**Nihi Voice Assistant**

**Project Report**

Mini project (ICI552)

**BACHELOR OF COMPUTER APPLICATION (CTIS)**

**( i Nurture )**

|  |  |
| --- | --- |
| PROJECT GUIDE:  **Shashank Saxena** | SUBMITTED BY:  **Sachin Kumar (TCA2056020)**  **Vivek Pal (TCA2056027)**  **Tushar Vishnoi (TCA2056025)** |

September, 2022



**FACULTY OF ENGINEERING & COMPUTING SCIENCES**

**TEERTHANKER MAHAVEER UNIVERSITY, MORADABAD**

**DECLARATION**

We hereby declare that this Project Report titled “ Voice based **Chatbot in Python**” submitted by us and approved by our project guide, Faculty of Engineering & Computing Sciences. Teerthanker Mahaveer University, Moradabad, is a bonafide work undertaken by us and it is not submitted to any other University or Institution for the award of any degree /diploma / certificate or published any time before.

|  |  |  |
| --- | --- | --- |
| **Project ID :** | Nihi chatbot | |
| **Student Name:** | Vivek pal | Signature |
| **Student Name:** | Sachin saini | Signature |
| **Student Name:** | Tushar Vishnoi | Signature |
|  |  |  |
| **Project Guide :** | Shashank Saxena | Signature |

Table of Contents

[1 Project Title 4](#_Toc31392321)

[2 Problem Statement 4](#_Toc31392322)

[3 Project Description 4](#_Toc31392323)

[3.1 Scope of the Work 7](#_Toc31392324)

[3.2 Project Modules 7](#_Toc31392325)

[3.3 Context Diagram (High Level) 7](#_Toc31392326)

[4 Implementation Methodology 7](#_Toc31392327)

[5 Technologies to be used 10](#_Toc31392328)

[5.1 Software Platform 12](#_Toc31392329)

[5.2 Hardware Platform 16](#_Toc31392330)

[5.3 Tools, if any 16](#_Toc31392331)

[6 Advantages of this Project 16](#_Toc31392332)

[7 Assumptions, if any 19](#_Toc31392333)

[8 Future Scope and further enhancement of the Project 19](#_Toc31392334)

[9 Project Repository Location 19](#_Toc31392335)

[10 Definitions, Acronyms, and Abbreviations 20](#_Toc31392336)

[11 Conclusion 21](#_Toc31392337)

[12 References 23](#_Toc31392338)

**Appendix**

**A: Data Flow Diagram (DFD)**

**B: Entity Relationship Diagram (ERD)**

**C: Use Case Diagram (UCD)**

**D: Data Dictionary (DD)**

**E: Screen Shots**

# Project Title

Nihi Voice assistance

# Problem Statement

**A chatbot is an artificially intelligent creature which can converse with humans. This could be text-based, or a spoken conversation (in case of voice-based queries). Chat bots are basically used for information acquisition. It can run on the local PCs and mobile phones, though most of the time it is accessed through the internet. It can be compelling, captivating and spell-bounding. It is a conversational agent which interacts with users in a certain domain or on a particular topic with input in natural language sentences. Mainly a chatbot works by a user asking some question or initiating a new topic of discussion. Chat bots can be referred as software agents that pretend as human entity. These are the agents with AI embedded and using NLP they can answer to user questions. Predefined knowledge base helps develop a response to the query.**

**Chatbot is an intelligent piece of software that is capable of communicating and performing actions similar toa human.In other terms, a bot is a computer program that is designed to communicate with human users through the internet**

**For businesses, it has become necessary to solve the queries and problems of the customers to ensure consumer loyalty along with the brand establishment. And just like the earlier times, man has looked to take help of machines to remove the constraints of human limitations. This time it is the customer service industry which has been revolutionized, and the innovation responsible for this is chatbot. Chatbots are considered the future of customer service and management.  
A chatbot interacts on a format similar to instant messaging. By artificially replicating the patterns of human interactions in**[**machine learning**](https://mindmajix.com/machine-learning-vs-ai)**allows computers to learn by themselves without programming natural language processing.**

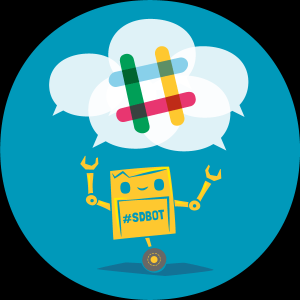
**While a bot is a computer’s ability to understand human speech or text short for chat robot. A chatbot is merely a computer program that fundamentally simulates human conversations. It allows a form of interaction between a human and a machine the communication, which happenvia messages or voice command.**

**A chatbot is programmed to work independently from a human operator. It can answer questions formulated to it in natural language and respond like a real person. It provides responses based on a combination of predefined scripts and machine learning applications.**

**When it is asked a question, the chatbot will respond based on the knowledge database available to it at that point in time. If the conversation introduces a concept it is not programmed to understand, it will either deflect the conversation or potentially pass the communication to a human operator. Either way, it will also learn from that interaction as wellas from future interactions. Thus, the chatbot will gradually grow in scope and gain relevance.**

**We are selected this topic because it provides many benefits like:**

**Work Automation:**



People tend to be less productive when given a recurring job or work. We humans usually get bored doing the same thing over and over again. Chatbots can now automate tasks which are to be done frequently and at the right time.

### ****Cost Effective:****

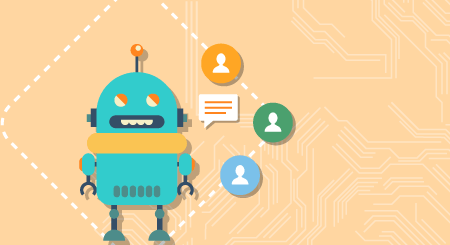


Hiring a human for a job is never a cheap affair, and it will be expensive if your revenue are not high or sales targets are not met and would create havoc in the business. Due to the boundaries of human beings, a single human can only handle one or two people at the same time. More than that would be extremely tough for the employee.

[Chatbots](https://wotnot.io/templates) could help solve this age-old problem. As one chatbot is equal to loads of employees, it can easily communicate with thousands of customers at the same time. We would only need a handful of people to jump into conversations sometimes when necessary. Hence, it would drastically bring down the expenses and bring about a steep rise in revenue and customer satisfaction.

### Handling Capacity

Unlike humans who can only communicate with one human at a time, chat bots can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered immediately



# 

# Project Description

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 themost 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:

iLimited set of rules

ii. 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.

## Scope of the Work

1. Intelligent enough to understand the patterns and put across answers that are appropriate and relevant, Chabot have come a long way. With efficient Chabot 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.
2. Simple chatbots were capable of matching a text string and offering an answer only when the exact match is found.
3. 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.
4. Users love to interact with chatbot as it saves them time andin most cases offers them clear and concrete answers. They may not be perfect but they are scary close to be perfect.

## Project Modules

## A chatbot based on the retrieval-based model works on the concept of predefined responses. The chatbot picks appropriate responses from the repository stacked which is based on the context and query raised by the user. Generative models built using machine translation techniques come with the ability to generate new responses right from the word go. Generative models enable longer conversations where the chatbot deals with several users queries.

## Context Diagram (High Level)

 Context diagram, sometimes called a level 0 data-flow diagram, is drawn in order to define and clarify the boundaries of the software system. It identifies the flows of information between the system and external entities.

Bot

matchig

conversatinn

Sending

matching

heard

Load active conversation

received

Get message

Call matching method

Send bot response

# Implementation Methodology

**Our project methodology is as follows:-**

The project requires you to have good knowledge of Python, Keras, and [**Natural language processing (NLTK)**](https://data-flair.training/blogs/nltk-python-tutorial/). Along with them, we will use some helping modules which you can download using the python-pip command.

When working with text data, we need to perform various preprocessing on the data before we make a machine learning or a deep learning model. Tokenizing is the most basic and first thing you can do on text data. Tokenizing is the process of breaking the whole text into small parts like words.

**1-Import and load the data file**

First, make a file name as train\_chatbot.py. We import the necessary packages for our chatbot and initialize the variables we will use in our Python project.

    hour = int(datetime.datetime.now().hour)

    if hour>=0 and hour<12:

        speak("Good Morning!")

    elif hour>=12 and hour<16:

        speak("Good Afternoon!")

    elif hour>=16 and hour<24:

        speak("Good Evening!")

    else:

        speak("Good Night!")

    speak("I am Nihi Sir. Please tell me how may I help you")

Input—we process database ---- through file handling.

 elif 'open youtube' in query:

            webbrowser.open("youtube.com")

        elif 'open facebook' in query:

            webbrowser.open("facebook.com")

        elif 'open instagram' in query:

            webbrowser.open("instagram.com")

        elif 'open crackin' in query:

            webbrowser.open("crackin.com")

        elif 'open google' in query:

            webbrowser.open("google.com")

Methodology—python library

import pyttsx3 #pip install pyttsx3

import speech\_recognition as sr #pip install speechRecognition

import datetime

import wikipedia #pip install wikipedia

import webbrowser

import os

Description of some libraries in python

1. pyttsx3-It is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python . pip install pyttsx3. If you recieve errors such as No module named win32com.client, No module named win32, or No module named win32api, you will need to additionally install pypiwin32.
2. Library for performing speech recognition, with support for several engines and APIs, online and offline. Library for performing speech recognition, with support for several engines and APIs, online and offline. Speech recognition engine/API support: Quickstart: pip install SpeechRecognition.
3. Wikipedia-Wikipedia is a Python library that makes it easy to access and parse data from Wikipedia. Search Wikipedia, get article summaries, get data like links and images from a page, and more. Wikipedia wraps the MediaWiki API so you can focus on using Wikipedia data, not getting it.
4. According to Python's standard documentation, the webbrowser module provides a high-level interface to allow displaying Web-based documents to users. This topic explains and demonstrates proper usage of the webbrowser module

Recognition voice based

Output—after recognize the voice of the user, our chatbot instructs to browser to open the application

**Diagram to show the working of the chatbot**

Bot

heard

matching

Load active conversation

received

Get message

Sending

Call matching method

Send bot response

conversation

matching

# Technologies to be used

## Software Platform

1. **Front-end**

The most basic use case for Python is as a scripting and automation language. Python isn’t just a replacement for shell scripts or batch files, but is also used to automate interactions with web browsers or application GUIs or system provisioning and configuration in tools such as Ansible and Salt. But scripting and automation represent only the tip of the iceberg with Python.

 Python is used for general application programming. Both CLI and cross-platform GUI applications can be created with Python and deployed as self-contained executables. Python doesn’t have the native ability to generate a standalone binary from a script, but third-party packages like cx\_Freeze or PyInstaller can be used to accomplish that.

 Python is used for data science and machine learning. Sophisticated data analysis has become one of fastest moving areas of IT and one of Python’s star use cases. The vast majority of the libraries used for data science or machine learning have Python interfaces, making the language the most popular high-level command interface to for machine learning libraries and other numerical algorithms.

 Python is used for web services and RESTful APIs. Python’s native libraries and third-party web frameworks provide fast and convenient ways to create everything from simple REST APIs in a few lines of code, to full-blown, data-driven sites. Python’s latest versions have powerful support for asynchronous operations, allowing sites to handle up to tens of thousands of requests per second with the right libraries.

 Python is used for metaprogramming. In Python, everything in the language is an object, including Python modules and libraries themselves. This allowsPython to work as a highly efficient code generator, making it possible to write applications that manipulate their own functions and have the kind of extensibility that would be difficult or impossible to pull off in other languages.

 Python is used for glue code. Python is often described as a “glue language,” meaning it can allow disparate code (typically libraries with C language interfaces) to interoperate. Its use in data science and machine learning is in this vein, but that’s just one incarnation of the general idea.

Also worth noting are the sorts of tasks Python is not well-suited for. Python is a high-level language, so it’s not suitable for system-level programming—device drivers or OS kernels are straight out. It’s also not ideal for situations that call for cross-platform standalone binaries. You could build a standalone Python app for Windows, Mac, and Linux, but not elegantly or simply. Finally, Python is not the best choice when speed is an absolute priority in every aspect of the application. For that you’re better off with C/C++ or another language of that caliber.

**The Python language’s pros and cons**

Python syntax is meant to be readable and clean, with little pretense. A standard “hello world” in Python 3.x is nothing more than:

* print(“Hello world!”)
* Python provides many syntactical elements that make it possible to concisely express many common program flows. Consider a sample program for reading lines from a text file into a list object, stripping each line of its terminating newline character along the way:
* with open(‘myfile.txt’) as my\_file:
* file\_lines = [x.strip(‘\n’) for x in my\_file]
* The with/as construction is a “context manager,” which provides an efficient way to instantiate a given object for a block of code and then dispose of it outside of that block. In this case, the object in question is my\_file, instantiated with the open() function. This takes the place of several lines of boilerplate to open the file, read individual lines from it, then close it up
* The [x … for x in my\_file] construction is another Python idiosyncrasy, the “list comprehension.” It allows a given item that contains other items (here, my\_file and the lines it contains) to be iterated through, and to allow each iterated element (that is, each x) to be processed and automatically appended into a list.
* You could write such a thing as a formal for… loop in Python, much as you would in another language. The point is that Python has a way to economically express things like loops that iterate over multiple objects and perform some simple operation on each element in the loop, or work with things that require explicit instantiation and disposal. Constructions like this allow Python developers to balance terseness and readability.
* Python’s other language features are meant to complement common use cases. Most modern object types—Unicode strings, for instance—are built directly into the language. Data structures—like lists, dictionaries (i.e., hashmaps), tuples (for storing immutable collections of objects), and sets (for storing collections of unique objects)—are available as standard-issue items.
* Like C#, Java, and Go, Python has garbage-collected memory management, meaning the programmer doesn’t have to implement code to track and release objects. Normally garbage collection happens automatically in the background, but if that poses a performance problem, it can be triggered manually or disabled entirely.
* An important aspect of Python is its dynamism. Everything in the language, including functions and modules themselves, are handled as objects. This comes at the expense of speed (more on that below), but makes it far easier to write high-level code. Developers can perform complex object manipulations with only a few instructions, and even treat parts of an application as abstractions that can be altered if needed.
* Python’s use of significant whitespace has been cited as both one of Python’s best and worst attributes. The indentation on the second line shown above isn’t just for readability; it is part of Python’s syntax. Python interpreters will reject programs that don’t use proper indentation to indicate control flow

1. **Back-end**

## Hardware Platform

1. Windows 10, p
2. Python 2.7,
3. 4 GBRAM
4. 1 Tb HDD

## Tools, if any

Microsoft Visual Studio is the main Integrated Development Environment (IDE) from Microsoft. It can be used to develop console and Graphical user interface applications along with Windows Forms applications, web sites, web applications, and web services.

In visual studio we used third-party libraries. Third party libraries are the fastest way to solve your problem. We use pip, pypl and virtual environment support to manage your projects and dependencies.

None of us write perfect code all the time, but when it goes wrong Visual studio can help. Visual step through your code , view or modify state, and interact with your program regardless of the operating system.

# Advantages of this Project

**1.Assibleanytime:**

I’m sure most of you are always kept on hold while operators connect you to a customer care executive. On an average people spend around 7 minutes until they are assigned to a person. Gone are the frustrating days of waiting in a queue for the next available operative. They are replacing live chat and other forms of slower contact methods such as emails and phone calls. Since chatbots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break. This improves your customer UX and helps you rank highly in your sector. Another advantage of this instant response is that you can also skillfully craft your chatbot to maintain your image and brand.

**2. Handling Capacity:**

Unlike humans who can only communicate with one human at a time, chat bots can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered immediately.Imagine you own a restaurant, and you have a good reputation for your food of which most of your revenues come from delivery. As the demand keeps rising, you will have more customers to take orders from but very few staff to attend them all. Having a chatbot would eliminate such problem and cater to each and every person and ensure that no order is missed. Companies like Taco Bell and Dominos are already using

chatbots to arrange delivery of parcels.

**3. Flexible attribute**:

Chatbots have the benefit that it can quite easily be used in any industry. Unlike other products where you have to do a lot of development and testing to change platforms, chatbots are relatively easy to switch. One has to just train the bot by giving the right conversation structure and flow to switch its current field or industry.

Or if there is a lot of back and forth between two sections of the industry say customer support and sales, then you could have custom built presets which would already have the conversation flow and structure to carry out the interactions with the user.

**4. Customer Satisfaction:**

Humans are bound to change of emotions. Chatbots, on the other hand, are bound by some rules and obey them as long as they’re programmed to. They will always treat a customer in the perfect way no matter how rough the person is or how foul language the person uses.Not everyone orders the same food everyday, people’s choices may change everyday. In this case, it can use your order history to make suggestions for the next order, learn your address details and much more. Customers love this smooth interaction and want all their transactions to be as simple as possible.

**5. Cost effective:**

Hiring a human for a job is never a cheap affair, and it will be expensive if your revenue are not high or sales targets are not met and would create havoc in the business. Due to the boundaries of human beings, a single human can only handle one or two people at the same time. More than that would be extremely tough for the employee.Chatbots could help solve this age-old problem. As one chatbot is equal to loads of employees, it can easily communicate with thousands of customers at the same time. We would only need a handful of people to jump into conversations sometimes when necessary. Hence, it would drastically bring down the expenses and bring about a steep rise in revenue and customer satisfaction.

# Assumptions, if any

None

# Future Scope and further enhancement of the Project

There are limitations to what has been currently achieved with chatbots. The limitations of data processing and retrieval are hindering chatbots to reach their full potential. It is not that we lack the computational processing power to do so. However, there is a limitation on “How” we do it. One of the biggest examples is the retail customer market. Retail customers are primarily interested in interacting with humans because of nature of their needs. They don’t want bots to process their needs and respond according to the user need.

Instead of AIML based bot, other algorithms can be implemented. We can include voice-based queries. The users will have to give voice input and the system will give the text output. Also, after successful execution of chatbot in college domain, we can implement it in other domains like medical, forensic, sports, etc. It will be beneficial in all the fields as without spending much time, we are accessing the relevant information and that too without any sorting.

# Project Repository Location

*<Guidelines: Mention the location of the latest Source Code and all related documents, like- Project Synopsis Report, Project Progress updates, Project Requirement Details, Project Report (Softcopy), Test Repository (all test scenarios, test cases etc.) used for Functional Testing of the project etc. The repository location must be somewhere in CCSIT-Lab>*

| **S#** | **Project Artifacts (softcopy)** | **Location** (Mention Lab-ID, Server ID, Folder Name etc.) | **Verified by Project Guide** | **Verified by Lab In-Charge** |
| --- | --- | --- | --- | --- |
|  | Project Synopsis Report (Final Version) |  | Name and Signature | Name and Signature |
|  | Project Progress updates |  | Name and Signature | Name and Signature |
|  | Project Requirement specifications |  | Name and Signature | Name and Signature |
|  | Project Report (Final Version) |  | Name and Signature | Name and Signature |
|  | Test Repository |  | Name and Signature | Name and Signature |
|  | Project Source Code (final version) with executable |  | Name and Signature | Name and Signature |
|  | Any other document |  | Name and Signature | Name and Signature |
|  |  |  |  |  |
|  |  |  |  |  |

# Definitions, Acronyms, and Abbreviations

There are two basic interactive platform for chatbot application: Chatbot(Nihi) and User.

* User give command/ instruction to chatbot application.
* Chatbot(Nihi) have some predefined library to fullfilll the requirement/instruction of user.

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| ERD | Entity-Relationship Diagram |
| DD | Data Dictionary |
| IEEE | Institute of Electrical and Electronic Engineers |
| DFD | Data Flow Diagram |

# 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 enjoying 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

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.

Time and money are one of the most important factors to any organization. Implementing such software in the college stationery department can surely be a profitable deal as this application helps to carry out tasks with ease and thereby reduces time and money on manpower and materials. This is an open source application so that others can edit and transform this system application according to their neds.

# References

*<Guidelines: In this subsection:*

1. *Provide a complete list of all documents referenced elsewhere in the SRS*
2. *Identify each document by title, report number (if applicable), date, and publishing organization*
3. *Specify the sources from which the references can be obtained.*

*This information can be provided by reference to an appendix or to another document. If the application uses specific protocols or RFC’s, then reference them here so designers know where to find them.*>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S#** | **Reference Details** | **Owner** | **Version** | **Date** |
|  | Project Synopsis | <Project Group ID> | 5.0 | DD-MM-YY |
|  | Project Requirements | <Project Group ID> |  | 06/10/2022 |
|  |  |  |  |  |

**Annexure A**

**Data Flow Diagram (DFD)**

**(Mandatory)**

matching

conversation

heard

matching

Load active conversation

received

Get message

Sending

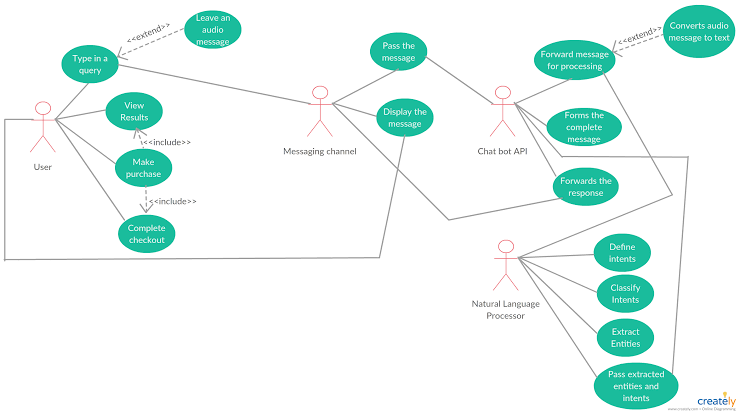
Call matching method

Send bot response

**Annexure B**

**Entity-Relationship Diagram (ERD)**

**(Mandatory)**

****

**Annexure C**

**Use-Case Diagram (UCD)**

**(Optional)**

**Annexure D**

**Data Dictionary (DD)**

**(Mandatory)**

**Example:**

**User Table (USR)**

|  |  |  |
| --- | --- | --- |
| **Fields** | **Data type** | **Description** |
| USR-Name | Text | Admin name |
| USR-Password | Text | Admin password |
| USR-Contact-No | Number | Admin Contact |
| USR-Address | Text | City |

**Supplier Table (SUPP)**

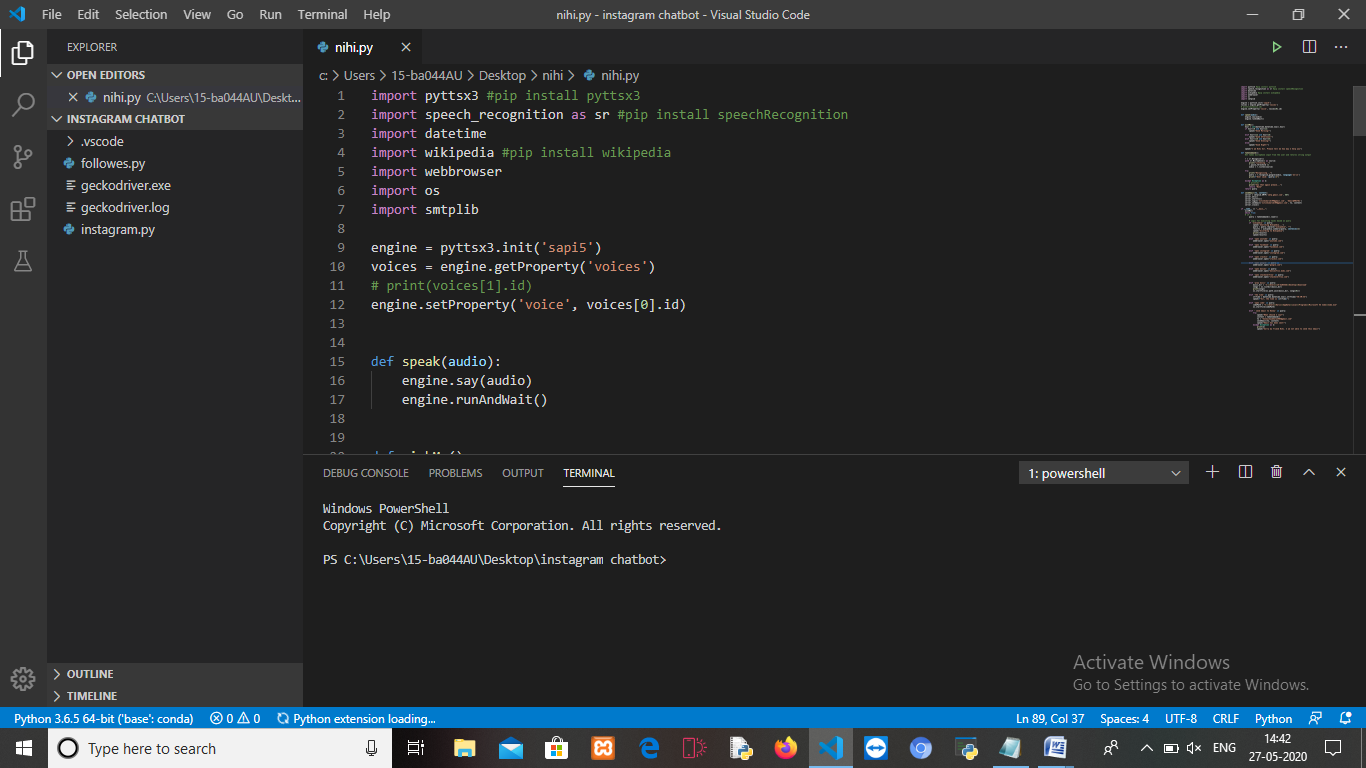
|  |  |  |
| --- | --- | --- |
| **Fields** | **Data type** | **Description** |
| SUPP-ID | Number | Supplier ID |
| SUPP-Name | Text | Supplier Name |
| SUPP-Address | Text | Supplier Address |
| SUPP-Contact | Number | Supplier Contact |
| SUPP-Credit-Limit | Number | Credit Limit |

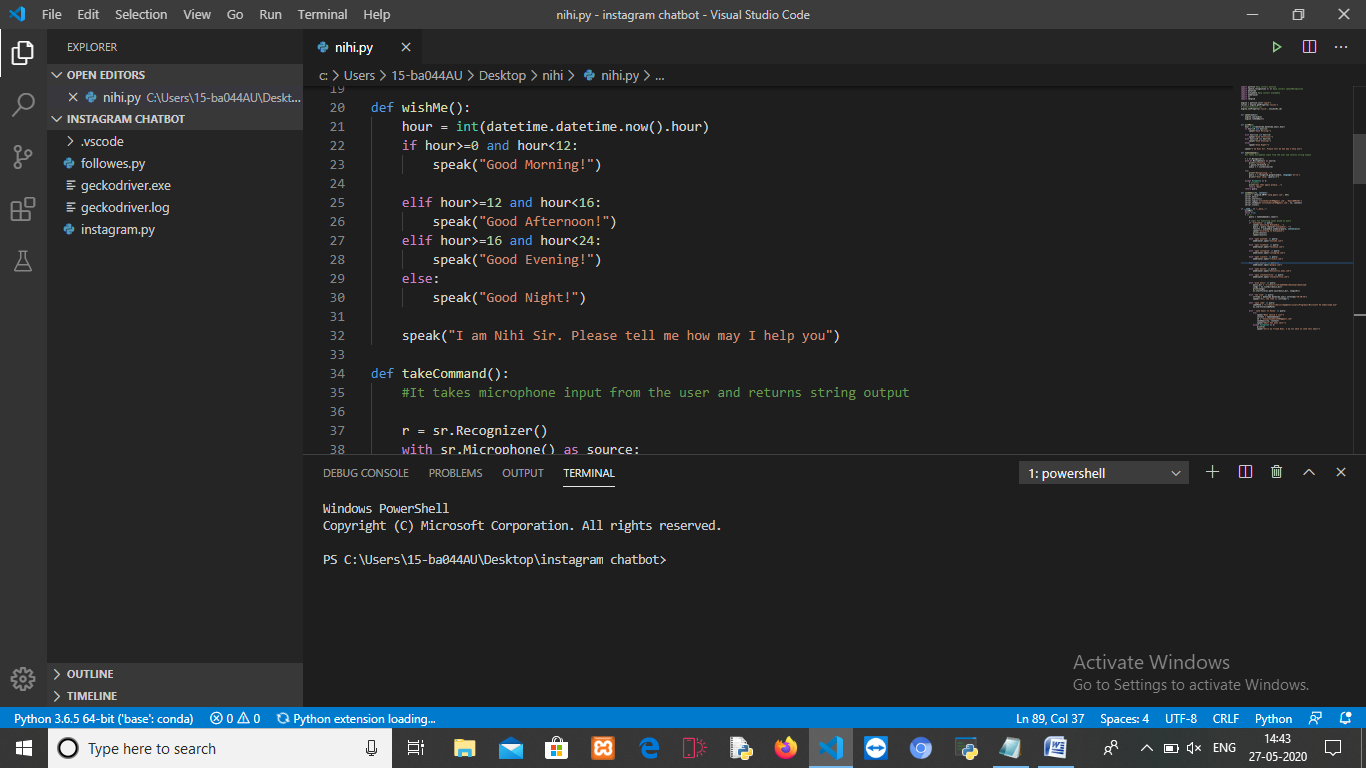
**Annexure E**

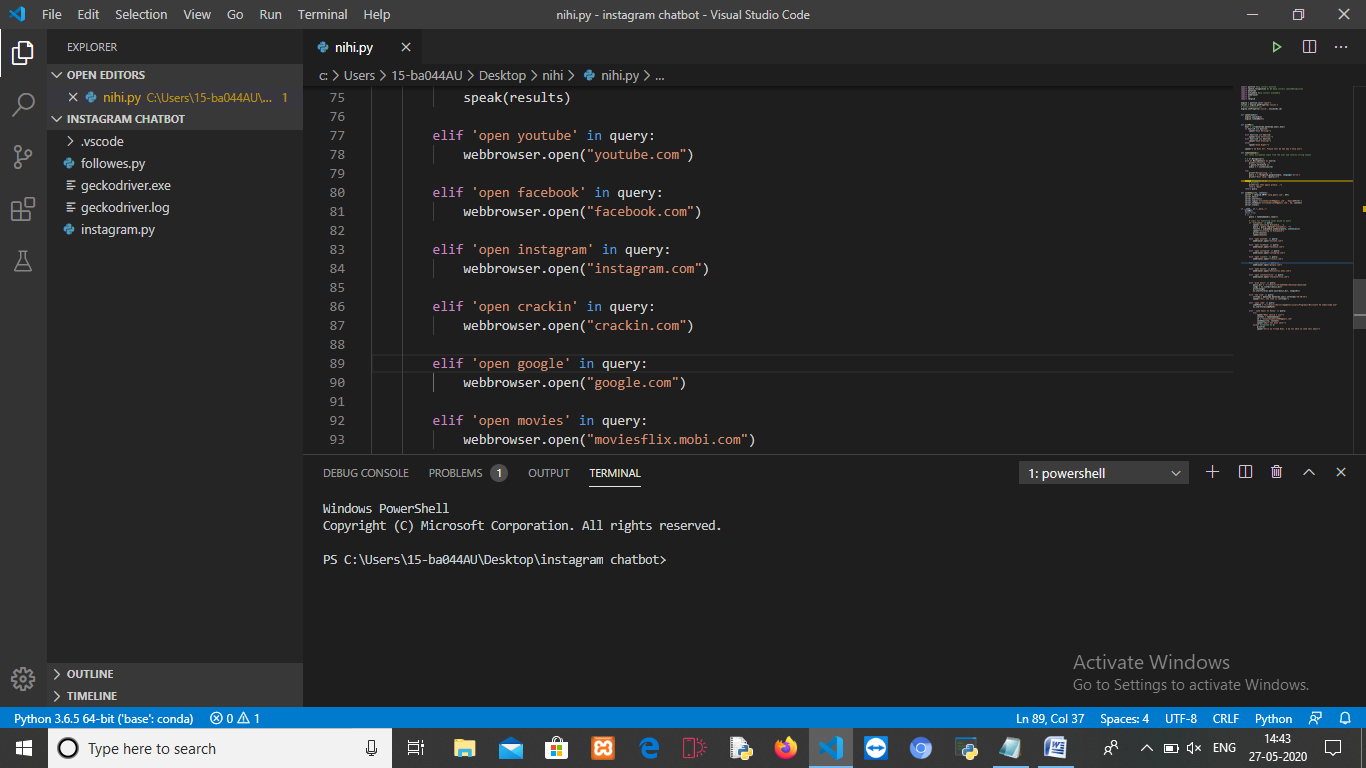
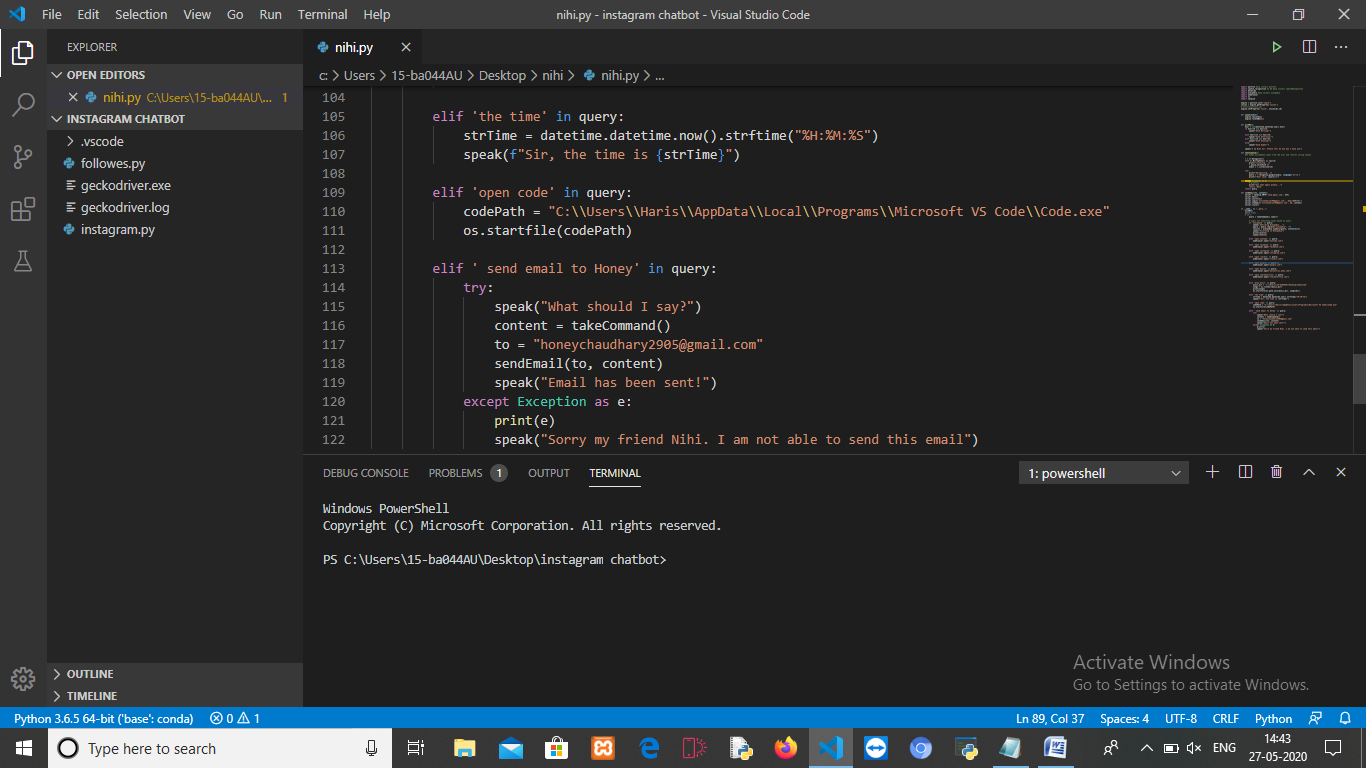
**Screen Shots**

*<Guidelines: Show all Pages>*

**Home Page:**







*Project code*

import subprocess

import wolframalpha

import pyttsx3

import tkinter

import json

import random

import operator

import datetime

import wikipedia

import webbrowser

import os

import winshell

import pyjokes

import feedparser

import smtplib

import ctypes

import time

import requests

import shutil

from twilio.rest import Client

from clint.textui import progress

from ecapture import ecapture as ec

from bs4 import BeautifulSoup

import win32com.client as wincl

from urllib.request import urlopen

engine = pyttsx3.init('sapi5')

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[1].id)

def speak(audio):

    engine.say(audio)

    engine.runAndWait()

def wishMe():

    hour = int(datetime.datetime.now().hour)

    if hour>= 0 and hour<12:

        speak("Good Morning Sir !")

    elif hour>= 12 and hour<18:

        speak("Good Afternoon Sir !")

    else:

        speak("Good Evening Sir !")

    assname =("I am Nihi")

    speak("I am your Assistant")

    speak(assname)

def usrname():

    speak("What should i call you sir")

    uname = takeCommand()

    speak("Welcome Mister")

    speak(uname)

    columns = shutil.get\_terminal\_size().columns

    print("#####################".center(columns))

    print("Welcome Mr.", uname.center(columns))

    print("#####################".center(columns))

    speak("How can i Help you, Sir")

def takeCommand():

    r = sr.Recognizer()

    with sr.Microphone() as source:

        print("Listening...")

        r.pause\_threshold = 1

        audio = r.listen(source)

    try:

        print("Recognizing...")

        query = r.recognize\_google(audio, language ='en-in')

        print(f"User said: {query}\n")

    except Exception as e:

        print(e)

        print("Unable to Recognizing your voice.")

        return "None"

    return query

def sendEmail(to, content):

    server = smtplib.SMTP('smtp.gmail.com', 587)

    server.ehlo()

    server.starttls()

    # Enable low security in gmail

    server.login('your email id', 'your email passowrd')

    server.sendmail('your email id', to, content)

    server.close()

if \_\_name\_\_ == '\_\_main\_\_':

    clear = lambda: os.system('cls')

    # This Function will clean any

    # command before execution of this python file

    clear()

    wishMe()

    usrname()

    while True:

        query = takeCommand().lower()

        # All the commands said by user will be

        # stored here in 'query' and will be

        # converted to lower case for easily

        # recognition of command

        if 'wikipedia' in query:

            speak('Searching Wikipedia...')

            query = query.replace("wikipedia", "")

            results = wikipedia.summary(query, sentences = 3)

            speak("According to Wikipedia")

            print(results)

            speak(results)

        elif 'open youtube' in query:

            speak("Here you go to Youtube\n")

            webbrowser.open("youtube.com")

        elif 'open google' in query:

            speak("Here you go to Google\n")

            webbrowser.open("google.com")

        elif 'open stackoverflow' in query:

            speak("Here you go to Stack Over flow.Happy coding")

            webbrowser.open("stackoverflow.com")

        elif 'play music' in query or "play song" in query:

            speak("Here you go with music")

            # music\_dir = "G:\\Song"

            music\_dir = "C:\\Users\\Nitin\\Music"

            songs = os.listdir(music\_dir)

            print(songs)

            random = os.startfile(os.path.join(music\_dir, songs[1]))

        elif 'the time' in query:

            strTime = datetime.datetime.now().strftime("% H:% M:% S")

            speak(f"Sir, the time is {strTime}")

        elif 'open opera' in query:

            codePath = r"C:\\Users\\Nitin\\AppData\\Local\\Programs\\Opera\\launcher.exe"

            os.startfile(codePath)

        elif 'email to Nitin' in query:

            try:

                speak("What should I say?")

                content = takeCommand()

                to = "Receiver email address"

                sendEmail(to, content)

                speak("Email has been sent !")

            except Exception as e:

                print(e)

                speak("I am not able to send this email")

        elif 'send a mail' in query:

            try:

                speak("What should I say?")

                content = takeCommand()

                speak("whome should i send")

                to = input()

                sendEmail(to, content)

                speak("Email has been sent !")

            except Exception as e:

                print(e)

                speak("I am not able to send this email")

        elif 'how are you' in query:

            speak("I am fine, Thank you")

            speak("How are you, Sir")

        elif 'fine' in query or "good" in query:

            speak("It's good to know that your fine")

        elif "change my name to" in query:

            query = query.replace("change my name to", "")

            assname = query

        elif "change name" in query:

            speak("What would you like to call me, Sir ")

            assname = takeCommand()

            speak("Thanks for naming me")

        elif "what's your name" in query or "What is your name" in query:

            speak("My friends call me")

            speak(assname)

            print("My friends call me", assname)

        elif 'exit' in query:

            speak("Thanks for giving me your time")

            exit()

        elif "who made you" in query or "who created you" in query:

            speak("I have been created by Nitin.")

        elif 'joke' in query:

            speak(pyjokes.get\_joke())

        elif "calculate" in query:

            app\_id = "Wolframalpha api id"

            client = wolframalpha.Client(app\_id)

            indx = query.lower().split().index('calculate')

            query = query.split()[indx + 1:]

            res = client.query(' '.join(query))

            answer = next(res.results).text

            print("The answer is " + answer)

            speak("The answer is " + answer)

        elif 'search' in query or 'play' in query:

            query = query.replace("search", "")

            query = query.replace("play", "")

            webbrowser.open(query)

        elif "who i am" in query:

            speak("If you talk then definately your human.")

        elif "why you came to world" in query:

            speak("Thanks to Nitin. further It's a secret")

        elif 'power point presentation' in query:

            speak("opening Power Point presentation")

            power = r"C:\\Users\\Nitin\\Desktop\\Minor Project\\Presentation\\Voice Assistant.pptx"

            os.startfile(power)

        elif 'is love' in query:

            speak("It is 7th sense that destroy all other senses")

        elif "who are you" in query:

            speak("I am your virtual assistant created by Nitin")

        elif 'reason for you' in query:

            speak("I was created as a Minor project by Mister Nitin ")

        elif 'change background' in query:

            ctypes.windll.user32.SystemParametersInfoW(20,

                                                    0,

                                                    "Location of wallpaper",

                                                    0)

            speak("Background changed succesfully")

        elif 'open bluestack' in query:

            appli = r"C:\\ProgramData\\BlueStacks\\Client\\Bluestacks.exe"

            os.startfile(appli)

        elif 'news' in query:

            try:

                jsonObj = urlopen('''https://newsapi.org / v1 / articles?source = the-times-of-india&sortBy = top&apiKey =\\times of India Api key\\''')

                data = json.load(jsonObj)

                i = 1

                speak('here are some top news from the times of india')

                print('''=============== TIMES OF INDIA ============'''+ '\n')

                for item in data['articles']:

                    print(str(i) + '. ' + item['title'] + '\n')

                    print(item['description'] + '\n')

                    speak(str(i) + '. ' + item['title'] + '\n')

                    i += 1

            except Exception as e:

                print(str(e))

        elif 'lock window' in query:

                speak("locking the device")

                ctypes.windll.user32.LockWorkStation()

        elif 'shutdown system' in query:

                speak("Hold On a Sec ! Your system is on its way to shut down")

                subprocess.call('shutdown / p /f')

        elif 'empty recycle bin' in query:

            winshell.recycle\_bin().empty(confirm = False, show\_progress = False, sound = True)

            speak("Recycle Bin Recycled")

        elif "don't listen" in query or "stop listening" in query:

            speak("for how much time you want to stop jarvis from listening commands")

            a = int(takeCommand())

            time.sleep(a)

            print(a)

        elif "where is" in query:

            query = query.replace("where is", "")

            location = query

            speak("User asked to Locate")

            speak(location)

            webbrowser.open("https://www.google.nl / maps / place/" + location + "")

        elif "camera" in query or "take a photo" in query:

            ec.capture(0, "Jarvis Camera ", "img.jpg")

        elif "restart" in query:

            subprocess.call(["shutdown", "/r"])

        elif "hibernate" in query or "sleep" in query:

            speak("Hibernating")

            subprocess.call("shutdown / h")

        elif "log off" in query or "sign out" in query:

            speak("Make sure all the application are closed before sign-out")

            time.sleep(5)

            subprocess.call(["shutdown", "/l"])

        elif "write a note" in query:

            speak("What should i write, sir")

            note = takeCommand()

            file = open('jarvis.txt', 'w')

            speak("Sir, Should i include date and time")

            snfm = takeCommand()

            if 'yes' in snfm or 'sure' in snfm:

                strTime = datetime.datetime.now().strftime("% H:% M:% S")

                file.write(strTime)

                file.write(" :- ")

                file.write(note)

            else:

                file.write(note)

        elif "show note" in query:

            speak("Showing Notes")

            file = open("jarvis.txt", "r")

            print(file.read())

            speak(file.read(6))

        # NPPR9-FWDCX-D2C8J-H872K-2YT43

        elif "jarvis" in query:

            wishMe()

            speak("Jarvis 1 point o in your service Mister")

            speak(assname)

        elif "weather" in query:

            # Google Open weather website

            # to get API of Open weather

            api\_key = "Api key"

            base\_url = "http://api.openweathermap.org / data / 2.5 / weather?"

            speak(" City name ")

            print("City name : ")

            city\_name = takeCommand()

            complete\_url = base\_url + "appid =" + api\_key + "&q =" + city\_name

            response = requests.get(complete\_url)

            x = response.json()

            if x["cod"] != "404":

                y = x["main"]

                current\_temperature = y["temp"]

                current\_pressure = y["pressure"]

                current\_humidiy = y["humidity"]

                z = x["weather"]

                weather\_description = z[0]["description"]

                print(" Temperature (in kelvin unit) = " +str(current\_temperature)+"\n atmospheric pressure (in hPa unit) ="+str(current\_pressure) +"\n humidity (in percentage) = " +str(current\_humidiy) +"\n description = " +str(weather\_description))

            else:

                speak(" City Not Found ")

        elif "wikipedia" in query:

            webbrowser.open("wikipedia.com")

        elif "Good Morning" in query:

            speak("A warm" +query)

            speak("How are you Mister")

            speak(assname)

        # most asked question from google Assistant

        elif "will you be my gf" in query or "will you be my bf" in query:

            speak("I'm not sure about, may be you should give me some time")

        elif "how are you" in query:

            speak("I'm fine, glad you me that")

        elif "i love you" in query:

            speak("It's hard to understand")

        elif "what is" in query or "who is" in query:

            # Use the same API key

            # that we have generated earlier

            client = wolframalpha.Client("API\_ID")

            res = client.query(query)

            try:

                print (next(res.results).text)

                speak (next(res.results).text)

            except StopIteration:

                print ("No results")

        # elif "" in query:

            # Command go here

            # For adding more commands