ABSTRACT

In this modern era, day to day life became smarter and interlinked with technology. Now in our voice assistance system, it can Recognise speech & Provide Wikipedia/Information, play songs which the user Requests, tell date & time when asked. This project works on voice input and give output through voice and displays the text on the screen. The main agenda of our voice assistance makes people smart and give instant and computed results. The voice assistance takes the voice input through our microphone (Bluetooth and wired microphone) and it converts our voice into computer understandable language gives the required solutions and answers which are asked by the user.

INTRODUCTION

Today the development of artificial intelligence (AI) systems that can organize a natural human-machine interaction (through voice, communication, gestures, facial expressions, etc.) are gaining in popularity. One of the most studied and popular was the direction of interaction, based on the understanding of the machine by the machine of the natural human language. It is no longer a human who learns to communicate with a machine, but a machine learns to communicate with a human, exploring his actions, habits, behaviour and trying to become his personalized assistant Virtual assistants are software programs that help you ease your day to day tasks, such as showing weather reports, creating remainders, making shopping lists etc. They can take commands via text (online chatbots) or by voice. Voice-based intelligent assistants need an invoking word or wake word to activate the listener, followed by the command. We have so many virtual assistants, such as Apple's Siri, Amazon's Alexa and Microsoft's Cortana.

This project was started on the premise that there is a sufficient amount of openly available data and information on the web that can be utilized to build a virtual assistant that has access to making intelligent decisions for routine user activities.

Keywords

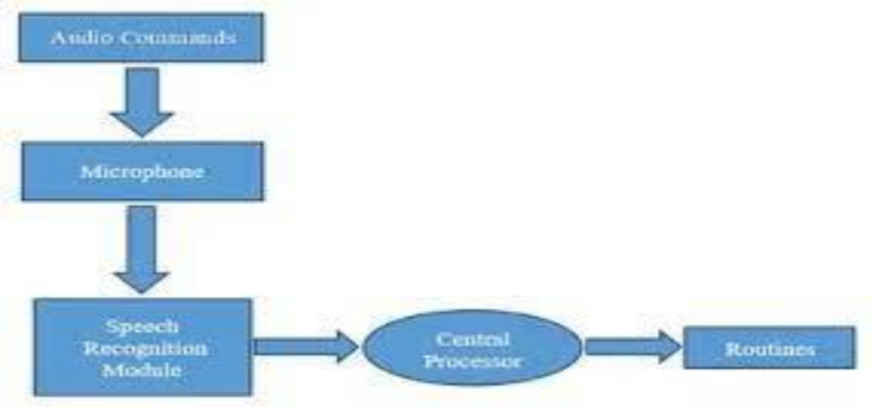
Virtual Assistant Using Python, AI, Virtual Assistance, Python.

PROBLEM STATEMENT

PROPOSED SYSTEM

The work started with analysing the audio commands given by the user through the microphone. This can be anything like getting any information, etc. This is an empirical qualitative study, based on reading above mentioned literature and testing their examples. Tests are made by programming according to books and online resources, with the explicit goal to find best practices and a more advanced understanding of Voice Assistant.

The system uses Google's online speech recognition system for converting speech input to text. The speech input Users can obtain texts from the special corpora organized on the computer network server at the information centre from the microphone is temporarily stored in the system which is then sent to Google cloud for speech recognition. The equivalent text is then received and fed to the central processor.



SOFTWARE AND HARDWARE ENVIRONMENT

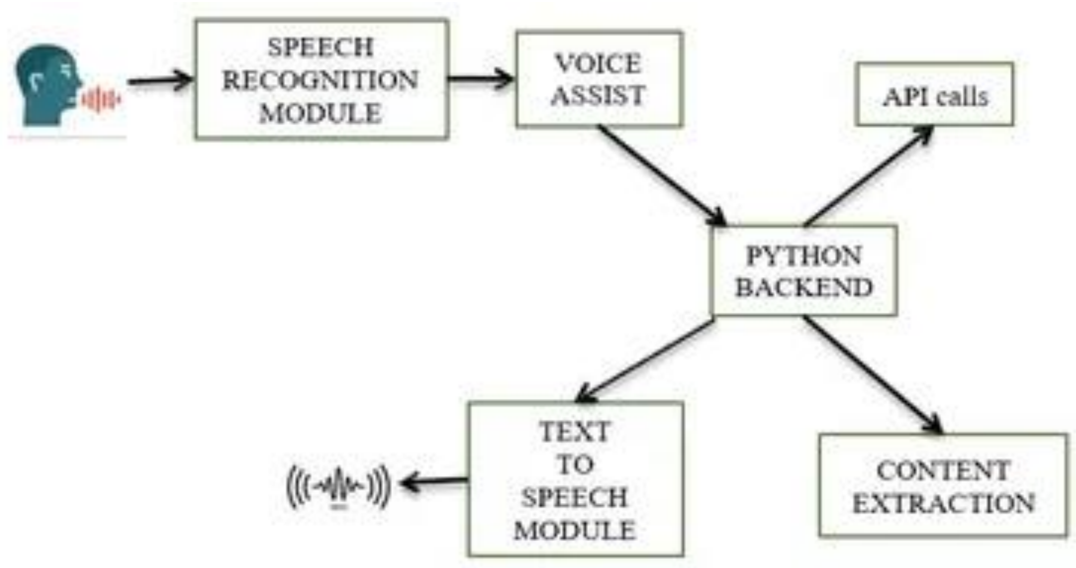
Software Requirements

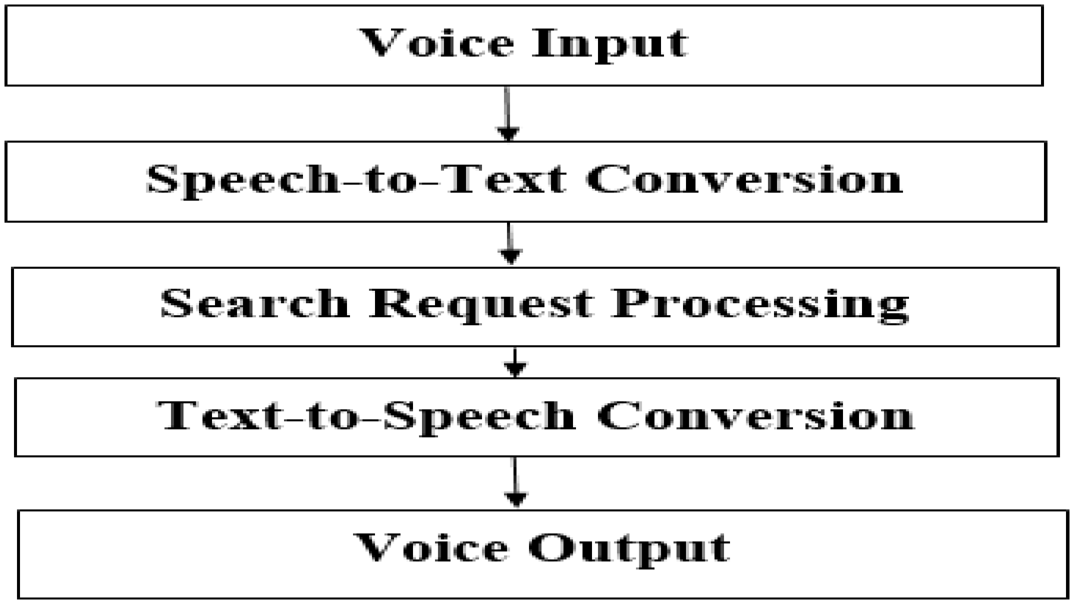
- Operating system : Windows XP/7/10
- Software : PyCharm

Hardware Requirements

- System : Pentium(min)
- Hard Disk : 500 GB.(min)
- Ram : 4 GB(min)

SYSTEM DESIGN




Voice Input
Speech-to-Text Conversion
Search Request Processing
Text-to-Speech Conversion
Voice Output

IMPLEMENTATION

Python Backend

The python backend gets the output from the speech recognition module and then identifies whether the command or the speech output is an API Call and Context Extraction. The output is then sent back to the python backend to give the required output to the user.

Content Extraction

Context extraction (CE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents. In most cases, this activity concerns processing human language texts using natural language processing (NLP). Recent activities in multimedia document processing like automatic annotation and content extraction out of images/audio/video could be seen as context extraction TEST RESULTS.

Text-to-speech module

Text-to-Speech (TTS) refers to the ability of computers to read text aloud. A TTS Engine converts written text to a phonemic representation, then converts the phonemic representation to waveforms that can be output as sound. TTS engines with different languages, dialects and specialized vocabularies are available through third-party publishers.

Libraries used

Speech Recognition module

The system uses Google's online speech recognition system for converting speech input to text. The speech input Users can obtain texts from the special corpora organized on the computer network server at the information centre from the microphone is temporarily stored in the system which is then sent to Google cloud for speech recognition. The equivalent text is then received and fed to the central processor.

Library for performing speech recognition, with support for several engines and APIs, online and offline.

Speech recognition engine/API support:

- Google Speech Recognition
- Google Cloud Speech API

API calls

API stands for Application Programming Interface. An API is a software intermediary that allows two applications to talk to each other. In other words, an API is a messenger that delivers your request to the provider that you're requesting it from and then delivers the response back to you.

Pyttsx3

Pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

PyWhatKit

PyWhatKit is a Python library with various helpful features. It's easy-touse and does not require you to do any additional setup. Currently, it has about 300k+ downloads and counting. New updates are released frequently with new features and bug fixes.

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.

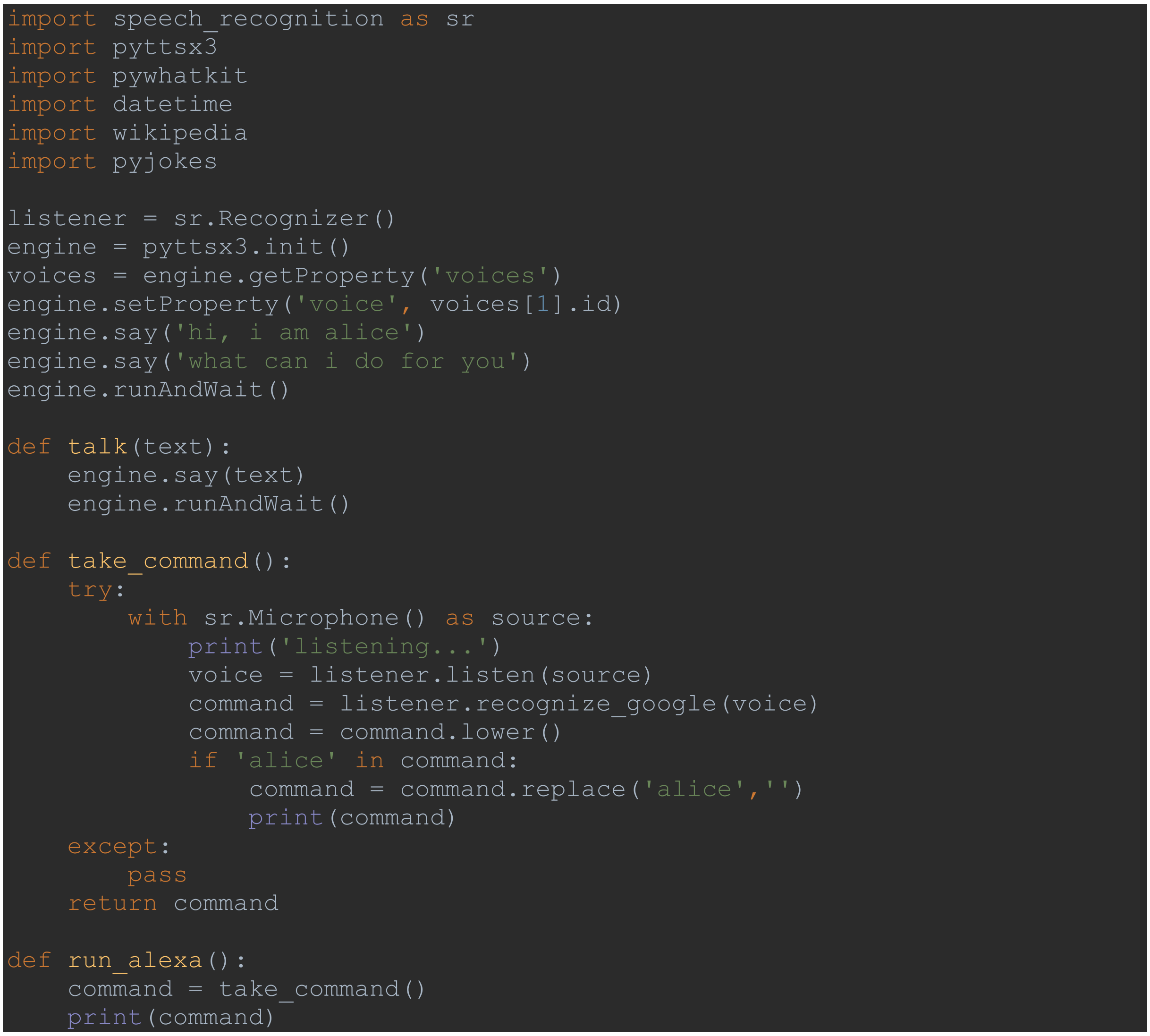
PYjokes

One-line jokes for programmers (jokes as a service).

PyAudio

PyAudio is a set of Python bindings for Port Audio, a cross-platform C++ library interfacing with audio drivers.

PYTHON CODE



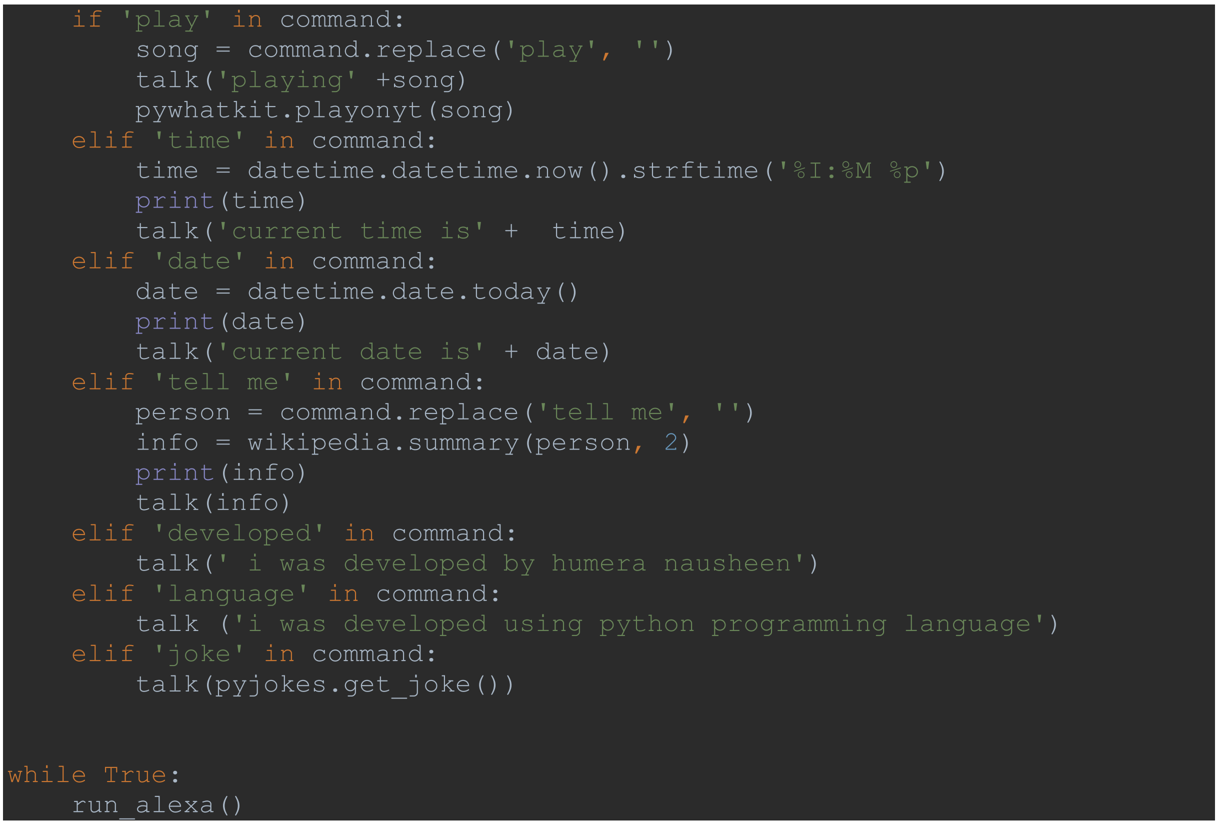
```
import speech_recognition as sr
import pyttsx3
import pywhatkit
import datetime
import wikipedia
import pyjokes

listener = sr.Recognizer()
engine = pyttsx3.init()
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)
engine.say('hi, i am alice')
engine.say('what can i do for you')
engine.runAndWait()

def talk(text):
    engine.say(text)
    engine.runAndWait()

def take_command():
    try:
        with sr.Microphone() as source:
            print('listening...')
            voice = listener.listen(source)
            command = listener.recognize_google(voice)
            command = command.lower()
            if 'alice' in command:
                command = command.replace('alice','')
                print(command)
    except:
        pass
    return command

def run_alexa():
    command = take_command()
    print(command)
```



```
    if 'play' in command:
        song = command.replace('play', '')
        talk('playing' +song)
        pywhatkit.playonyt(song)
    elif 'time' in command:
        time = datetime.datetime.now().strftime('%I:%M %p')
        print(time)
        talk('current time is' +  time)
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
        talk('current date is' + date)
    elif 'tell me' in command:
        person = command.replace('tell me', '')
        info = wikipedia.summary(person, 2)
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
        talk ('i was developed using python programming language')
    elif 'joke' in command:
        talk(pyjokes.get_joke())


while True:
    run_alexa()
```

TESTING

TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module’s control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All-important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing

Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches test each module individually and then each module is module is integrated with a main module and tested for functionality.

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error message. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true system test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

TESTING STRATEGY

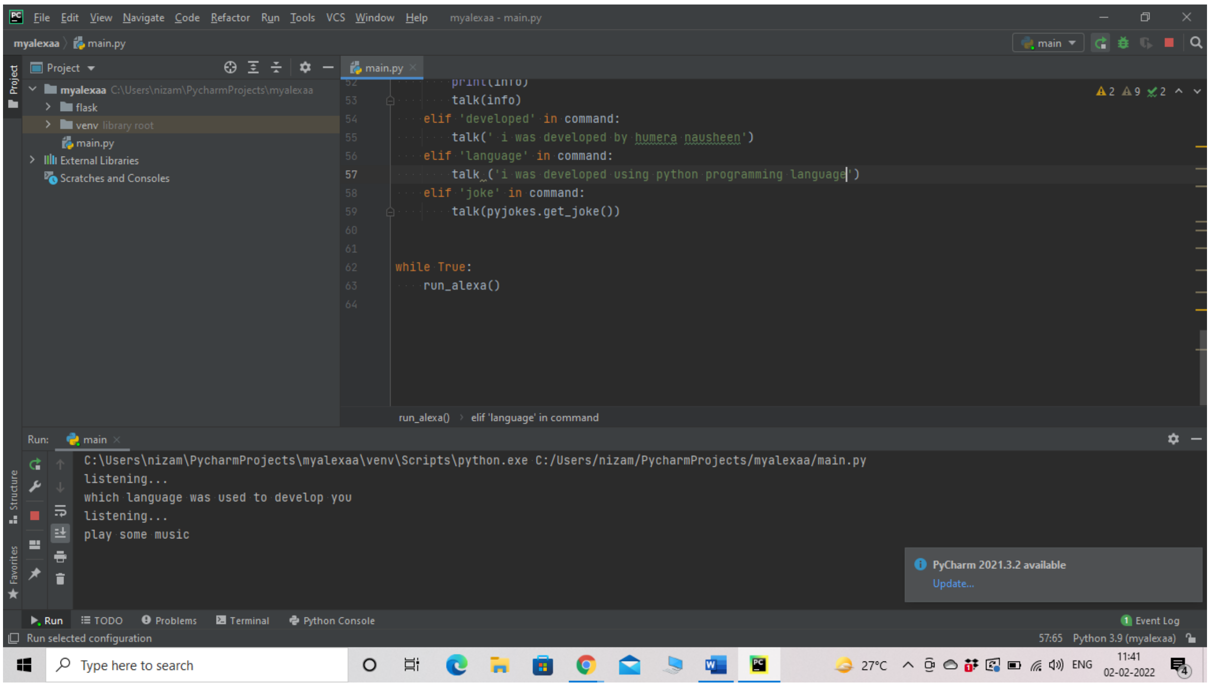
A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

RESULT



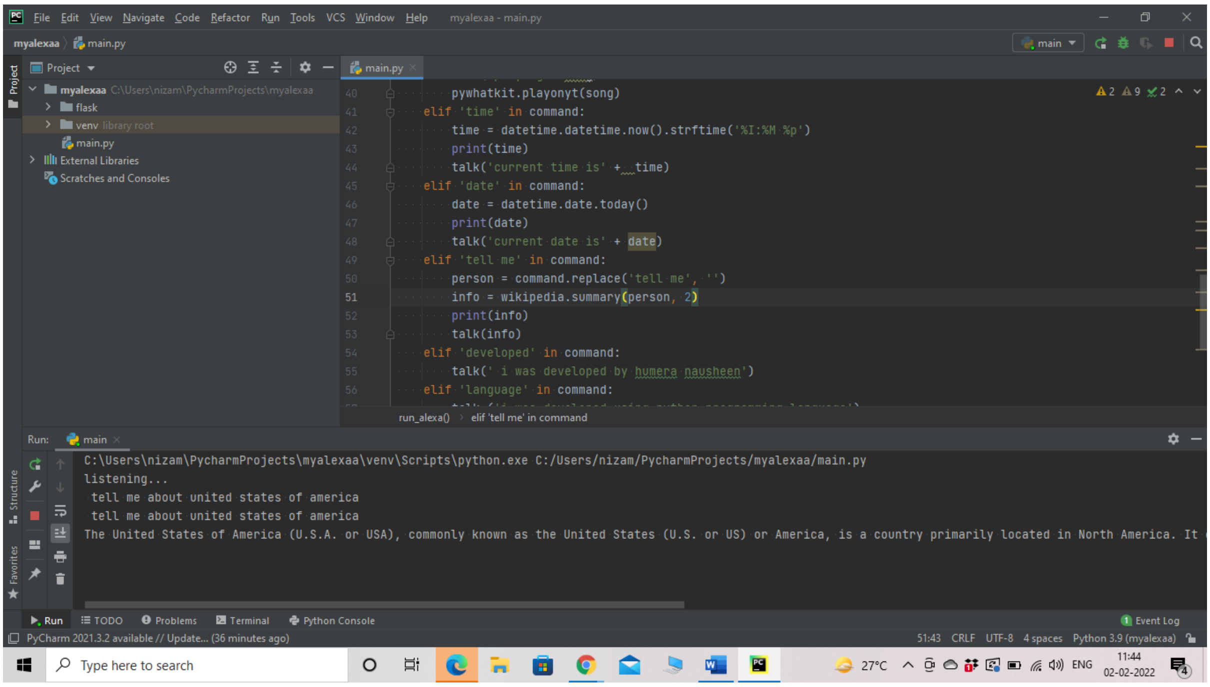
```
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
        talk_('i was developed using python programming language')
    elif 'joke' in command:
        talk(pyjokes.get_joke())


while True:
    run_alexa()
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
which language was used to develop you
listening...
play some music
```

PyCharm 2021.3.2 available
Update...

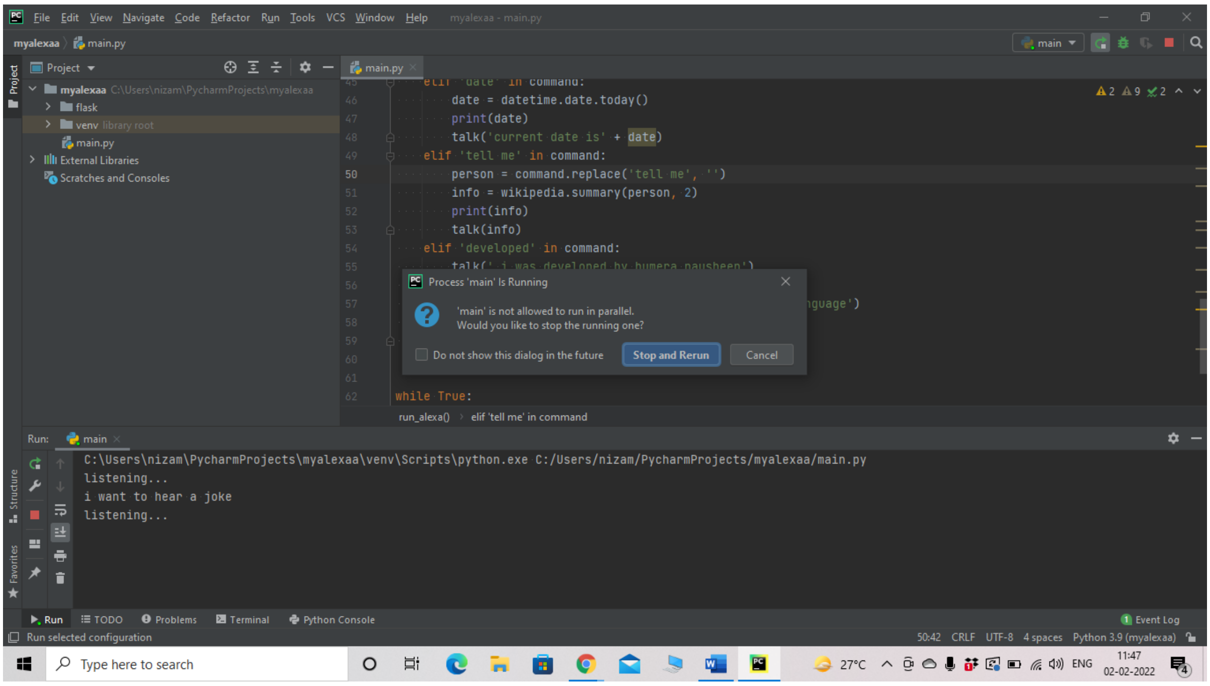
File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
run_alexa() elif 'language' in command
Run: main
Run TODO Problems Terminal Python Console Event Log
Run selected configuration
57:65 Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:41 02-02-2022



```
        pywhatkit.playonyt(song)
    elif 'time' in command:
        time = datetime.datetime.now().strftime('%I:%M %p')
        print(time)
        talk('current time is' + time)
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
        talk('current date is' + date)
    elif 'tell me' in command:
        person = command.replace('tell me', '')
        info = wikipedia.summary(person, 2)
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
 tell me about united states of america
 tell me about united states of america
The United States of America (U.S.A. or USA), commonly known as the United States (U.S. or US) or America, is a country primarily located in North America. It
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
run_alexa() elif 'tell me' in command
Run: main
Run TODO Problems Terminal Python Console Event Log
PyCharm 2021.3.2 available // Update... (36 minutes ago)
51:43 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:44 02-02-2022



```
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
        talk('current date is' + date)
    elif 'tell me' in command:
        person = command.replace('tell me', '')
        info = wikipedia.summary(person, 2)
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
```

Process 'main' Is Running

'main' is not allowed to run in parallel.
Would you like to stop the running one?

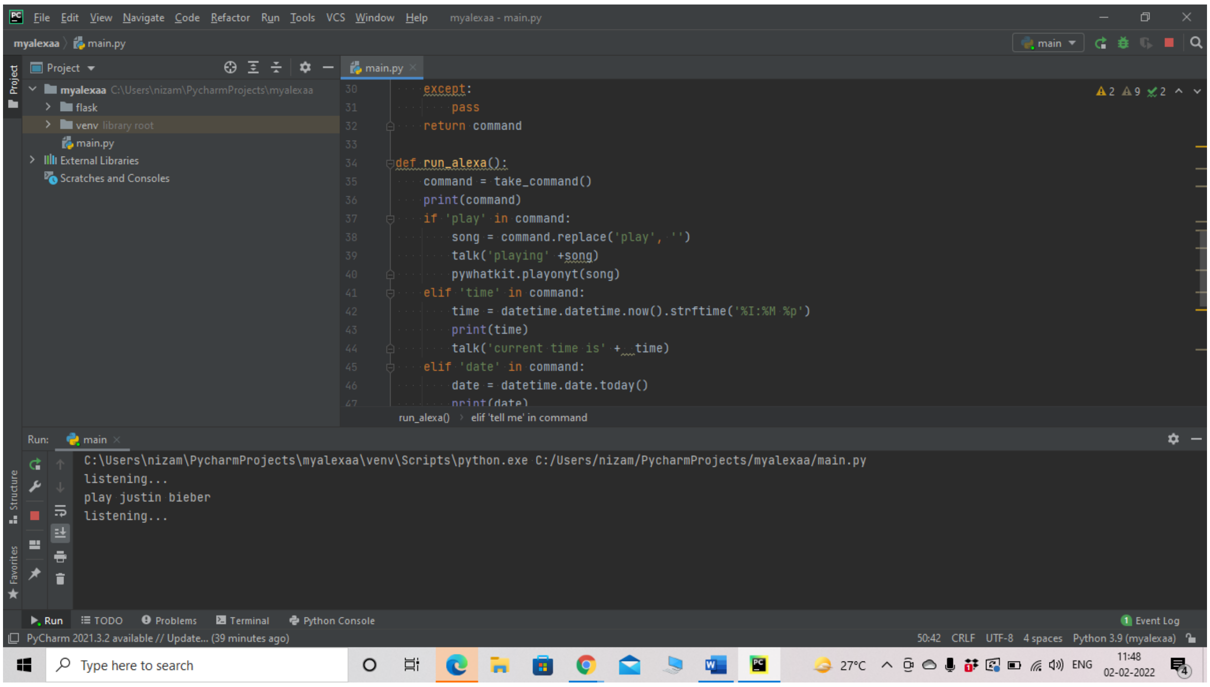
□ Do not show this dialog in the future

```
nguage')
```

```
while True:
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
i want to hear a joke
listening...
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
main
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
Stop and Rerun
Cancel
run_alexa() › elif 'tell me' in command
Run: main
Run TODO Problems Terminal Python Console Event Log
Run selected configuration
50:42 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:47 02-02-2022

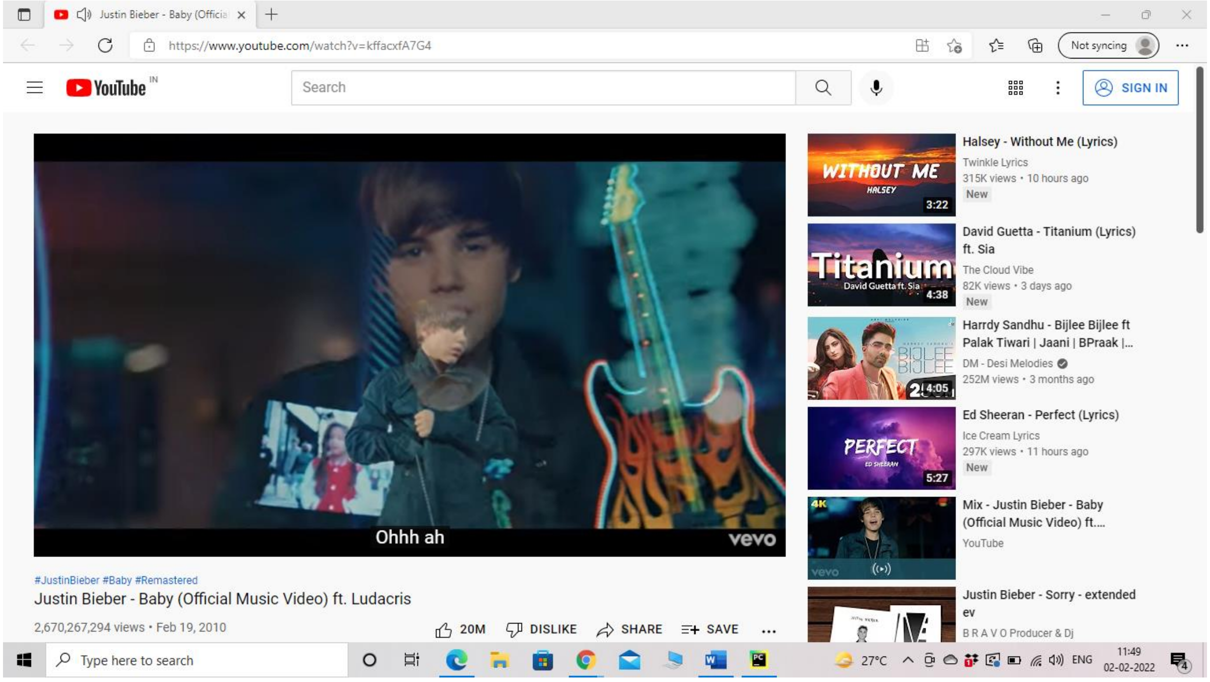


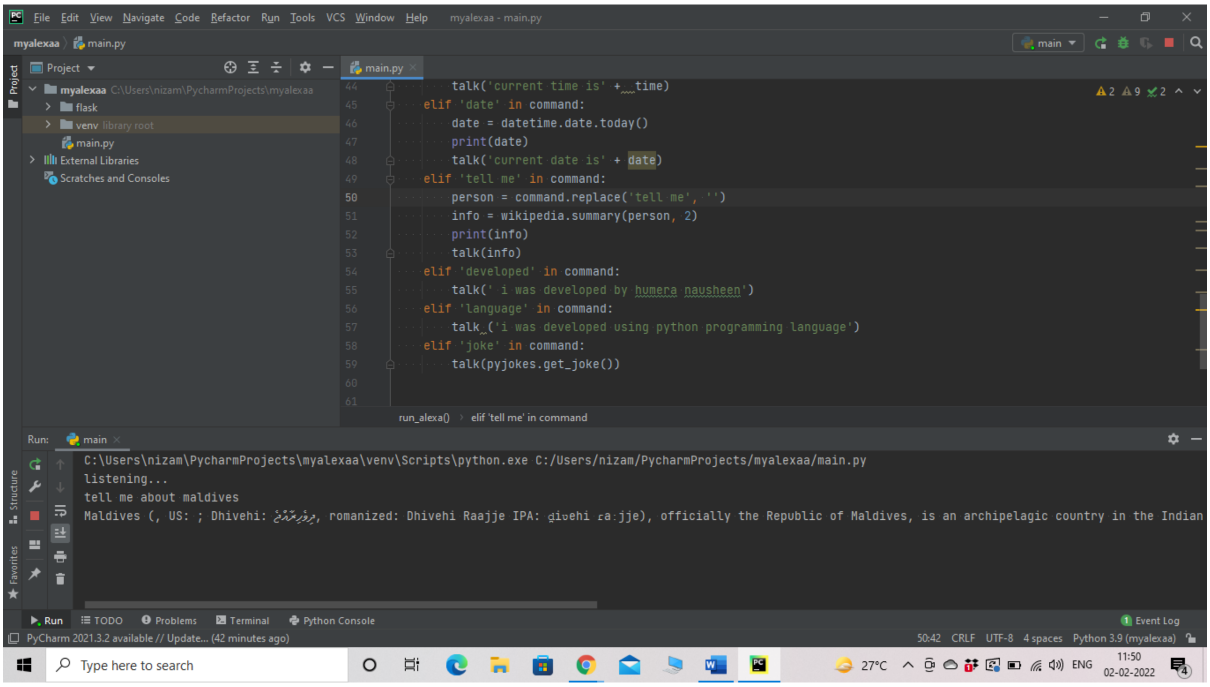
```
    except:
        pass
    return command

def run_alexa():
    command = take_command()
    print(command)
    if 'play' in command:
        song = command.replace('play', '')
        talk('playing' +song)
        pywhatkit.playonyt(song)
    elif 'time' in command:
        time = datetime.datetime.now().strftime('%I:%M %p')
        print(time)
        talk('current time is' +  time)
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
play justin bieber
listening...
```

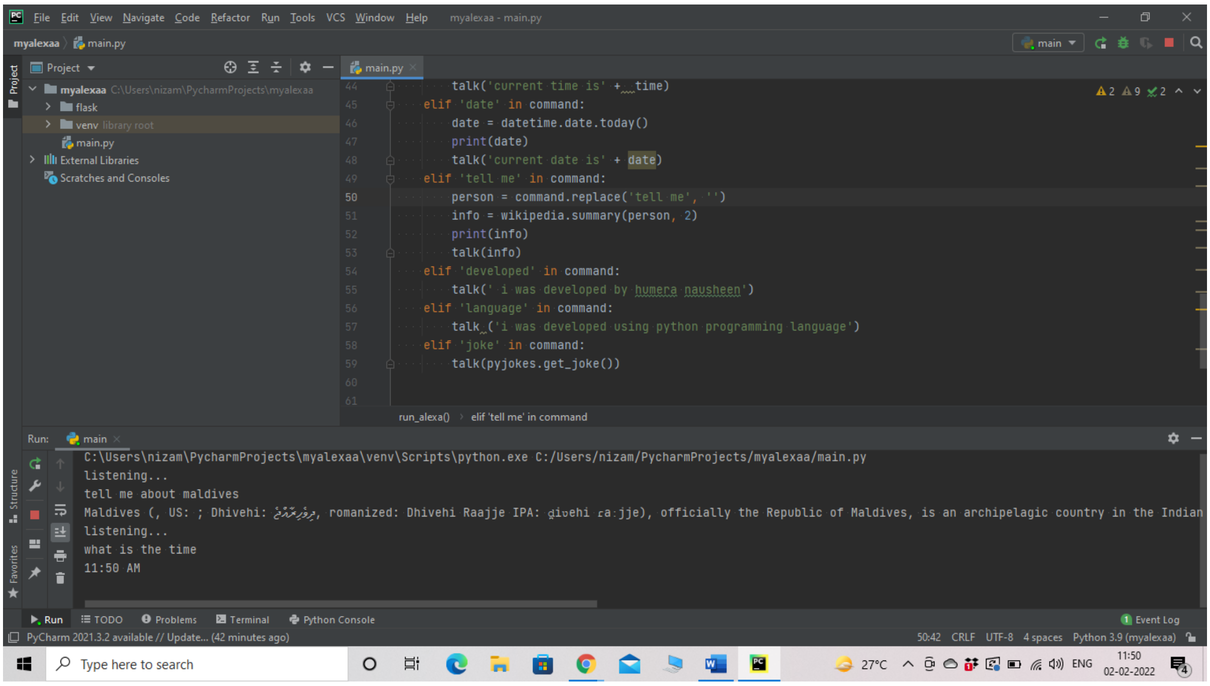
File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
main
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
run_alexa() › elif 'tell me' in command
Run: main
Run TODO Problems Terminal Python Console Event Log
PyCharm 2021.3.2 available // Update... (39 minutes ago)
50:42 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:48 02-02-2022


Justin Bieber - Baby (Officia
https://www.youtube.com/watch?v=kffacfA7G4
Not syncing
YouTube
Search
SIGN IN
Ohhh ah
vevo
#JustinBieber #Baby #Remastered
Justin Bieber - Baby (Official Music Video) ft. Ludacris
2,670,267,294 views • Feb 19, 2010
20M
DISLIKE
SHARE
SAVE
Halsey - Without Me (Lyrics)
Twinkle Lyrics
315K views • 10 hours ago
David Guetta - Titanium (Lyrics) ft. Sia
The Cloud Vibe
82K views • 3 days ago
Harrdy Sandhu - Bijlee Bijlee ft Palak Tiwari | Jaani | BPraak |...
DM - Desi Melodies
252M views • 3 months ago
Ed Sheeran - Perfect (Lyrics)
Ice Cream Lyrics
297K views • 11 hours ago
Mix - Justin Bieber - Baby (Official Music Video) ft....
YouTube
Justin Bieber - Sorry - extended ev
B R A V O Producer & Dj
Type here to search
27°C
ENG
11:49
02-02-2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa
main.py
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
run_alexa() › elif 'tell me' in command
Run: main
Run
TODO
Problems
Terminal
Python Console
Event Log
PyCharm 2021.3.2 available // Update... (42 minutes ago)
50:42 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C
ENG
11:50
02-02-2022

```
        talk('current time is' + time)
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
        talk('current date is' + date)
    elif 'tell me' in command:
        person = command.replace('tell me', '')
        info = wikipedia.summary(person, 2)
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
        talk ('i was developed using python programming language')
    elif 'joke' in command:
        talk(pyjokes.get_joke())
```

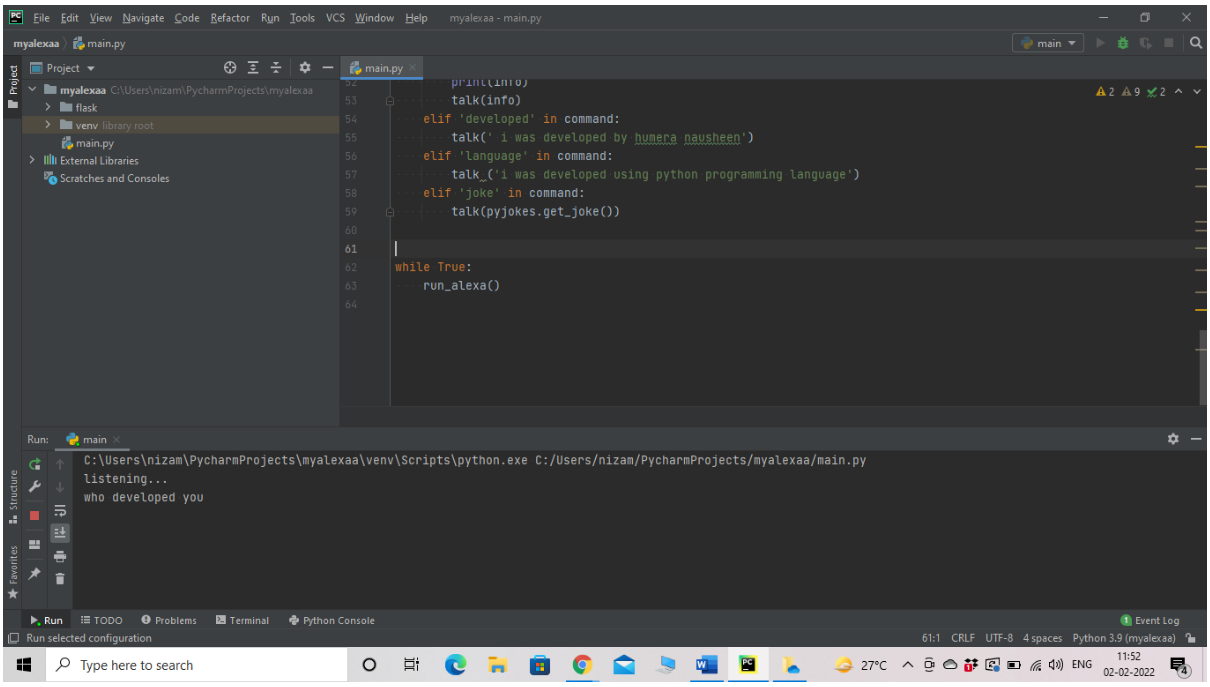
```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
tell me about maldives
Maldives (, US: ; Dhivehi: ދިވެހިރާއްޖެ, romanized: Dhivehi Raajje IPA: ɖiʋehi ɾaːjje), officially the Republic of Maldives, is an archipelagic country in the Indian
```



```
        talk('current time is' + time)
    elif 'date' in command:
        date = datetime.date.today()
        print(date)
        talk('current date is' + date)
    elif 'tell me' in command:
        person = command.replace('tell me', '')
        info = wikipedia.summary(person, 2)
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
        talk_('i was developed using python programming language')
    elif 'joke' in command:
        talk(pyjokes.get_joke())
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
tell me about maldives
Maldives (, US: ; Dhivehi: ދިވެހިރާއްޖެ, romanized: Dhivehi Raajje IPA: ɖiʋehi ɾaːjje), officially the Republic of Maldives, is an archipelagic country in the Indian
listening...
what is the time
11:50 AM
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
main
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
main.py
run_alexa() elif 'tell me' in command
Run: main
Run TODO Problems Terminal Python Console Event Log
PyCharm 2021.3.2 available // Update... (42 minutes ago)
50:42 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:50 02-02-2022



```
        print(info)
        talk(info)
    elif 'developed' in command:
        talk(' i was developed by humera nausheen')
    elif 'language' in command:
        talk_('i was developed using python programming language')
    elif 'joke' in command:
        talk(pyjokes.get_joke())


while True:
    run_alexa()
```

```
C:\Users\nizam\PycharmProjects\myalexaa\venv\Scripts\python.exe C:/Users/nizam/PycharmProjects/myalexaa/main.py
listening...
who developed you
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help myalexaa - main.py
myalexaa main.py
main
Project
myalexaa C:\Users\nizam\PycharmProjects\myalexaa
flask
venv library root
main.py
External Libraries
Scratches and Consoles
main.py
Run: main
Run TODO Problems Terminal Python Console Event Log
Run selected configuration
61:1 CRLF UTF-8 4 spaces Python 3.9 (myalexaa)
Type here to search
27°C ENG 11:52 02-02-2022

CONCLUSION

In this project “Virtual Assistant Using Python” we discussed the design and implementation. The project is built using open source software modules with PyCharm community backing which can accommodate any updates shortly. The modular nature of this project makes it more flexible and easier to add additional features without disturbing current system functionalities. The work started with analysing the audio commands given by the user through the microphone. This can be anything like getting any information.

FUTURE ENHANCEMENT

To make it not only work on human commands but also give responses to the user based on the query being asked or the words spoken by the user such as opening tasks and operations. It is greeting the user the way the user feels more comfortable and feels free to interact with the voice assistant. The application should also eliminate any kind of unnecessary manual work required in the user life of performing every task. The entire system works on the verbal input rather than the next one.

REFERENCES

- [1] R. Belvin, R. Burns, and C. Hein, “*Development of the HRL route navigation dialogue system*,” in Proceedings of ACL-HLT, 2001
- [2] V. Zue, S. Seneff, J. R. Glass, J. Polifroni, C. Pao, T.J.Hazen,and L.Hetherington, *“JUPITER: A Telephone Based Conversational Interface for Weather Information*,” IEEE Transactions on Speech and Audio Processing, vol. 8, no. 1, pp. 85–96, 2000.
- [3] M. Kolss, D. Bernreuther, M. Paulik, S. St¨ucker, S. Vogel, and A. Waibel, “*Open Domain Speech Recognition & Translation: Lectures and Speeches*,” in Proceedings of ICASSP, 2006.
- [4] D. R. S. Caon, T. Simonnet, P. Sendorek, J. Boudy, and G. Chollet, *“vAssist: The Virtual Interactive Assistant for Daily Homer-Care*,” in Proceedings of pHealth, 2011.
- [5] Crevier, D. (1993). *AI: The Tumultuous Search for Artificial Intelligence*. New York, NY: Basic Books, ISBN 0-465-02997-3.
- [6] Sadun, E., &Sande, S. (2014). *Talking to Siri: Mastering the Language of Apple’s Intelligent Assistant*.