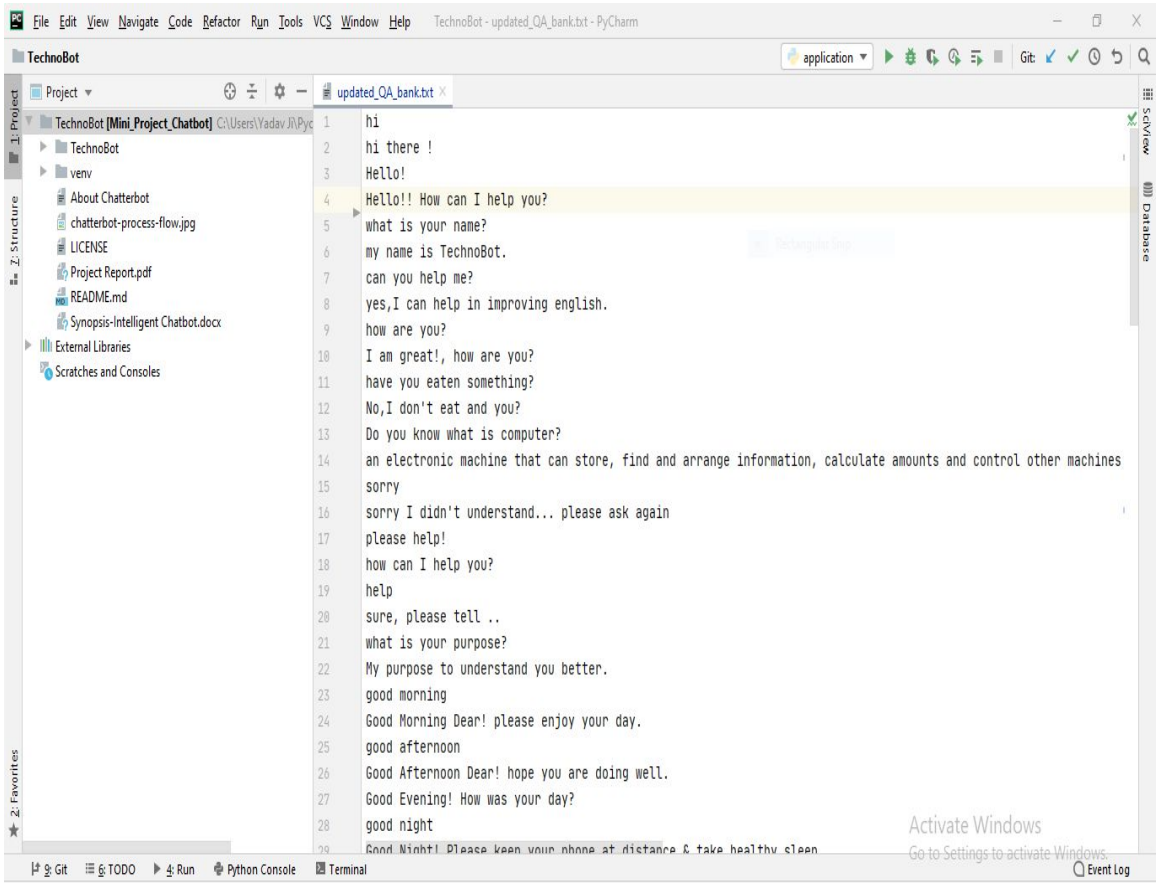


Figure 16: Q/A .text file

Rule-01 : **Must write questions before you are going to write each answer to be displayed.**

Rule-02 : **Try to write your questions list of text file in lowercase.**

Rule-03 : **Must run application.py for training your technobot with the text file.**

As per requirements, we can update this text file for the best outputs and we are sure that this feature will definitely help all technobot users.

technobot.py

```
"""..."""

from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer

# Create a new chat-bot named as TechnoBot
chatbot = ChatBot('TechnoBot')
trainer = ListTrainer(chatbot)

# using file-handling concept to accessing training-data
training_data = open('training_data/updated_QA_bank.txt').read().splitlines()

trainer.train(training_data)

# while True:
#     query = input()
#     if query == "exit":
#         break
#     bot_response = chatbot.get_response(query)
#     print("Bot : ", bot_response)
```

Figure 17: technobot.py file

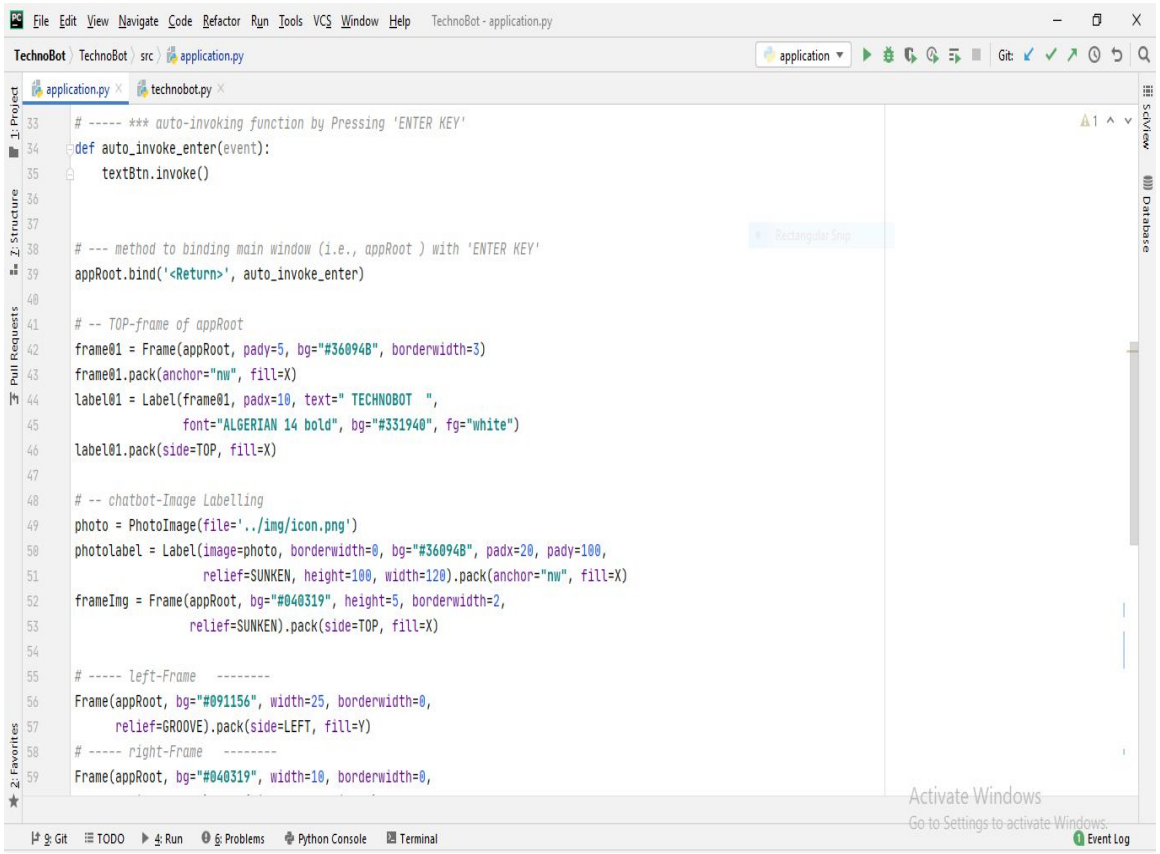
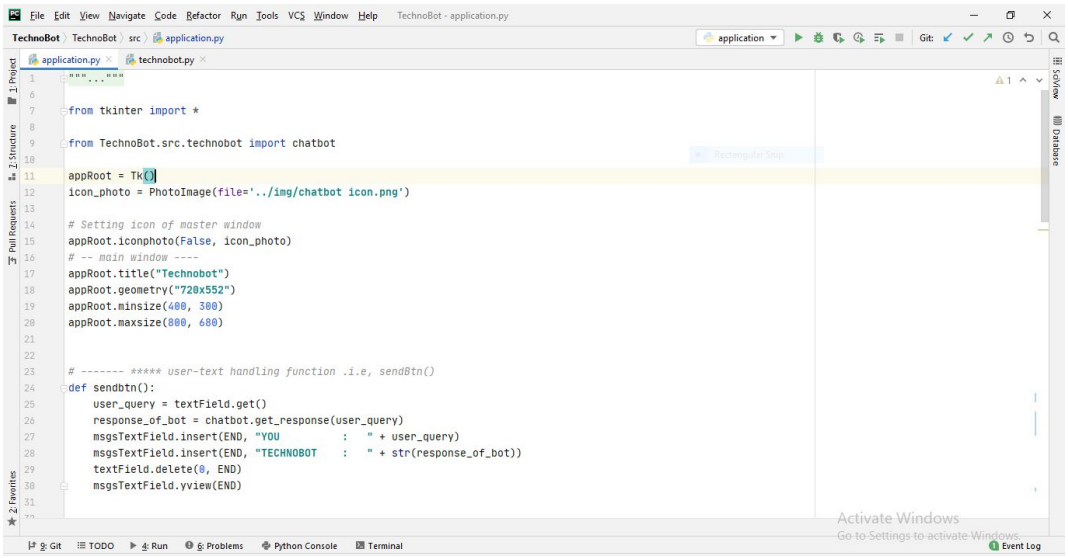
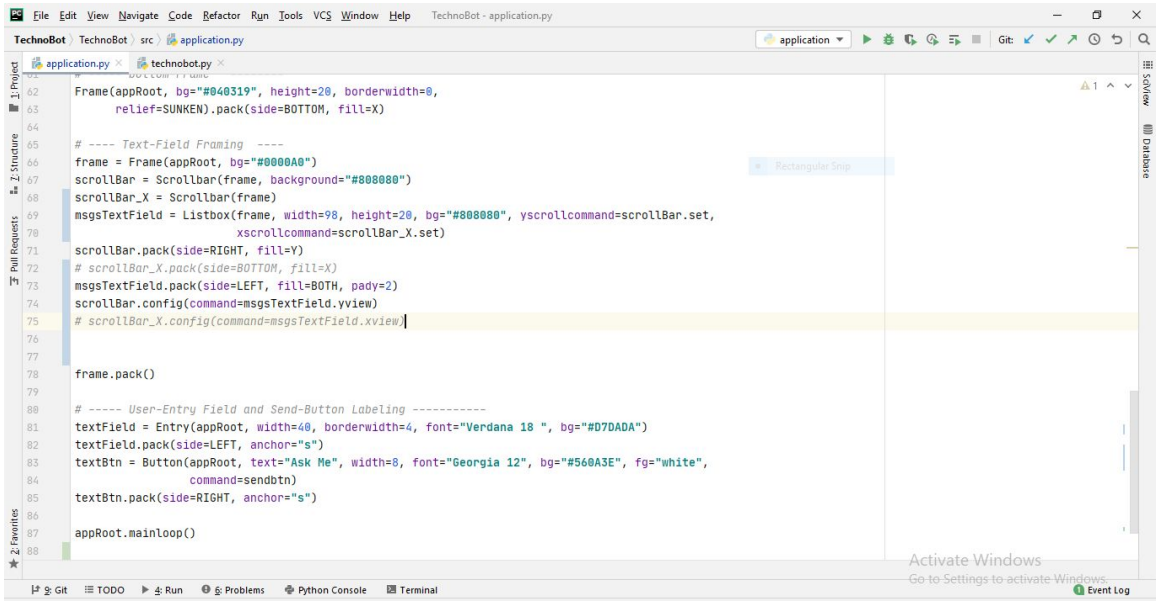


Figure 18



```
"""..."""

from tkinter import *

from TechnoBot.src.technobot import chatbot

appRoot = Tk()
icon_photo = PhotoImage(file='../img/chatbot icon.png')

# Setting icon of master window
appRoot.iconphoto(False, icon_photo)
# -- main window ----
appRoot.title("Technobot")
appRoot.geometry("720x552")
appRoot.minsize(400, 300)
appRoot.maxsize(800, 680)


# ------- ***** user-text handling function .i.e, sendBtn()
def sendbtn():
    user_query = textField.get()
    response_of_bot = chatbot.get_response(user_query)
    msgsTextField.insert(END, "YOU          :   " + user_query)
    msgsTextField.insert(END, "TECHNOBOT    :   " + str(response_of_bot))
    textField.delete(0, END)
    msgsTextField.yview(END)
```

Figure 19



```
Frame(appRoot, bg="#040319", height=20, borderwidth=0,
      relief=SUNKEN).pack(side=BOTTOM, fill=X)

# ---- Text-Field Framing  ----
frame = Frame(appRoot, bg="#0000A0")
scrollBar = Scrollbar(frame, background="#808080")
scrollBar_X = Scrollbar(frame)
msgsTextField = Listbox(frame, width=98, height=20, bg="#808080", yscrollcommand=scrollBar.set,
                        xscrollcommand=scrollBar_X.set)
scrollBar.pack(side=RIGHT, fill=Y)
# scrollBar_X.pack(side=BOTTOM, fill=X)
msgsTextField.pack(side=LEFT, fill=BOTH, pady=2)
scrollBar.config(command=msgsTextField.yview)
# scrollBar_X.config(command=msgsTextField.xview)


frame.pack()

# ----- User-Entry Field and Send-Button Labeling ----------
textField = Entry(appRoot, width=40, borderwidth=4, font="Verdana 18 ", bg="#D7DADA")
textField.pack(side=LEFT, anchor="s")
textBtn = Button(appRoot, text="Ask Me", width=8, font="Georgia 12", bg="#560A3E", fg="white",
                 command=sendbtn)
textBtn.pack(side=RIGHT, anchor="s")

appRoot.mainloop()
```

Figure 20

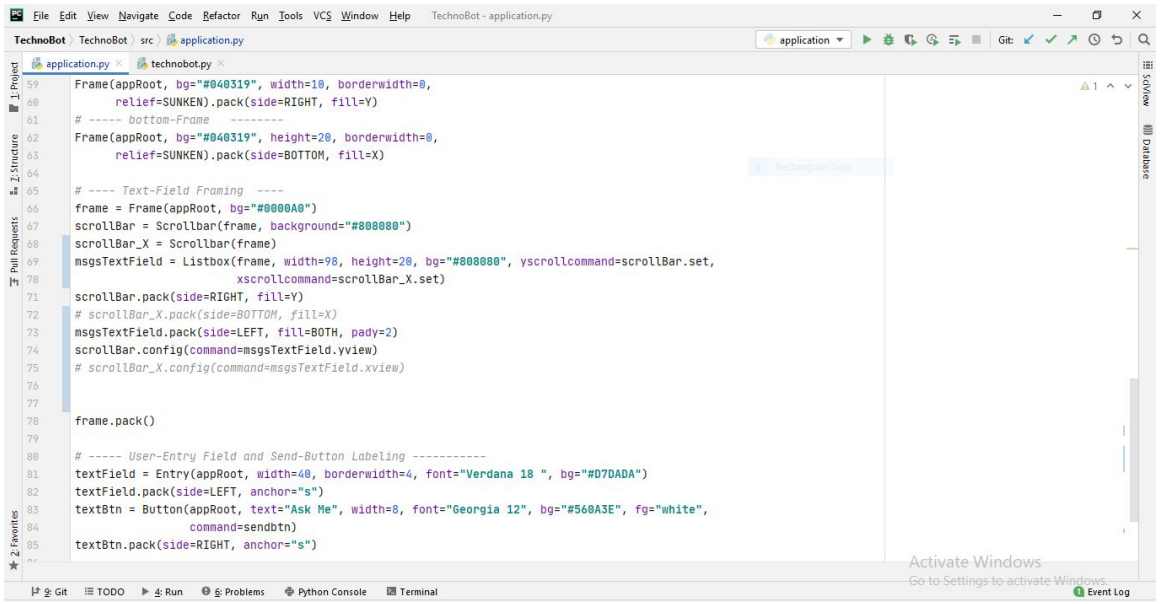


Figure 21

Application.py file Source Code

```
"""
Author: Rakesh Yadav,Manish Sharma
Aim: To make a conversational chatbot by using ML, NLTK & tkinter
       through python.
"""

from tkinter import *

from TechnoBot.src.technobot import chatbot

appRoot = Tk()
icon_photo = PhotoImage(file='../img/chatbot icon.png')


# Setting icon of master window
appRoot.iconphoto(False, icon_photo)
# -- main window ----
appRoot.title("Technobot")
appRoot.geometry("720x552")
appRoot.minsize(400, 300)
appRoot.maxsize(800, 680)


# ------- ***** user-text handling function .i.e, sendBtn()
def sendbtn():
   user_query = textField.get()
   response_of_bot = chatbot.get_response(user_query)
   msgsTextField.insert(END, "YOU           :    " + user_query)
```

```
   msgsTextField.insert(END,    "TECHNOBOT          :          "    +
str(response_of_bot))
   textField.delete(0, END)
   msgsTextField.yview(END)


# ----- *** auto-invoking function by Pressing 'ENTER KEY'
def auto_invoke_enter(event):
   textBtn.invoke()


# --- method to binding main window (i.e., appRoot ) with 'ENTER
KEY'
appRoot.bind('<Return>', auto_invoke_enter)

# -- TOP-frame of appRoot
frame01 = Frame(appRoot, pady=5, bg="#36094B", borderwidth=3)
frame01.pack(anchor="nw", fill=X)
label01 = Label(frame01, padx=10, text=" TECHNOBOT  ",
                font="ALGERIAN 14 bold", bg="#331940", fg="white")
label01.pack(side=TOP, fill=X)

# -- chatbot-Image Labelling
photo = PhotoImage(file='../img/icon.png')
photolabel   =   Label(image=photo,   borderwidth=0,   bg="#36094B",
padx=20, pady=100,
                  relief=SUNKEN,                          height=100,
width=120).pack(anchor="nw", fill=X)
frameImg = Frame(appRoot, bg="#040319", height=5, borderwidth=2,
                relief=SUNKEN).pack(side=TOP, fill=X)

# ----- left-Frame   --------
Frame(appRoot, bg="#091156", width=25, borderwidth=0,
     relief=GROOVE).pack(side=LEFT, fill=Y)
# ----- right-Frame   --------
Frame(appRoot, bg="#040319", width=10, borderwidth=0,
     relief=SUNKEN).pack(side=RIGHT, fill=Y)
# ----- bottom-Frame   --------
Frame(appRoot, bg="#040319", height=20, borderwidth=0,
     relief=SUNKEN).pack(side=BOTTOM, fill=X)

# ---- Text-Field Framing  ----
frame = Frame(appRoot, bg="#0000A0")
scrollBar = Scrollbar(frame, background="#808080")
scrollBar_X = Scrollbar(frame)
msgsTextField  =  Listbox(frame,  width=98,  height=20,  bg="#808080",
yscrollcommand=scrollBar.set,
                        xscrollcommand=scrollBar_X.set)
scrollBar.pack(side=RIGHT, fill=Y)
# scrollBar_X.pack(side=BOTTOM, fill=X)
msgsTextField.pack(side=LEFT, fill=BOTH, pady=2)
scrollBar.config(command=msgsTextField.yview)
# scrollBar_X.config(command=msgsTextField.xview)
```

```
frame.pack()

# ----- User-Entry Field and Send-Button Labeling ----------
textField = Entry(appRoot, width=40, borderwidth=4, font="Verdana
18 ", bg="#D7DADA")
textField.pack(side=LEFT, anchor="s")
textBtn = Button(appRoot, text="Ask Me", width=8, font="Georgia
12", bg="#560A3E", fg="white",
                 command=sendbtn)
textBtn.pack(side=RIGHT, anchor="s")

appRoot.mainloop()
```

..... Technobot.py - source code .....

```
"""
Author: Shivam Yadav, Manish Sharma
Aim: To make a conversational chatbot by using ML, NLTK & tkinter
       through python.
"""


from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer

# Create a new chat-bot named as TechnoBot
chatbot = ChatBot('TechnoBot')
trainer = ListTrainer(chatbot)
```

```
# using file-handling concept to accessing training-data
training_data                                                    =
open('training_data/updated_QA_bank.txt').read().splitlines()

trainer.train(training_data)

# while True:
#     query = input()
#     if query == "exit":
#         break
#     bot_response = chatbot.get_response(query)
#     print("Bot : ", bot_response)
```