**The Project Report on**

# JARVIS AI VOICE ASSISTANT

**Submitted To**

**Acharya Nagarjuna University**

**A project report submitted in the partial fulfillment of the requirements for the**

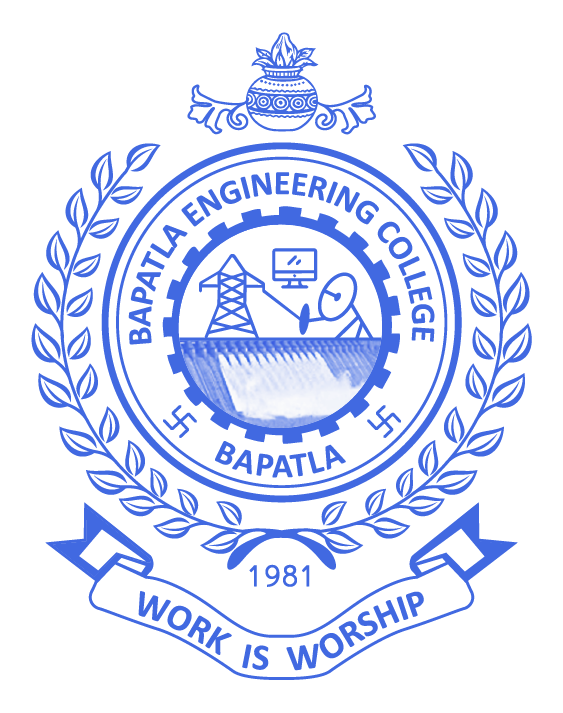
**Award of the Degree of**

**MASTER OF COMPUTER APPLICATIONS**

**Submitted By**

**ARUNURU NAVEEN**

**Reg.No: L20MC23014**

****

**Under The Esteemed Guidance Of**

**Sri N. KIRAN KUMAR M. Tech**

**Assistant Professor**

**DEPARTMENT OF MCA**

**BAPATLA ENGINEERING COLLEGE**

**(Approved by A.I.C.T.E)**

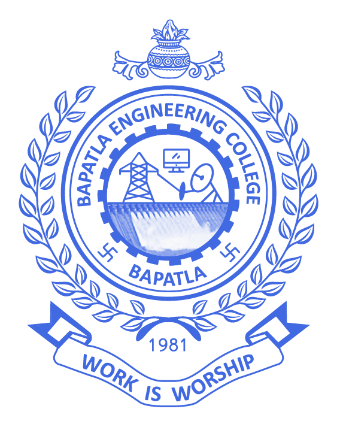
**(Affiliated to ACHARYA NAGARJUNA UNIVERSITY)**

**2019-2021**

**DEPARTMENT OF MCA**

**BAPATLA ENGINEERING COLLEGE**

**BAPATLA-522101**

****

**CERTIFICATE**

This is to certify that this project work entitled “**Secure Cloud Storage based on RLWE Problem”** is the bonafide work carried out by **ARUNURU NAVEEN**, **Reg.No: L20MC23014** submitted in Partial fulfillment of the requirement for the Award of Degree of “**Master of** **Computer Applications**”, during the academic year 2019-2021.

The results submitted in this project have been verified and are found to be satisfactory. The results embodied in this thesis have not been submitted to any other university for the award of the any other degree/diploma.

**Project Guide: Head of the Department:**

**Sri N. Kiran Kumar M. Tech Sri K.N. Prasad M. Tech**

**Assistant Professor Associate Professor & Head**

**Dept. of MCA Dept. of MCA**

**Bapatla Engineering College Bapatla Engineering College**

**Bapatla. Bapatla.**

**Signature of External Examiner**

# ACKNOWLEDGEMENT

I sincerely thank the following distinguished personalities who have given their advice and support for successful completion of this work.

I would like to thank **Dr. V. Damodara Naidu**, Principal, Bapatla Engineering College, Bapatla for providing the online resources and allowing me to carry out this work.

I extended my sincere thanks to **Sri K.N. Prasad**, Head of the Department for extending their cooperation and providing the required resources.

I am deeply indebted to most respect and my project guide **Sri N. Kiran Kumar**, Assistant Professor Dept. of MCA for his valuable and inspiring guidance, comments, suggestions and encouragements.

I extended my sincere thanks to all other teaching and non-teaching staff of the department who helped directly or indirectly for their cooperation and encouragement.

**ARUNURU NAVEEN**

**(L20MC23014)**

**DECLARATION**

This is to declare that the project **“Secure Cloud Storage based on RLWE Problem”** at Bapatla Engineering College has been presented by me during the academic year **2019-2021** in partial fulfillment of the requirements for the **“Master of Computer Application”**.

I also declare that this project is the result of my own efforts and that it has not been submitted to any other universities for the award of degree or diploma.

**ARUNURU NAVEEN**

**(L20MC23014)**

Table of Contents

[ACKNOWLEDGEMENT iii](#_Toc80180563)

[ABSTRACT viii](#_Toc80180564)

[CHAPTER-1 1](#_Toc80180565)

[INTRODUCTION 1](#_Toc80180566)

[1.1 INTRODUCTION 1](#_Toc80180567)

[CHAPTER- 2 5](#_Toc80180568)

[LITERATURE SURVEY 5](#_Toc80180569)

[2.1 LITERATURE REVIEW 5](#_Toc80180570)

[CHAPTER-3 7](#_Toc80180571)

[THEORETICAL BACKGROUND 7](#_Toc80180572)

[3.1 INTRODUCTION: 7](#_Toc80180573)

[3.2 INTRODUCTION TO PYTHON 12](#_Toc80180574)

[3.3 BENFITS OF PYTHON 21](#_Toc80180575)

[CHAPTER-4 30](#_Toc80180576)

[SYSTEM ANALYSIS 30](#_Toc80180577)

[4.1 EXISTING SYSTEM: 30](#_Toc80180578)

[4.1.1 DISADVANTAGES OF EXISTING SYSTEM: 30](#_Toc80180579)

[4.2 PROPOSED SYSTEM: 30](#_Toc80180580)

[4.2.1 ADVANTAGES OF PROPOSED SYSTEM: 31](#_Toc80180581)

[CHAPTER- 5 32](#_Toc80180582)

[SYSTEM DESIGN 32](#_Toc80180583)

[5.1 INTRODUCTION 32](#_Toc80180584)

[5.2 MODULES 32](#_Toc80180585)

[5.2.1 DATASET: 32](#_Toc80180586)

[5.2.2 PREPROCESSING: 32](#_Toc80180587)

[5.2.3 GRAPHS: 32](#_Toc80180588)

[5.2.4 PREDICTION: 32](#_Toc80180589)

[5.3 SYSTEM ARCHITECTURE 33](#_Toc80180590)

[5.4 UML DAIGRAMS 34](#_Toc80180591)

[5.4.1 CONSTRUCTION OF USE CASE DIAGRAMS: 37](#_Toc80180592)

[5.4.2 SEQUENCE DIAGRAMS: 40](#_Toc80180593)

[5.4.3 CLASS DIAGRAM: 42](#_Toc80180594)

[5.4.4 ACTIVITY DIAGRAM: 42](#_Toc80180595)

[CHAPTER-6 44](#_Toc80180596)

[SYSTEM REQUIREMENTS 44](#_Toc80180597)

[6.1 SYSTEM REQUIREMENTS 44](#_Toc80180598)

[6.1.1 HARDWARE REQUIREMENTS: 44](#_Toc80180599)

[6.1.2 SOFTWARE REQUIREMENTS: 44](#_Toc80180600)

[CHAPTER-7 45](#_Toc80180601)

[SYSTEM IMPLEMENTATION 45](#_Toc80180602)

[7.1 INPUT AND OUTPUT DESIGNS 45](#_Toc80180603)

[7.1.1 LOGICAL DESIGN 45](#_Toc80180604)

[7.1.2 PHYSICAL DESIGN 45](#_Toc80180605)

[7.2 INPUT & OUTPUT REPRESENTATION 46](#_Toc80180606)

[7.2.1 INPUT DESIGN 46](#_Toc80180607)

[7.2.2 OBJECTIVES 47](#_Toc80180608)

[7.2.3 OUTPUT DESIGN 47](#_Toc80180609)

[CHAPTER-8 69](#_Toc80180610)

[SYSTEM TESTING 69](#_Toc80180611)

[8.1 INTRODUCTION: 69](#_Toc80180612)

[8.2 LEVELS OF TESTING 69](#_Toc80180613)

[8.2.1 BLACK BOX TESTING 70](#_Toc80180614)

[8.2.2 WHITE BOX TESTING 72](#_Toc80180615)

[CHAPTER-9 74](#_Toc80180616)

[OUTPUT SCREENS 74](#_Toc80180617)

[CONCLUSION 75](#_Toc80180618)

[REFERENCES 76](#_Toc80180619)

**List of Figures**

**Name of the figure Pg.no**

[Figure 3. 1 Structure of cloud computing 7](#_Toc80179715)

[Figure 3. 2 Characteristics of cloud computing 9](#_Toc80179716)

[Figure 3. 3 Structure of service models 10](#_Toc80179717)

[Figure 5. 1 System Architecture 33](#_Toc80179735)

[Figure 5. 2 Use Case Diagram 39](#_Toc80179736)

[Figure 5. 3 Sequence diagram 41](#_Toc80179737)

[Figure 5. 4 Class Diagram 42](#_Toc80179738)

[Figure 5. 5 Activity Diagram 43](#_Toc80179739)

# ABSTRACT

The project aims to develop a personal-assistant for Window-based systems. Jarvis draws its inspiration from virtual assistants like Cortana for Windows, and Siri for iOS. It has been designed to provide a user-friendly interface for carrying out a variety of tasks by employing certain well-defined commands. Users can interact with the assistant either through voice commands or using keyboard input. As a personal assistant, Jarvis assists the end-user with day-to-day activities like general human conversation, searching queries in google, searching for videos, retrieving images, tells jokes, temperature, make a note, about corona cases , can open Facebook, Instagram, calculator, command prompt, date and time. The user statements/commands are analyzed with the help of machine learning to give an optimal solution.

In this Project Jarvis is Digital Life Assistant which uses mainly human communication means such Twitter, instant message and voice to create two way connections between human and his apartment, controlling lights and appliances, assist in cooking, notify him of breaking news, Facebook’s Notifications and many more. In our project we mainly use voice as communication means so the Jarvis is basically the Speech recognition application. The concept of speech technology really encompasses two technologies: Synthesizer and recognizer. A speech synthesizer takes as input and produces an audio stream as output. A speech recognizer on the other hand does opposite. It takes an audio stream as input and thus turns it into text transcription. The voice is a signal of infinite information. A direct analysisand synthesizing the complex voice signal is due to too much information contained in the signal

.

.

# CHAPTER-1

# INTRODUCTION

## Introduction

Speech is an effective and natural way for people to interact with applications, complementing or even replacing the use of mice, keyboards, controllers, and gestures. A hands-free, yet accurate way to communicate with applications, speech lets people be productive and stay informed in a variety of situations where other interfaces will not. Speech recognition is a topic that is very useful in many applications and environments in our daily life. Generally speech recognizer is a machine which understands humans and their spoken word in some way and can act thereafter. A different aspect of speech recognition is to facilitate for people with functional disability or other kinds of handicap. To make their daily chores easier, voice control could be helpful. With their voice they could operate the light switch turn off/on or operate some other domestic appliances. This leads to the discussion about intelligent homes where these operations can be made available for the common man as well as for handicapped With the information presented so far one question comes naturally: how is speech recognition done? To get knowledge of how speech recognition problems can be approached today, a review of some research highlights will be presented. The earliest attempts to devise systems for automatic speech recognition by machine were made in the 1950’s, when various researchers tried to exploit the fundamental ideas of acoustic-phonetics. In 1952, at Bell Laboratories, Davis, Biddulph, and Balashek built a system for isolated digit recognition for a single speaker [12]. The system relied heavily on measuring spectral resonances during the vowel region of each digit. In 1959 another attempt was made by Forgie , constructed at MIT Lincoln Laboratories. Ten vowels embedded in a /b/-vowel-/t/ format were recognized in a speaker independent manner [13]. In the 1970’s speech recognition research achieved a number of significant mile- stones. First the area of isolated word or discrete utterance recognition became a viable and usable technology based on the fundamental studies by Velichko and Zagoruyko in Russia , Sakoe and Chiba in Japan and Itakura in the United States . The Russian studies helped advance the use of pattern recognition ideas in speech recognition; the Japanese research showed how dynamic programming methods could be successfully applied; and Itakura’s research showed how the ideas of linear predicting coding (LPC). At AT&T Bell Labs, began a series of experiments aimed at making speech recognition systems that were truly speaker independent. They used a wide range of sophisticated clustering algorithms to determine the number of distinct patterns required to represent all Jarvis, Digital Life Assistant ShrutiKa Khobragade variations of different words across a wide user population. In the 1980’s a shift in technology from template-based approaches to statistical modeling methods, especially the hidden Markov model (HMM) approach [1]. The purpose of this paper is getting a deeper theoretical and practical understanding of a speech recognizer. The work started by examines a currently existing state of the art for feature extracting method MFCC. With this study from MFCC applying this knowledge in practical manner, the speech recognizer is implemented in .Net technology in C# language developed by Microsoft [11]. In our project we use The Speech Application Programming Interface or SAPI is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications. Applications that use SAPI include Microsoft Office,Microsoft Agent and Microsoft Speech Server.. In general all API have been designed such that a software developer can write an application to perform speech recognition and synthesis by using a standard set of interfaces, accessible from a variety of programming languages. In addition, it is possible for a 3rd-party company to produce their own Speech Recognition and Text-ToSpeech engines or adapt existing engines to work with SAPI. Basically Speech platform consist of an application runtimes that provides speech functionality, an Application Program Interface (API) for managing the runtime and Runtime Languages that enable speech recognition and speech synthesis (text-to-speech or TTS) in specific languages

**Problem Identification & Objectives**

A voice assistant or intelligent personal assistant is a software agent that can perform tasks or services for an individual based on verbal commands i.e. by interpreting human speech and respond via synthesized voices.

Users can ask their assistants’ questions, control home automation devices, and media playback via voice, and manage other basic tasks such as email, to-do lists, open or close any application etc with verbal commands.

Who doesn't want to have the luxury to own an assistant who always listens for your call, anticipates your every need, and takes action when necessary? That luxury is now available thanks to artificial intelligence-based voice assistants.

Voice assistants come in somewhat small packages and can perform a variety of actions after hearing your command. They can answer questions, play music, place online orders and do all kinds of AI-based stuff.

Voice assistants are not to be confused with virtual assistants, which are people who work remotely and can, therefore, handle all kinds of tasks. Rather, voice assistants are technology based. As voice assistants become more robust, their utility in both the personal and business realms will grow as well.

# CHAPTER- 2

# literature survey

## 2.1 literature review

This discipline of digital assistants having speech reputation has visible a few fundamental improvements or innovations. that is regularly mainly due to its call for gadgets like smartwatches or health bands, speakers, Bluetooth earphones, cell telephones, computers or desktops, television, etc. maximum the virtual gadgets that rectangular degree coming nowadays rectangular degree returning with voice assistants that facilitate to govern the tool with speech reputation solely. a modern set of strategies is being advanced continually to decorate the overall performance of voice computerized search [2] As the amount of statistics is growing exponentially presently famous as big statistics the only way to enhance the effects of digital assistants is to encompass our assistants with the device getting to know and educating our gadgets constant with their uses. Extraordinary fundamental strategies that rectangular degree similarly essential rectangular degree pc science, Internet of Things, big statistics get right of entry to and management, etc. Machine Learning [1] is truly hard and fast for pc science. This has been one of the most beneficial improvements in an era. Before AI we have a propensity to have been the ones UN business enterprise had been upgrading era to do an undertaking but presently the device is itself equipped to counter new responsibilities and remedy it even as now no longer was given to contain the people to adopt it. With the usage of voice assistants [2,3,8], we can adjust the undertaking truly, simply provide the entrance to the device inside the speech type and each one of the responsibilities is going to be completed via way of means of it converting your speech into textual content shape to getting rid of key phrases from that textual content and execute the question to provide effects to the user. This has been beneficial in everyday existence style. From cell telephones to personal computers to mechanical industries those assistant’s rectangular degrees in appreciably call for automating responsibilities and growing efficiency [3].

# Chapter-3

# Theoretical background

## 3.1 Introduction:

## 3.2 Introduction to PYTHON

**Python Technology**

Python technology is both a programming language and a platform.

**The python Programming Language**

THE PYTHON PROGRAMMING LANGUAGE IS A HIGH-LEVEL LANGUAGE THAT CAN BE CHARACTERIZED BY ALL OF THE FOLLOWING BUZZWORDS:

* + - Simple
    - Architecture neutral
    - Object oriented
    - Portable
    - Distributed
    - High performance
    - Interpreted
    - Multithreaded
    - Robust
    - Dynamic
    - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Python programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Python byte codes —the platform-independent codes interpreted by the interpreter on the Python platform. The interpreter parses and runs each Python byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

FEATURES OF MACHINE LEARNING

• It is nothing but automating the Automation.

• Getting computers to program themselves.

• Writing Software is bottleneck.

• Machine leaning models involves machines learning from data without the help of humans or any kind of human intervention.

• Machine Learning is the science of making of making the computers learn and act like humans by feeding data and information without being explicitly programmed.

• Machine Learning is totally different from traditionally programming, here data and output is given to the computer and in return it gives us the program which provides solution to the various problems. Below is the figure.

**Traditional Programming vs Machine Learning**

• Machine Learning is a combination of Algorithms, Datasets, and Programs.

• There are Many Algorithms in Machine Learning through which we will provide us the exact solution in predicting the disease of the patients.

• How Does Machine Learning Works?

• Solution to the above question is Machine learning works by taking in data, finding relationships within that data and then giving the output.

**Machine Learning Model**

• There are various applications in which machine learning is implemented such as Web search, computing biology, finance, e-commerce, space exploration, robotics, social networks, debugging and much more.

• There are 3 types of machine learning supervised, unsupervised, and reinforcement.

**BENEFITS OF PYTHON**

• Presence of Third-Party Modules

• Extensive Support Libraries

• Open Source and Community Development

• Learning Ease and Support Available

• User-friendly Data Structures

• Productivity and Speed

• Highly Extensible and Easily Readable Language.

**Python**

Python is high level language and it is also integrated version of the program. Python is an object-oriented approach and its main aim to help programmers to write the code clearly, logical code for small and large scale of project.

Pytrhon is dynamically typed and garbage collected it also support multiple programming and it is both procedure and object oriented and also functional programming. And structural programming also supported. It has many built in function it also supports filter, map and reduce function. All the machine learning algorithm and the libraries are being supported by the python programming language. Python also support list, dict, sets and other generators. Python code can be run in different platform such as anaconda, PyCharm etc.

The main goal of this programing language is as follows:

• Python is simple, object-oriented programming language.

• The language and implementation should provide support for software engineering principles such as strong type library preset for different machine learning algorithm, and all other algorithm in simple manner.

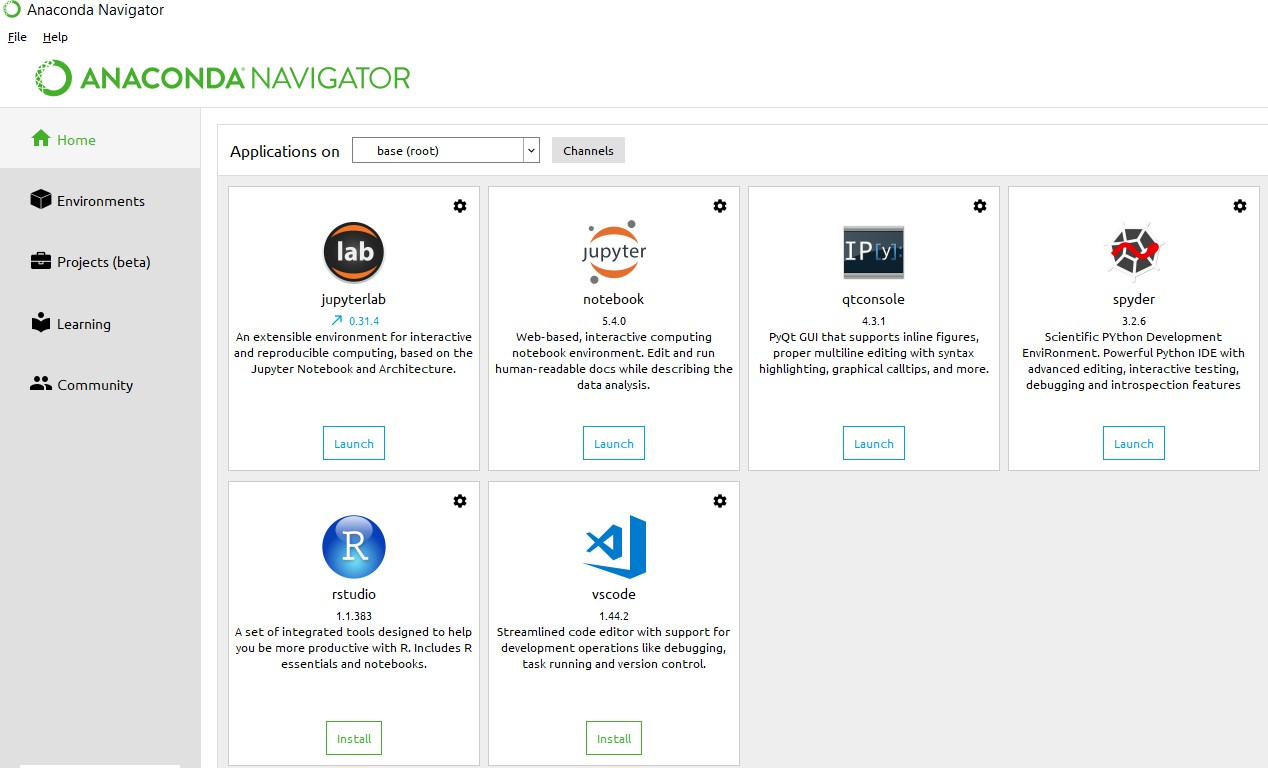
• Coding will be smooth in python and the data analysis can be easily done in python.

This is so much so to the point where we now have modules and APIs at our disposal, and you can engage in machine learning very easily without almost any knowledge at all of how it works. With the defaults from Scikit-learn, you can get 90-95% accuracy on many tasks right out of the gate. Machine learning is a lot like a car, you do not need to know much about how it works in order to get an incredible amount of utility from it.

Despite the apparent age and maturity of machine learning, I would say there's no better time than now to learn it, since you can actually use it. Machines are quite powerful, the one you are working on can probably do most of this series quickly. Data is also very plentiful lately.

**Anaconda**

Anaconda is free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine Learning applications, Large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. It is developed and maintained by Anaconda, Inc. The distribution incudes data-science packages suitable for Windows, Linux, and macOS. Packaged versions are required and are managed by the package management system anaconda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depends on, and a small number of other packages.



**Anaconda Console**

**Jupyter notebook**

Jupiter Notebook or so called IPython Notebook is an interactive web based computational mean for starting with Jupiter Notebook documents. The term notebook itself is a huge entity to represent the integration with different entity sets. JSON is the main document form from the same for the execution which follows the brief on the schema and the input and output means. It has high integration with several language set and has various flexibilities with the choices. The extension used for the same is “.ipynb” which runs in this platform. It’s an open-source software package with interactive communication means. It has it’s open standards for the same. It’s an open community best for budding programmers . The flexibility of the same is phenomenon and splendidly done the configuration and integration of the same is simplest and easy on hold so that no prior distortion is generated and the efficiency of the same is measured through out any system of choice.

It’s the best software sets that been used across cross for designing and developing of the products and support wide help support. Not only to that, it provides scalability in the code and the deployment of the same. Various Language can be changed and the project can be undertaken on the same. The created notebook files can be shared and stored in various means for further utilization. It supports cultivated and interactive output sets. Easily crossed over for graphing, plotting and visualizing of the elements. Data Integration of the same is to it’s best. The integration of big data and it can process chunks of values in an approx. time which gives a better performance and the higher computational means. Various works on data like cleaning, cleansing, transforming modeling and visualizing can be done by the same

Machine learning is the ability that gives the computer to learn without being explicitly programmed. There are two types of machine learning:

Supervised Learning: supervised learning is the learning of the labelled data. It is the types of machine learning that maps the input and output based on the examples input-output pairs. In supervised learning each training data having pairs of input and desired outputs values. Supervised learning algorithm analyzes the training data and produces a function which can be used for mapping of new data.

Fig 2.1 Supervised Learning The output to solve the supervised learning algorithm are as:

• Determine the types of data, before doing anything else the user should understand which types of data set is to be used for training the data.

• Gathered the training data sets either in form of human experts or from measurements.

• Determine the feature of inputs from the learned data and depends on the inputs it changed into feature vector; number of features should not be large but should contains enough information to accurately predict the outputs.

• Check the learned function and the learned algorithm for example we use support vector machines or decisions tree.

• Complete the design and run the trained data sets.

• Analyzed the output and verify the data sets to get the accurate outputs.

Unsupervised Learning:

Unsupervised learning is a type of machine learning that helps in finding the previously unknown patterns in the data set without any known labels. It is known as self- organization and allows modelling probability densities of given inputs.

Fig 2.2 unsupervised Learning Some of the algorithm used in unsupervised learning are:

• Clustering

• Anomaly detection

• Neural networks

• Approach for learning latent variable models

• Non labelled data

Semi Supervised Machine Learning algorithm: It’s like the middle man which have some labeled data and some unlabeled which can be prosed by the both the structured and unsupervised learning.

The algorithms have been compared based upon the parameters: Size of the dataset and Number of technical indicators used. Accuracy and F-measure values have been computed for each algorithm. Long term model has been used to compute the accuracy and F-measure.

Reinforcement Learning: This type of learning is used to reinforce or strengthen the network based on critic information. That is, a network being trained under reinforcement learning, receives some feedback from the environment. However, the feedback is evaluative and not instructive as in the case of supervised learning. Based on this feedback, the network performs the adjustments of the weights to obtain better critic information in future.

This learning process is similar to supervised learning but we might have very less information. The following figure gives the block diagram of reinforcement learning:

**import numpy as np**

* NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
* At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:
  + NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
  + The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
  + NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
  + A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

**import time**

This module provides various time-related functions. For related functionality, see also the datetime and calendar modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

An explanation of some terminology and conventions is in order.

The epoch is the point where the time starts, and is platform dependent. For Unix, the epoch is January 1, 1970, 00:00:00 (UTC). To find out what the epoch is on a given platform, look at time.gmtime(0).

The term seconds since the epoch refers to the total number of elapsed seconds since the epoch, typically excluding leap seconds. Leap seconds are excluded from this total on all POSIX-compliant platforms.

The functions in this module may not handle dates and times before the epoch or far in the future. The cut-off point in the future is determined by the C library; for 32-bit systems, it is typically in 2038.

Function strptime() can parse 2-digit years when given %y format code. When 2-digit years are parsed, they are converted according to the POSIX and ISO C standards: values 69–99 are mapped to 1969–1999, and values 0–68 are mapped to 2000–2068.

UTC is Coordinated Universal Time (formerly known as Greenwich Mean Time, or GMT). The acronym UTC is not a mistake but a compromise between English and French.

DST is Daylight Saving Time, an adjustment of the timezone by (usually) one hour during part of the year. DST rules are magic (determined by local law) and can change from year to year. The C library has a table containing the local rules (often it is read from a system file for flexibility) and is the only source of True Wisdom in this respect.

The precision of the various real-time functions may be less than suggested by the units in which their value or argument is expressed. E.g. on most Unix systems, the clock “ticks” only 50 or 100 times a second.

On the other hand, the precision of time() and sleep() is better than their Unix equivalents: times are expressed as floating point numbers, time() returns the most accurate time available (using Unix gettimeofday() where available), and sleep() will accept a time with a nonzero fraction (Unix select() is used to implement this, where available).

The time value as returned by gmtime(), localtime(), and strptime(), and accepted by asctime(), mktime() and strftime(), is a sequence of 9 integers. The return values of gmtime(), localtime(), and strptime() also offer attribute names for individual fields.

See struct\_time for a description of these objects.

Changed in version 3.3: The struct\_time type was extended to provide the tm\_gmtoff and tm\_zone attributes when platform supports corresponding struct tm members.

Changed in version 3.6: The struct\_time attributes tm\_gmtoff and tm\_zone are now available on all platforms.

**import os**

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions:

The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function os.stat(path) returns stat information about path in the same format (which happens to have originated with the POSIX interface).

Extensions peculiar to a particular operating system are also available through the os module, but using them is of course a threat to portability.

All functions accepting path or file names accept both bytes and string objects, and result in an object of the same type, if a path or file name is returned.

On VxWorks, os.popen, os.fork, os.execv and os.spawn\*p\* are not supported.

## Pyttsx3:

1. Pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both python 2 and pyttsx3 is a cross-platform text to speech library which is platform-independent. The major advantage of using this library for text-to-speech conversion is that it works offline. To install this module, type the below command in the terminal:

$ pip install pyttsx3

## 2.Speech Recognition :

## Speech Recognition Is An Important Feature In Several Applications Used Such As Home Automation, Artificial Intelligence, Etc. This Article Aims To Provide An Introduction On How To Make Use Of The Speech recognition Library Of Python.

## Speech recognition Allows Us To Convert Audio Into Text For Further Processing. To Install This Module, Type The Below Command In The Terminal:

$ pip install SpeechRecognition

## 3.DateTime:

The datetime module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

$ pip install Datetime

**4.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.

$ pip install Wikipedia

**5.Web browser:**  
The webbrowser module provides a high-level interface to allow displaying Web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.  
  
**6.Os :**

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

# Chapter-4

# System analysis

## 4.1 EXISTING SYSTEM:

Existing methods user manual email processing or web-based applications with manual text processing techniques using NLTK and natural language processing.

### 4.1.1 DISADVANTAGES OF EXISTING SYSTEM:

* + - Send Gmail messages
    - Dynamic News reporting at any time
    - Open any website with just a voice command
    - Plays music
    - Tells time
    - Wikipedia powered AI
    - Dictionary with intelligent Sensing i.e auto checking if spell mistake
    - Weather report such as temping
    - Speed, Humidity, Weather description
* Latitude and longitude

## 4.2 PROPOSED SYSTEM:

* A voice assistant is a digital assistant that uses voice recognition, natural language processing and speech synthesis to provide aid to users through desktop and voice recognition. Voice assistants are built on artificial intelligence (AI), machine learning and voice recognition technology. As the end user interacts with the digital assistant, the AI programming uses sophisticated algorithms to learn from data input and better itself at predicting the user's needs. Some assistants are built with more advanced cognitive computing technologies which will allow a digital assistant to understand and carry out multi-step requests with numerous interactions and perform more tasks Digital assistants can be contrasted with another application of consumer-facing AI called smart advisors. Smart advisor programs are knowledge-oriented, while digital assistants are taskoriented, although some perform both roles. Popular voice assistants currently include Apple's Siri, Amazon's Alexa, Google Now, Google Assistant and Microsoft's Cortana.
* This Software aims at developing a personal assistant for Windows-based systems. The main purpose of the software is to perform the tasks of the user at certain commands, provided in either of the ways, speech or text. It will ease most of the work of the user as a complete task can be done on a single command. Jarvis draws its inspiration from Virtual assistants .Users can interact with the assistant either through voice commands or keyboard input. Artificial intelligence (AI) is the ability of a computer program or a machine to think and learn. They work on their own without being encoded with commands. These software components cover machine learning, image recognition, natural language processing, data mininig and more.

### 4.2.1 ADVANTAGES OF PROPOSED SYSTEM:

* + Voice commands are used to processing commands
  + Only voice-based system is used manual work is reduced.
  + User can access information on travel also.

.

# CHAPTER- 5

# SYSTEM design

## 5.1 introduction

System Design Introduction:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

## 5.2 modules

### 5.2.1 Methodology:

At graduation, we have got a bent to shape our application able to the utilization of gadget voice with the assistance of sapi5 and pyttsx3. Pyttsx3 can also be a text-to-speech conversion library in Python. Now not like definitely distinctive libraries, it works offline and is similar temperament with each Python 2 and three. The Speech Application Programming Interface or SAPI is a diploma API evolved through Microsoft to permit the usage of speech popularity and speech synthesis inside Windows applications. Then we have got a bent to stipulate the talk function to differ this technique to talk the outputs. at that time, we're visiting outline a characteristic to want voice instructions the utilization of the gadget microphone. The foremost function is then made public during which all of the competencies of this technique rectangular degree are made public. The planned system is meant to possess the following functionality: (a) The Jarvis asked the user for input and keeps listening for orders. The time for a hearing is visiting be set up in step with the user's control (b) If the assistant fails to grasp the command it's visiting keep asking the user to repeat the command once again} and yet again. (c) This assistant is visiting be bespoken to possess either male or female voice in step with user’s demand. (d) the current version of the assistant supports choices like Checking weather updates, deed and checking emails, Searching Wikipedia, Stream music, Open applications, Text messages, checking dates and times, taking notes, show notes, Open YouTube, etc..

### 5.2.2 INputs:

Streaming Music: The user can command Jarvis to play a music track and it's going to execute a command and search into it from the song Folder. Read the latest news from headlines: Jarvis will examine out latest headlines from the knowledge retailers of the required topics you care about or need information Keep Tabs on the traffic & the weather: Jarvis can research the weather forecast or alert you if there is an accident that will delay your morning journey. Set Reminders/timers: You’ll be able to tell Jarvis to wake you up daily morning at 4 a.m. Answer the following questions: Jarvis can look up simple information, solve mathematical problems, or tell you a joke However, Alexa, Sleek, and homey aren't the sole ones altogether of most home automation systems that reply to user voice directions. Castles has been around since late 2012, however, its huge hub and Kinetic voice management programmer will solely be used on a Windows machine.

### 5.2.3 Modules Imported:

WIKIPEDIA

Wikipedia may be a library in python which it possible for the virtual assistant to process the queries regarding Wikipedia and display the results to users. This can be an internet library and wishes an internet connection to fetch the results. The no. of lines that the user wants to urge as a result are often set manually.

SPEECH RECOGNITION The speech popularity module that used this method is Google’s Speech Recognition API that's imported in python the usage of the command ‚import speech popularity as sr‛. This module is employed to apprehend the voice that's given as entered with the help of using the user. This is often a loose API this can be furnished and supported with the help of using Google. This is a mild API that facilitates lowering the size of our application. TTS & STT The voice that's given as entering is first transformed to textual format the usage of the speech reputation module. The textual content is then processed to present the top results of the question given via way of means of the person. The final step is the conversion of the tip results of the processed question to speech which is the end output. most time eating some of them is STT because of the fact the gadget first needs to concentrate on the person and exclusive customers have exclusive, some are clean to apprehend at the same time as some aren't simply audible. This is often the step upon which our general execution time depends. Once the speech is transformed into textual content executing instructions and giving the outcomes again to the person isn't a time-eating step.

PYTTSX3 The pyttsx3 offline library is used in Python for TTS conversion and is supported by both Python 2 and Python 3.The run and wait functionality is additionally during this module only. It determines what quantity of time the system will sit up for another input or in other words the interval between inputs. This can be a free module available within the python community which may be installed using the pip command rather like other modules.

DATETIME The Date Time module is imported to assist the potential of the date and time. For instance, the buyer desires to understand the cutting date and time or the patron desires to agenda a mission at a positive time. Briefly, this module helps lessons to control date and time and perform operations in line with it only. This is a critical module, especially in duties during which we'd like to carry the music of your time. This module might be very small long and allows us to manipulate the size of our program. If the modules are too huge or heavy then the gadget will lag and deliver sluggish responses. WEBBROWSER This module enables the computer to show consumers internet-based full records. For example, if the customer wants to access any website, he might write "Open Google." The input is processed through the use of the internet browser module, and the consumer receives a browser with Google open in it. The browser that will soon be used is the default internet browser. DESIGN The overall design of our system consists of the following phases: (a)Taking input from the user in the form of voice (b) Converting the speech into text to be processed by the assistant. (c) The converted text is now processed to get the required results.

**Algorithms**

The speech production begins in the humans mind, when he or she forms a thought that is to be produced and transferred to the listener. After having formed the desired thought, he or she constructs a phrase/sentence by choosing a collection of finite mutually exclusive sounds. The basic theoretical unit for describing how to bring linguistic meaning to the formed speech, in the mind, is called phonemes. Phonemes can be seen as a way of how to represent the different parts in a speech waveform, produced via the human vocal mechanism and divided into continuant (stationary) or non-continuant parts. A phoneme is continuant if the speech sound is produced when the vocal tract is in a steady-state. In opposite of this state, the phoneme is non-continuant when the vocal tract changes its characteristics during the production of speech. For example if the area in the vocal tract changes by opening and closing the mouth or moving your tongue in different states, the phoneme describing the speech produced is non-continuant Phonemes can be grouped based on the properties of either the time waveform or frequency characteristics and classified in different sounds produced by the human vocal tract

## 5.3 system architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system. Organized in a way that supports reasoning about the structures and behaviors of the system.

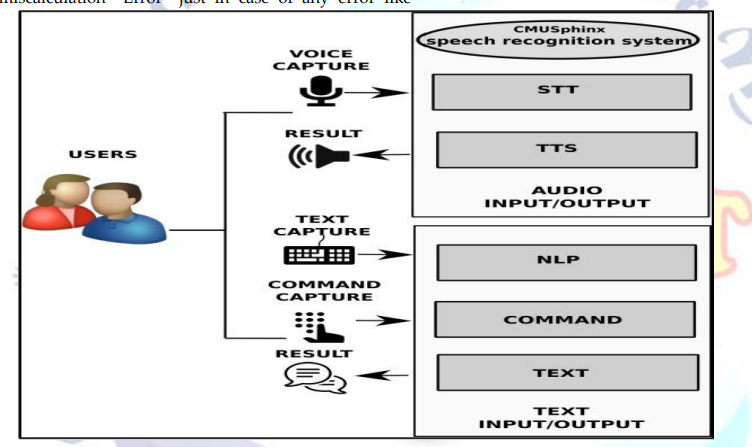
****

Figure 5. 1 System Architecture

3-Tier Architecture:

The three-tier software architecture (a three-layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems.

**Advantages of Three-Tier:**

* Separates functionality from presentation.
* Clear separation – better understanding.
* Changes limited to well define components.
* Can be running on WWW.
* Effective network performance.

## 5.4 UML DAIGRAMS

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



<<Actor name>>

An actor is someone or something that:

Interacts with or uses the system.

* Provides input to and receives information from the system.
* Is external to the system and has no control over the use cases.

Actors are discovered by examining:

* Who directly uses the system?
* Who is responsible for maintaining the system?
* External hardware used by the system.
* Other systems that need to interact with the system.

Questions to identify actors:

* + Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
  + Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
  + Which external hardware or systems (if any) use the system to perform tasks?
  + What problems does this application solve (that is, for whom)?
  + And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

1. System Administrator
2. Customer
3. Customer Care

Identification of use cases:

Use case: A use case can be described as a specific way of using the system from a user’s (actor’s) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

* Pattern of behavior the system exhibits
* A sequence of related transactions performed by an actor and the system
* Delivering something of value to the actor

Use cases provide a means to:

* capture system requirements
* communicate with the end users and domain experts
* test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

* For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
* Name the use cases.
* Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

* What are the tasks of each actor?
* Will any actor create, store, change, remove or read information in the system?
* What use case will store, change, remove or read this information?
* Will any actor need to inform the system about sudden external changes?
* Does any actor need to inform about certain occurrences in the system?
* What usecases will support and maintains the system?

**1.2 Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

* When and how the use case starts and ends
* Use case/actor interactions
* Data needed by the use case
* Normal sequence of events for the use case
* Alternate or exceptional flows

### 5.4.1 Construction of Use case diagrams:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

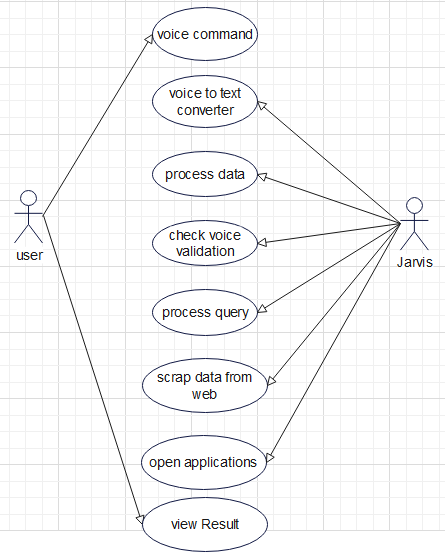


Figure 5. 2 Use Case Diagram

### 5.4.2 SEQUENCE DIAGRAMS:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

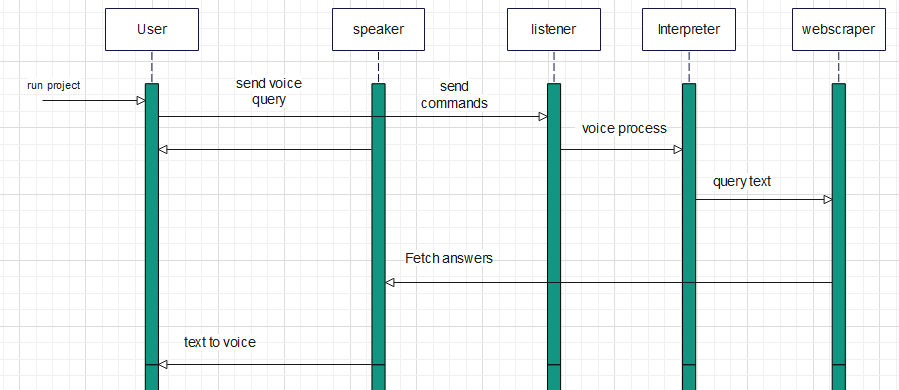
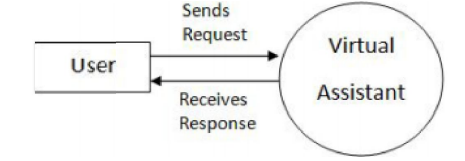
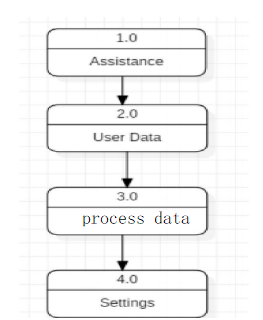


Figure 5. 3 Sequence diagram

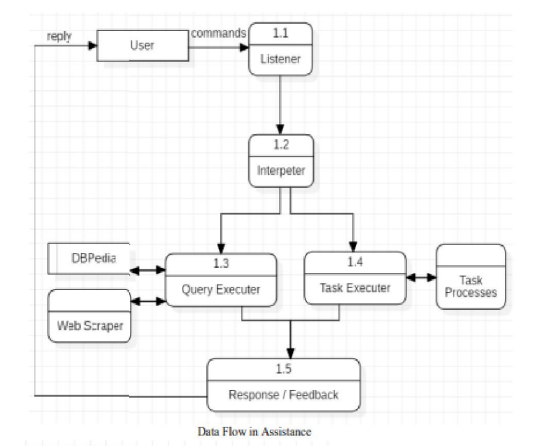
DFD level0:



DFD level 1



DFD Level 2:



### 5.4.3 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

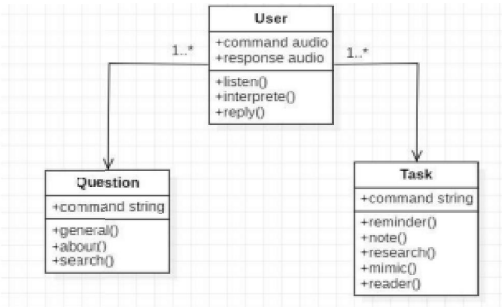


Figure 5. 4 Class Diagram

### 5.4.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

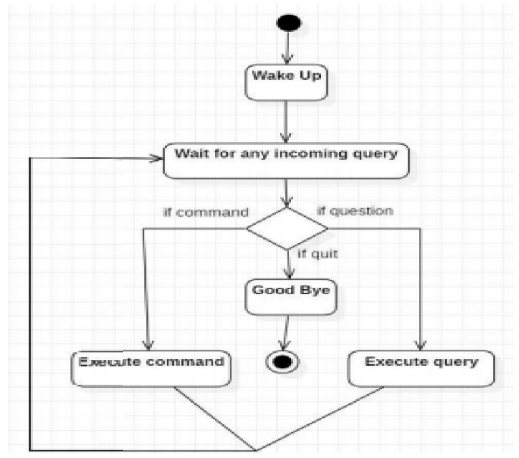


Figure 5. 5 Activity Diagram

# 

# CHAPTER-6

# system requirements

## 6.1 SYSTEM REQUIREMENTS

### 6.1.1 HARDWARE REQUIREMENTS:

* System : Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz
* Hard Disk : 1 TB.
* Input Devices : Keyboard, Mouse
* Ram : 4 GB.

### 6.1.2 SOFTWARE REQUIREMENTS:

* Operating system : Windows XP/7/10.
* Coding Language : Python
* Tool : Anaconda
* Interface : OPENCV

# Chapter-7

# System implementation

To conduct studies and analyses of an operational and technological nature, and To promote the exchange and development of methods and tools for operational analysis as applied to defense problems.

## 7.1 input and output designs

### 7.1.1 Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems design are included. Logical design includes ER Diagrams i.e. Entity Relationship Diagrams

### 7.1.2 Physical design

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified / authenticated, how it is processed, and how it is displayed as output. In Physical design, following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

## 7.2 INPUT & OUTPUT REPRESENTATION

### 7.2.1 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

### 7.2.2 Objectives

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

### Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

* 1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
  2. Select methods for presenting information.
  3. Create document, report, or other formats that contain information produced by the system.

**Code**

from PyQt5 import QtWidgets, QtGui,QtCore

from PyQt5.QtGui import QMovie

import sys

from PyQt5.QtWidgets import \*

from PyQt5.QtCore import \*

from PyQt5.QtGui import \*

from PyQt5.uic import loadUiType

import pyttsx3

import speech\_recognition as sr

import os

import time

import webbrowser

import datetime

import re

import wikipedia

flags = QtCore.Qt.WindowFlags(QtCore.Qt.FramelessWindowHint)

engine = pyttsx3.init('sapi5')

voices = engine.getProperty('voices')

engine.setProperty('voice',voices[0].id)

engine.setProperty('rate',180)

def **speak**(audio):

    engine.say(audio)

    engine.runAndWait()

def **wish**():

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

    if hour>=0 and hour <12:

        speak("Good morning")

    elif hour>=12 and hour<18:

        speak("Good Afternoon")

    else:

        speak("Good night")

def **takeCommand**():

    r = sr.Recognizer()

    with sr.Microphone() as source:

**print**("Jarvis At Your Service!")

        r.pause\_threshold = 2

        command = r.listen(source)

    try:

**print**("Recognizing...")

        recognized = r.recognize\_google(command, language='en-in')

**print**(recognized)

    except Exception as e:

**print**(e)

        statement="Pardon mam.., I Couldn't Recognize Your Voice, If its nothing to command, i'll take a leave"

**print**(statement)

        speak(statement)

        return None

    return recognized

class **mainT**(**QThread**):

    def **\_\_init\_\_**(self):

        super(mainT,self).**\_\_init\_\_**()

    def **run**(self):

        self.JARVIS()

    def **STT**(self):

        R = sr.Recognizer()

        with sr.Microphone() as source:

**print**("Listning...........")

            audio = R.listen(source)

        try:

**print**("Recog......")

            text = R.recognize\_google(audio,language='en-in')

**print**(">> ",text)

        except Exception:

            speak("Sorry Speak Again")

            return "None"

        text = text.lower()

        return text

    def **JARVIS**(self):

        wish()

        while True:

            self.query = self.STT()

            if 'good bye' in self.query:

                sys.exit()

            elif 'open google' in self.query:

                webbrowser.open('www.google.co.in')

                speak("opening google")

            elif 'open youtube' in self.query:

                webbrowser.open("www.youtube.com")

            elif 'play music' in self.query:

                speak("playing music from pc")

                self.music\_dir ="./music"

                self.musics = os.listdir(self.music\_dir)

                os.startfile(os.path.join(self.music\_dir,self.musics[0]))

            elif 'notepad' in self.query:

                os.startfile("C:\\Windows\\notepad.exe")

            elif "open chrome" in self.query:

                os.startfile("C:\\Windows\\notepad.exe")

            elif 'wikipedia' in self.query:  *#if wikipedia found in the query then this block will be executed*

                speak('Searching Wikipedia...')

**print**('Searching Wikipedia...')

                self.query = self.query.replace("wikipedia", "sachin tendulkar")

                results = wikipedia.summary(self.query, sentences=2)

                speak("According to Wikipedia")

**print**("According to Wikipedia")

**print**(results)

                speak(results)

            elif re.search("jokes|joke|Jokes|Joke", self.query):

                joke\_ = obj.tell\_me\_joke('en', 'neutral')

**print**(joke\_)

                speak(joke\_)

            elif re.search('weather|temperature', self.query):

*# city = res.split(' ')[-1]*

*# weather\_res = obj.weather(city=city)*

                weather\_res = obj.get\_weather(self.query)

**print**(weather\_res)

                speak(weather\_res)

            elif re.search('news', self.query):

                news\_res = obj.news()

                pprint.pprint(news\_res)

                speak(f"I have found {**len**(news\_res)} news. You can read it. Let me tell you first 2 of them")

                speak(news\_res[0])

                speak(news\_res[1])

            elif re.search('date', self.query):

                date = obj.tell\_me\_date()

**print**(date)

**print**(speak(date))

            elif re.search('time', self.query):

                time = obj.tell\_me\_time()

**print**(time)

                speak(time)

            elif "what is your name" in self.query:

                speak("iam jarvis ai voice assistant")

**print**("iam jarvis ai voice assistant")

            elif 'how are you' in self.query:

                speak("I am fine, Thank you")

**print**("I am fine, Thank you")

                speak("How are you, mam")

**print**("How are you, mam")

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

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

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

            elif "who are you" in self.query:

                speak("I am your jarvis voice assistant ")

**print**("I am your jarvis voice assistant ")

            elif "open facebook" in self.query:

                webbrowser.open\_new\_tab("facebook.com")

            elif "make a note" in self.query:

                speak("What Should i write down sir?")

**print**("What Should i write down sir?")

                note = takeCommand()

                remember = **open**('pytext.txt', 'w')

                remember.write(note)

                remember.close()

                speak("content added successfully in pytext.txt" + note)

**print**("content added successfully in pytext.txt" + note)

            elif 'exit' in self.query:

                speak('mam . Call Me Anytime, at your service')

**print**('mam . Call Me Anytime, at your service')

**quit**()

def **greet**():

    t\_hour = datetime.datetime.now().hour

    if 24> t\_hour <4:

        speak("Pleasant Night mam!, Jarvis at Your Command")

    elif 12> t\_hour >4:

        speak("Good Morning mam, Jarvis at Your Command")

    elif 18> t\_hour >12:

        speak("Good Afternoon mam!, Jarvis at Your Command")

    else:

        speak("Good Evening mam!, Jarvis at Your Command")

greet()

FROM\_MAIN,\_ = loadUiType(os.path.join(os.path.dirname(\_\_file\_\_),"./scifi.ui"))

class **Main**(**QMainWindow**,FROM\_MAIN):

    def **\_\_init\_\_**(self,parent=None):

        super(Main,self).**\_\_init\_\_**(parent)

        self.setupUi(self)

        self.setFixedSize(1920,1080)

        self.label\_7 = QLabel

        self.exitB.setStyleSheet("background-image:url(./lib/exit - Copy.png);\n"

        "border:none;")

        self.exitB.clicked.connect(self.close)

        self.setWindowFlags(flags)

        Dspeak = mainT()

        self.label\_7 = QMovie("./lib/gifloader.gif", QByteArray(), self)

        self.label\_7.setCacheMode(QMovie.CacheAll)

        self.label\_4.setMovie(self.label\_7)

        self.label\_7.start()

        self.ts = time.strftime("%A, %d %B")

        Dspeak.start()

        self.label.setPixmap(QPixmap("./lib/tuse.png"))

        self.label\_5.setText("<font size=8 color='white'>"+self.ts+"</font>")

        self.label\_5.setFont(QFont(QFont('Acens',8)))

app = QtWidgets.QApplication(sys.argv)

main = Main()

main.show()

**exit**(app.exec\_())

# Chapter-8

# System testing

## 8.1 INTRODUCTION:

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

## 8.2 Levels of Testing

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

### 8.2.1 BLACK BOX TESTING

What is Black Box Testing?

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

* Functional testing – This black box testing type is related to functional requirements of a system; it is done by software testers.
* Non-functional testing – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* Regression testing – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

### 8.2.2 WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

WHAT DO YOU VERIFY IN WHITE BOX TESTING?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

HOW DO YOU PERFORM WHITE BOX TESTING?

  To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This  method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

Unit testing:

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­1 |
| Name of Test: ­ | Input voice |
| Items being tested: ­ | Input voice from system |
| Sample Input: ­ | Voice data |
| Expected output: ­ | Voice converted to text |
| Actual output: ­ | Voice recognized |
| **Remarks: ­** | **Pass.** |

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­2 |
| Name of Test: ­ | Speech to text |
| Items being tested: ­ | Input voice converted to query |
| Sample Input: ­ | query |
| Expected output: ­ | Query given to system function |
| Actual output: ­ | Query is processed. |
| Remarks: ­ | pass |

**Integration Testing:**

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. It occurs after unit testing and before validation testing. Integration testing can be done in two ways: Bottom­up integration testing and Top­down integration testing.

* + 1. **Bottom­up Integration**

This testing begins with unit testing, followed by tests of progressively higher­level combinations of units called modules or builds.

* + 1. **Top­down Integration**

In this testing, the highest­level modules are tested first and progressively, lower­level modules are tested thereafter.

In a comprehensive software development environment, bottom­up testing is usually done first, followed by top­down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations. Table 6.5 shows the test cases for integration testing and their results

|  |  |
| --- | --- |
| Sl # Test Case : ­ | ITC­1 |
| Name of Test: ­ | Os functions |
| Item being tested: ­ | Query given input to os function |
| Sample Input: ­ | Speech to text converted voice to text |
| Expected output: ­ | Notepad and other functions are performed |
| Actual output: ­ | Result displayed |
| Remarks: ­ | Pass. |

**System testing**:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black­box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is important because of the following reasons:

System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.

The application is tested thoroughly to verify that it meets the functional and technical specifications.

The application is tested in an environment that is very close to the production environment where the application will be deployed.

System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

System Testing is shown in below tables

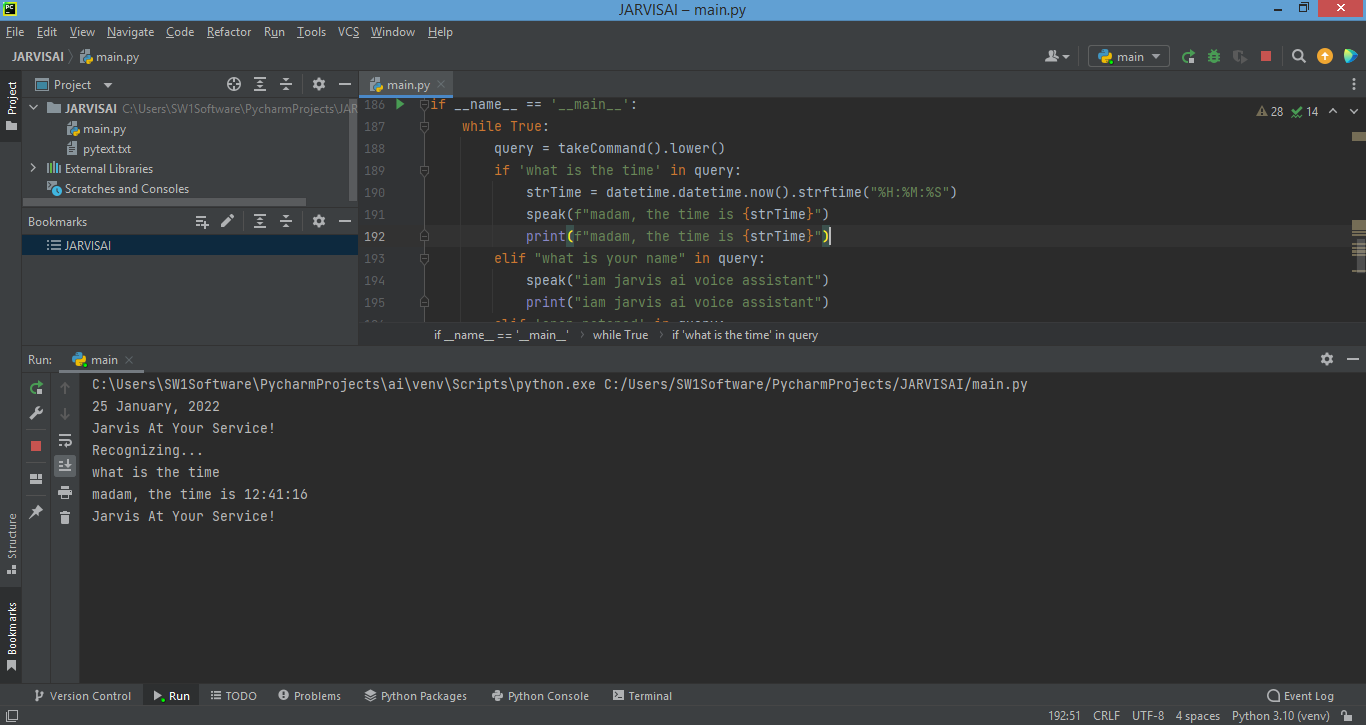
|  |  |
| --- | --- |
| Sl # Test Case : ­ | STC­1 |
| Name of Test: ­ | System testing in various versions of OS |
| Item being tested: ­ | OS compatibility. |
| Sample Input: ­ | Execute the program in windows XP/ Windows­7/8 |
| Expected output: ­ | Performance is better in windows­7 |
| Actual output: ­ | Same as expected output, performance is better in windows­7 |
| Remarks: ­ | Pass |

# CHAPTER-9

# Output Screens

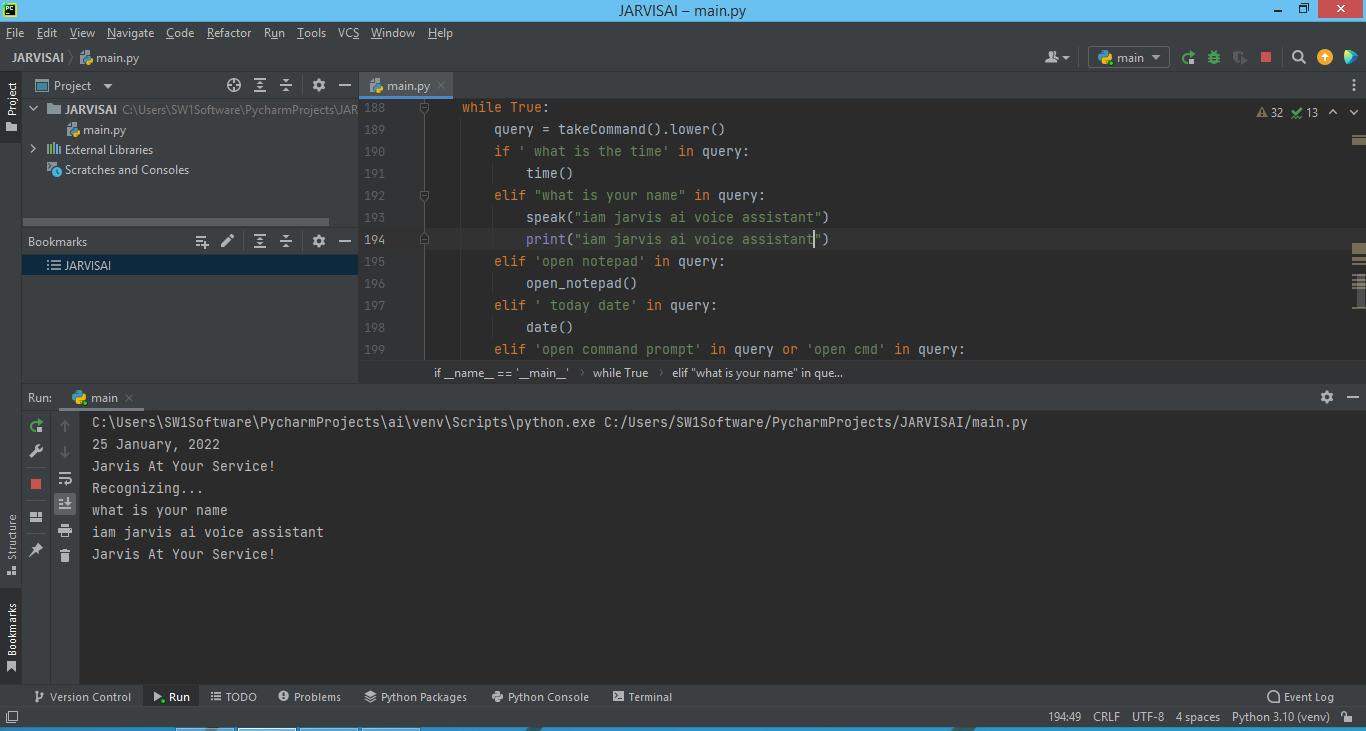
**What is the time?**

Jarvis voice assistant recognizes the voice input (what is the time) and displays the present time.



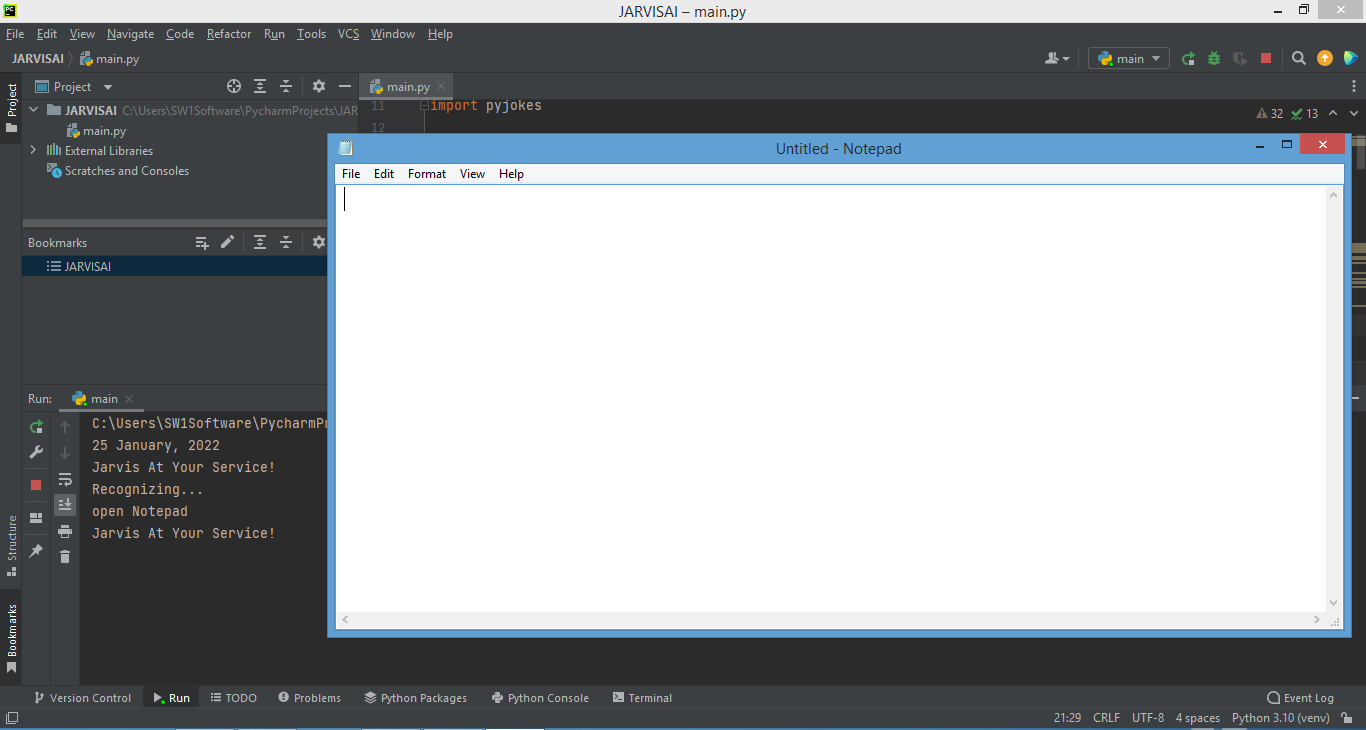
**What is your name?**

It displays its name as JARVIS AL voice assistant.



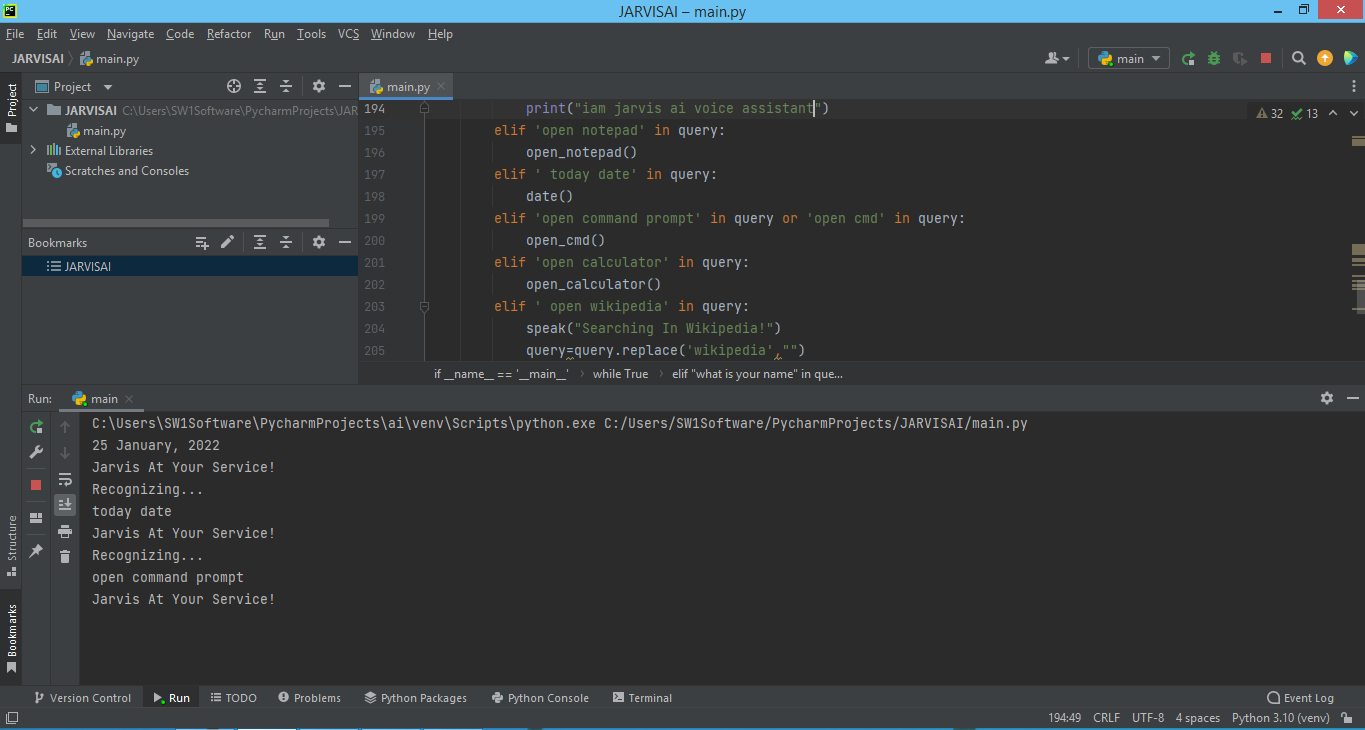
**Open notepad**

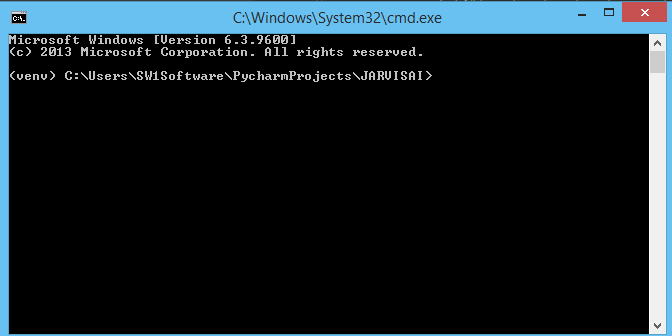
It opens the note which is displayed on the screen.



**Open Command prompt**

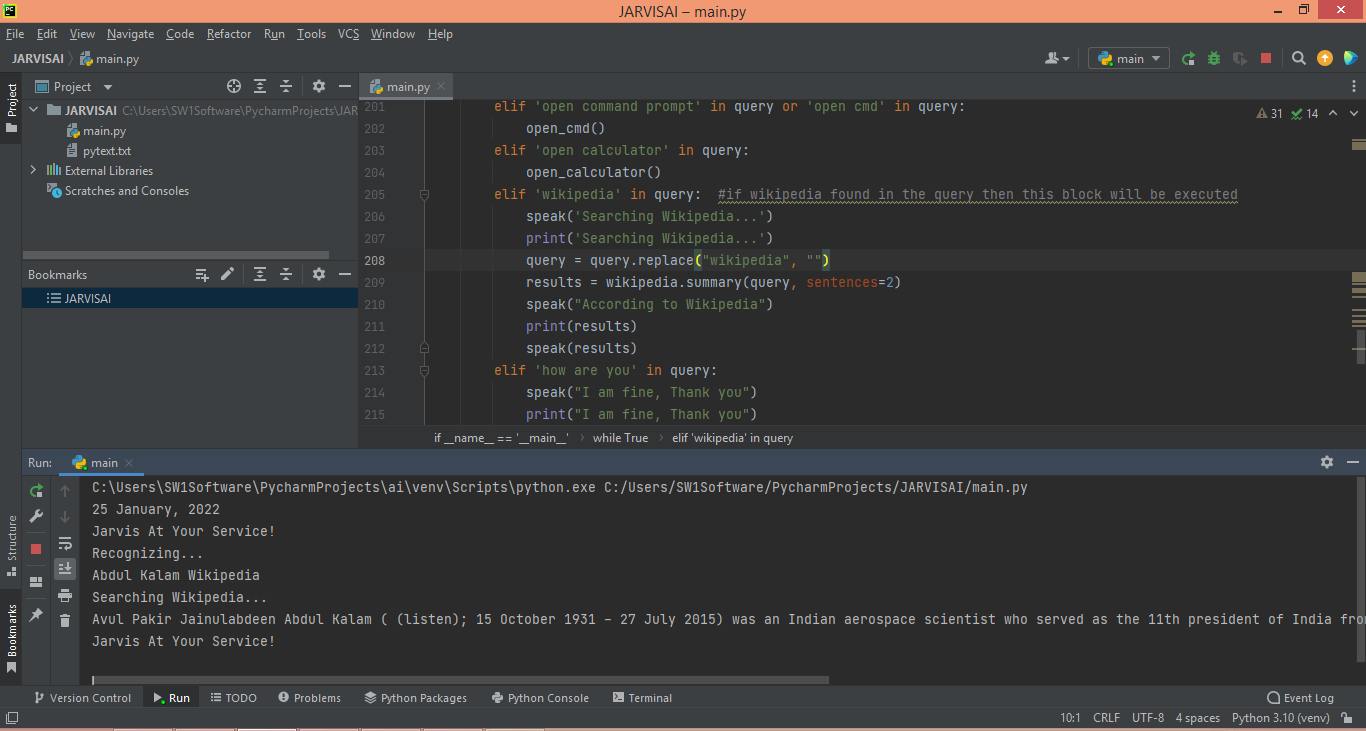
It recognizes voice input (open command prompt).



It displays the command prompt on the screen.

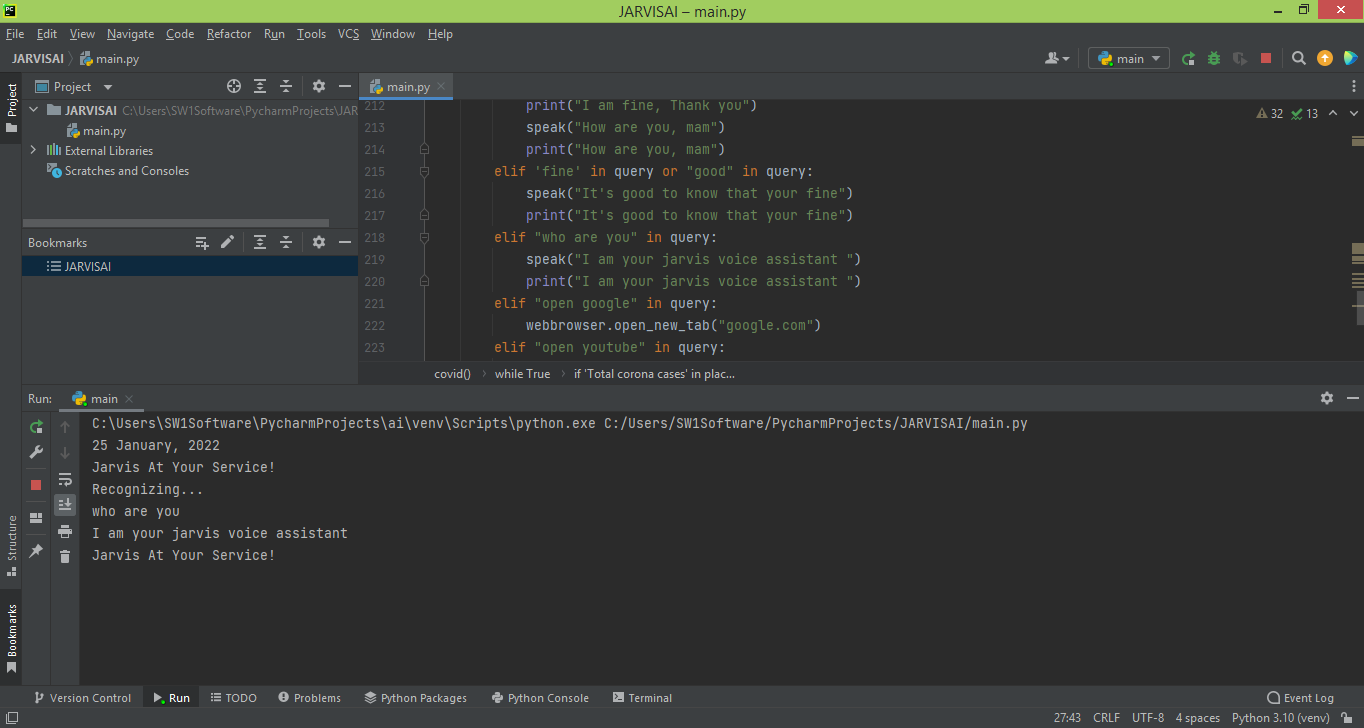
**Wikipedia**

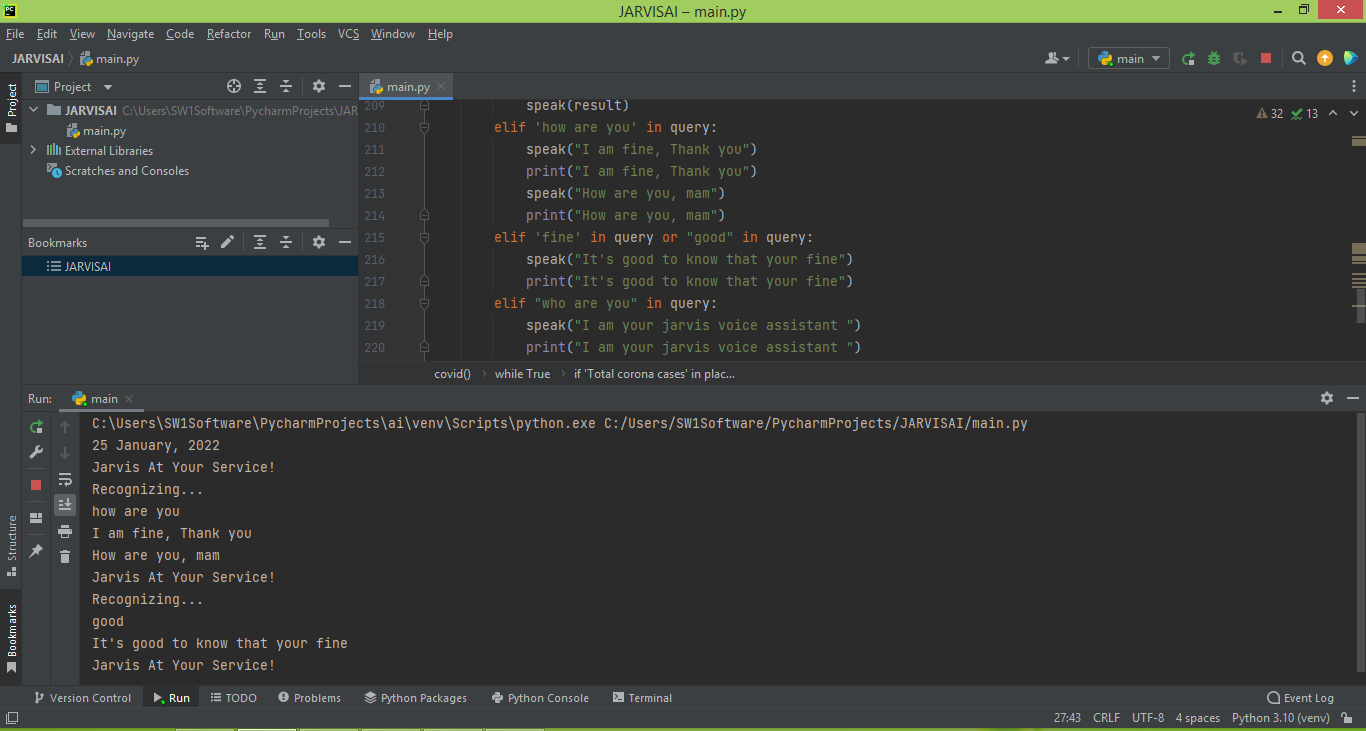
If we give the voice input as Abdul Kalam Wikipedia then it returns the output as below.



**Interactive Mode(Who are you?,How are you?)**

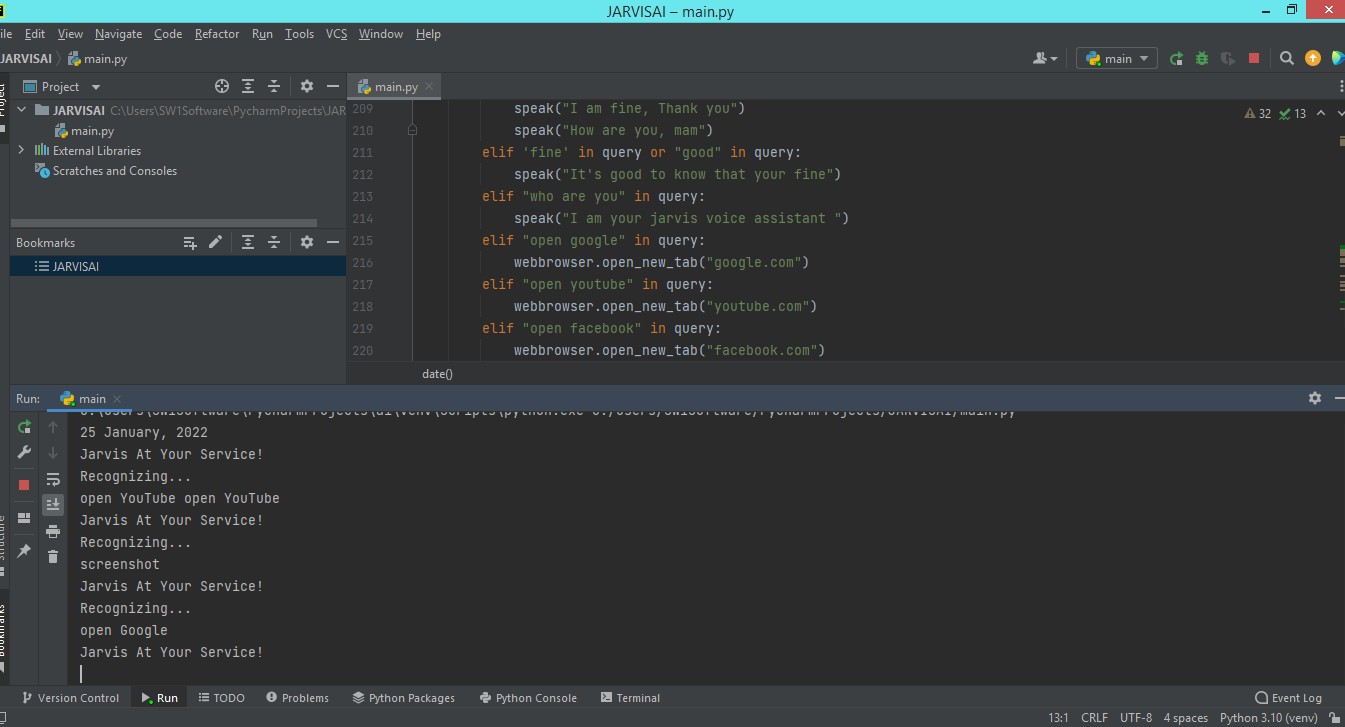
It interactively respond to our command.

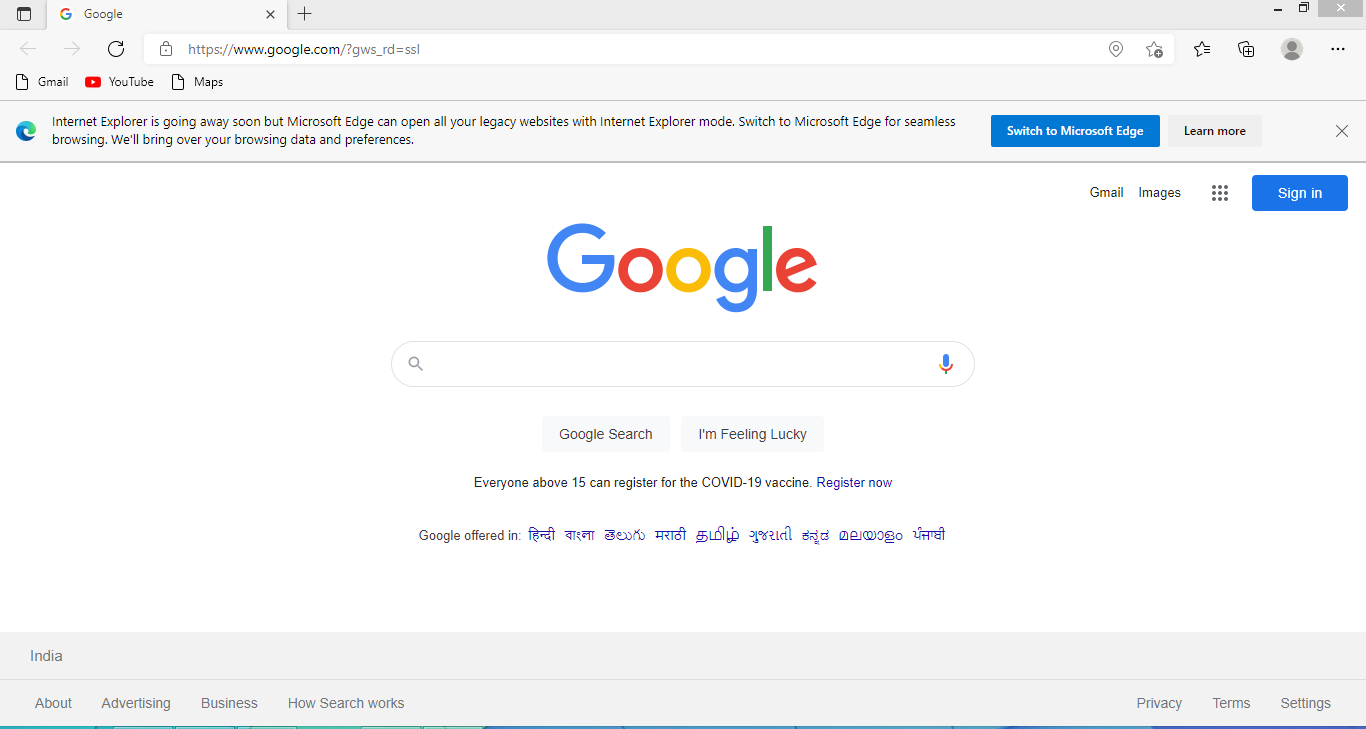




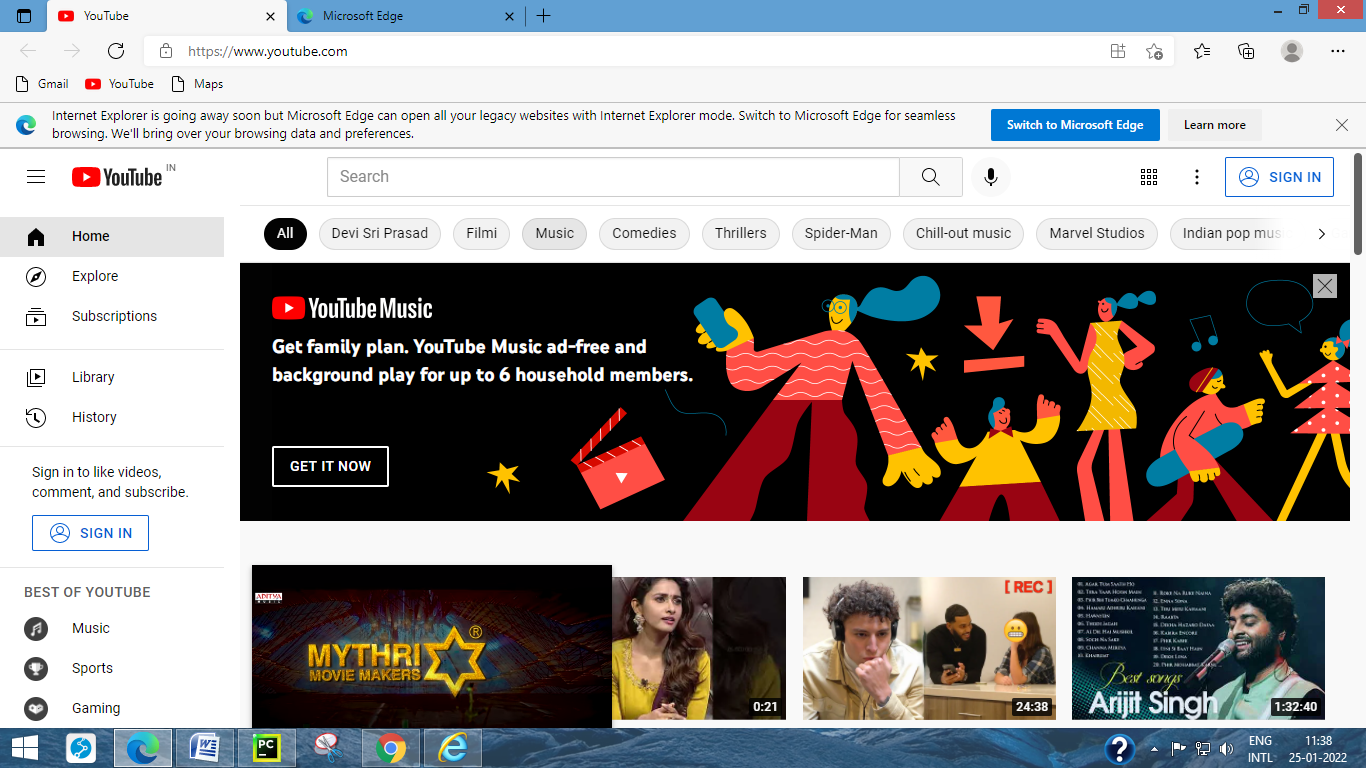
**Open Google**

It opens the Google.



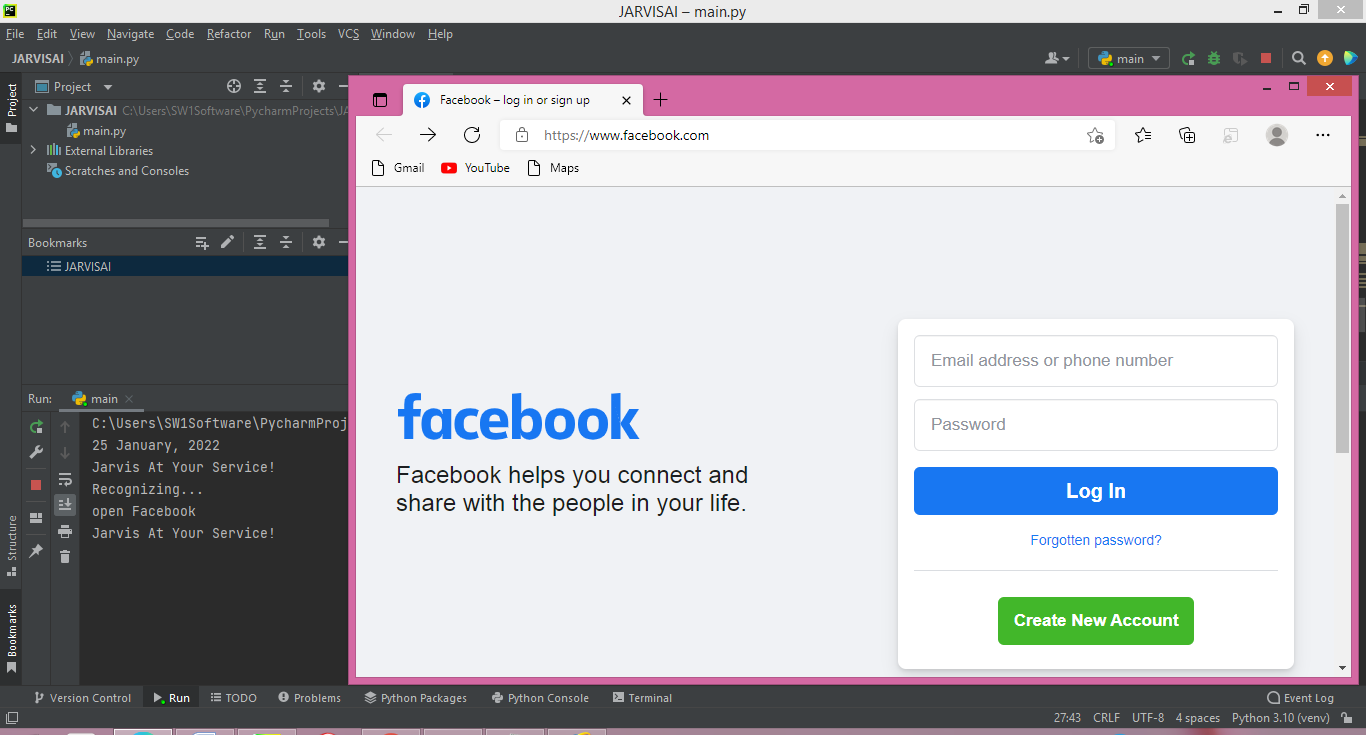
**Open YouTube**

Using voice command it redirects to YouTube..



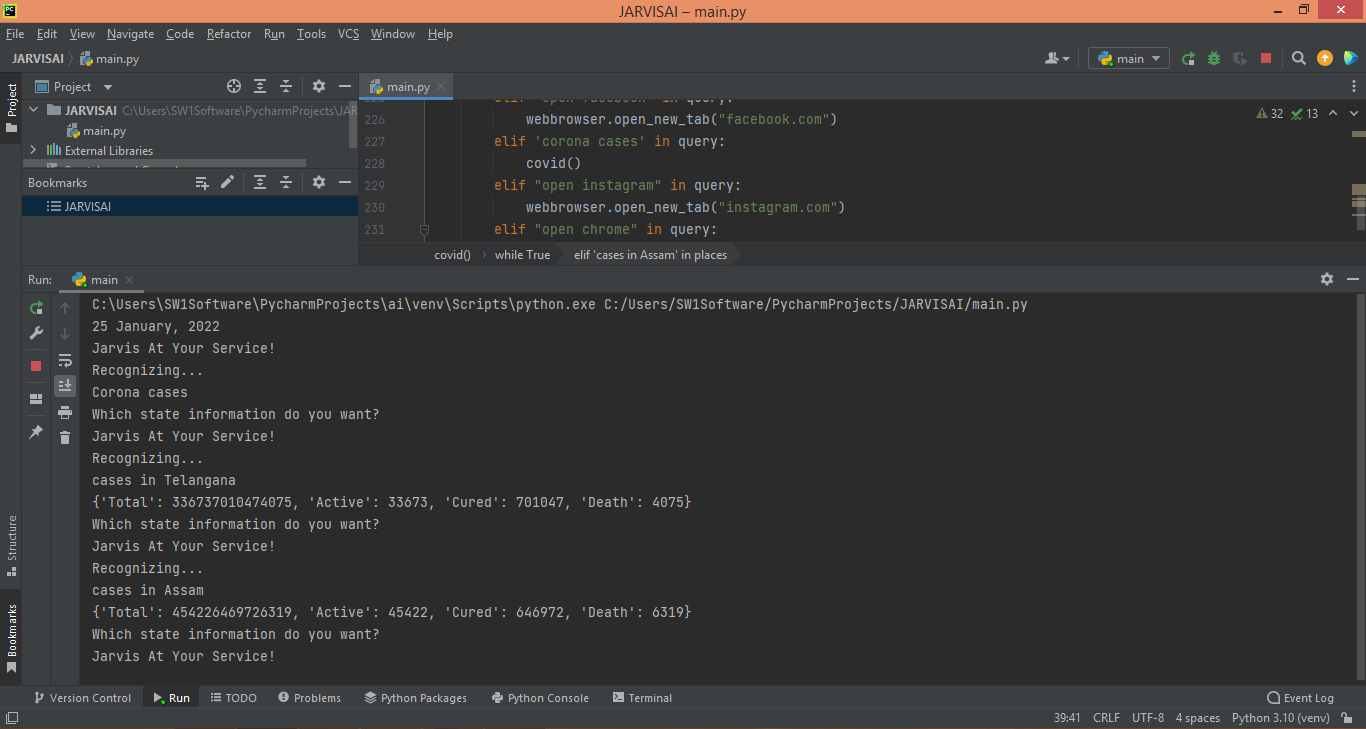
**Open face book**

When we give the command as open face book it directly opens the default facebook.com login page as displayed in the following figure.



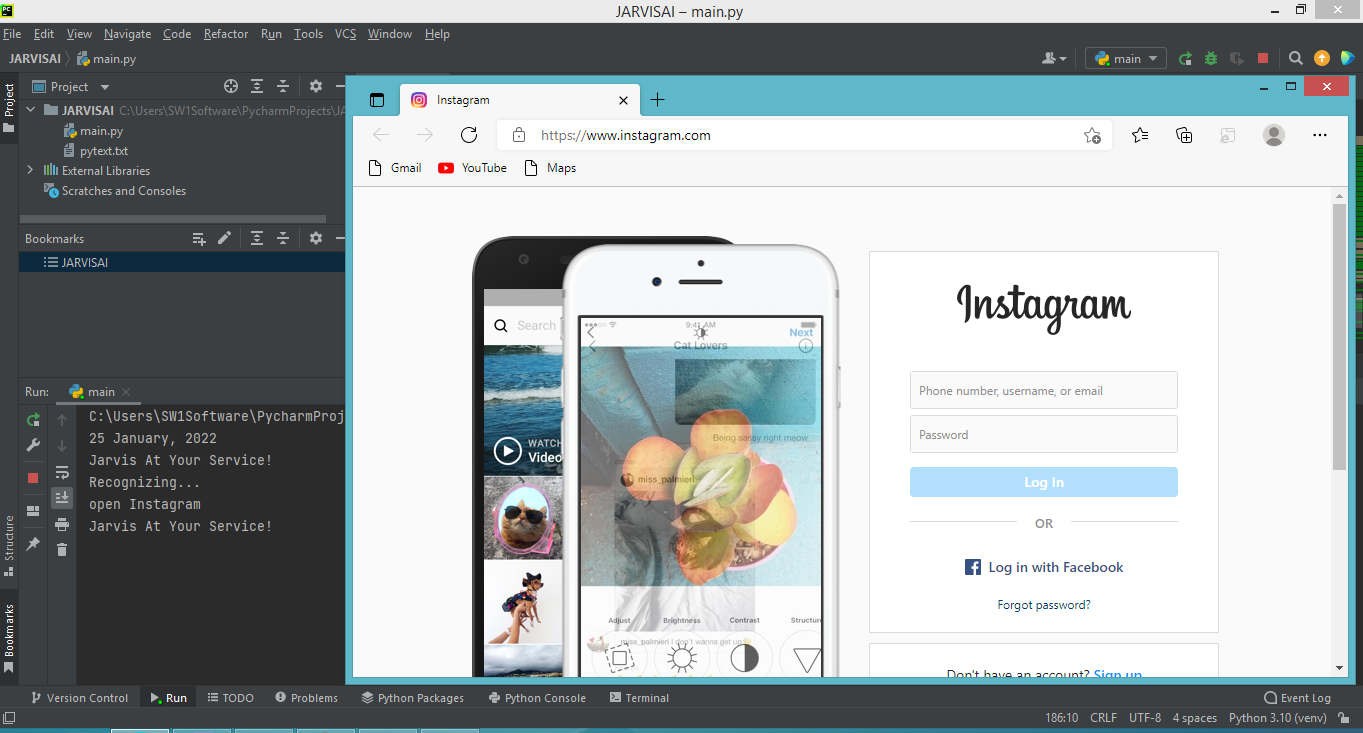
**Corona Cases**

It gives the details of total corona cases and active cases and cured cases and number of deaths in particular specified state.



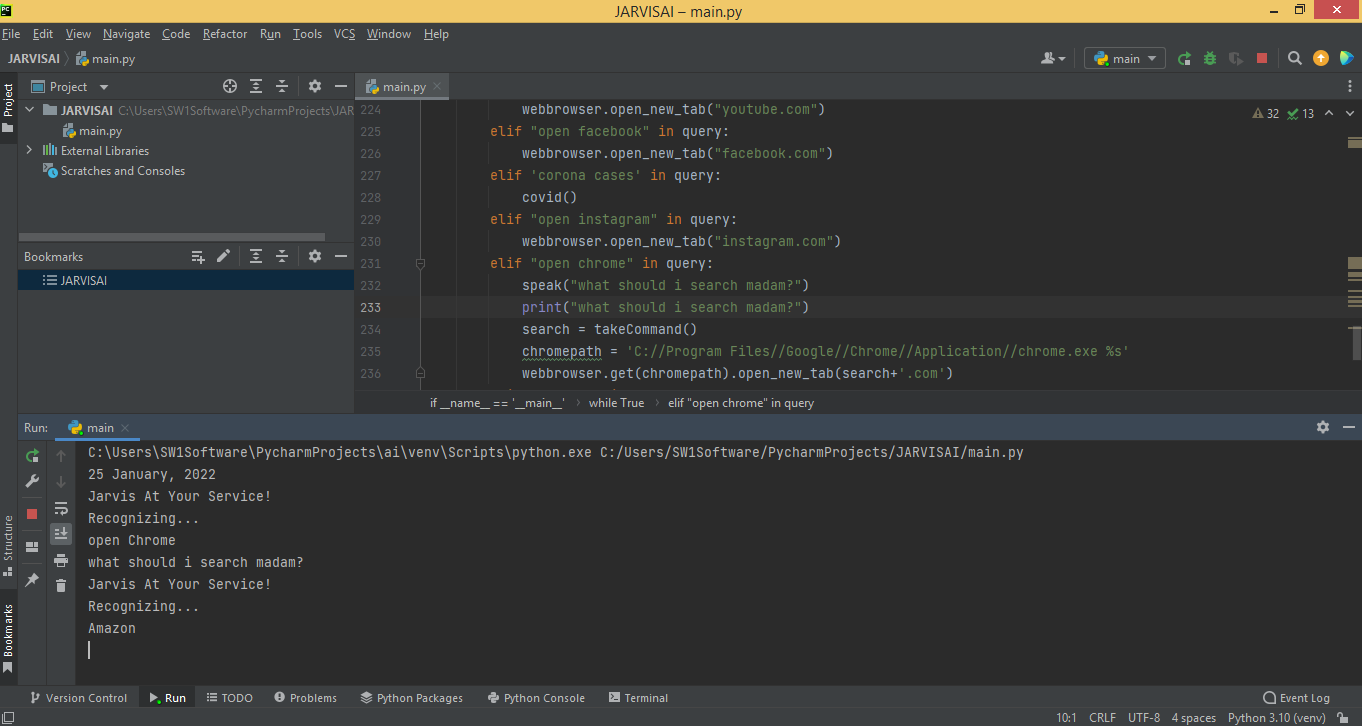
**Open Instagram**

It opens the instagram login page as shown in the following figure.

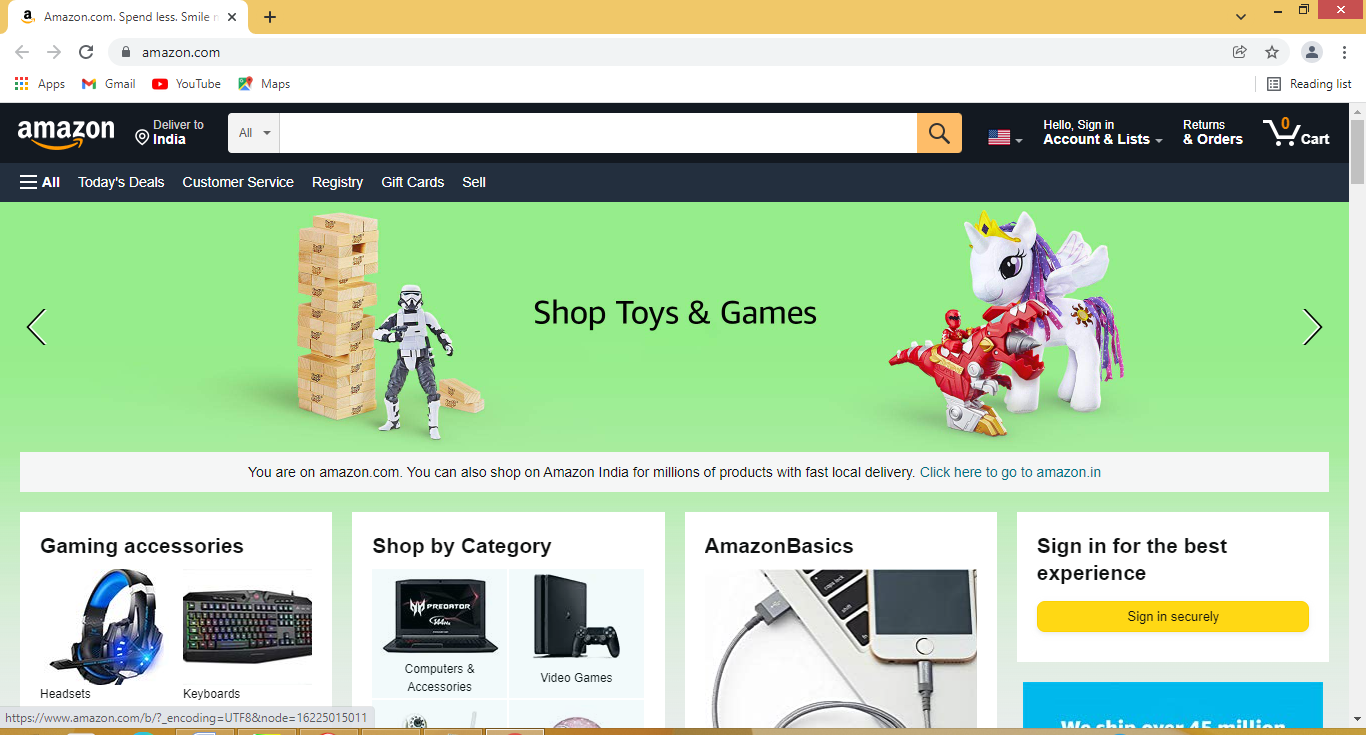


**Open chrome**

It redirects to chrome page

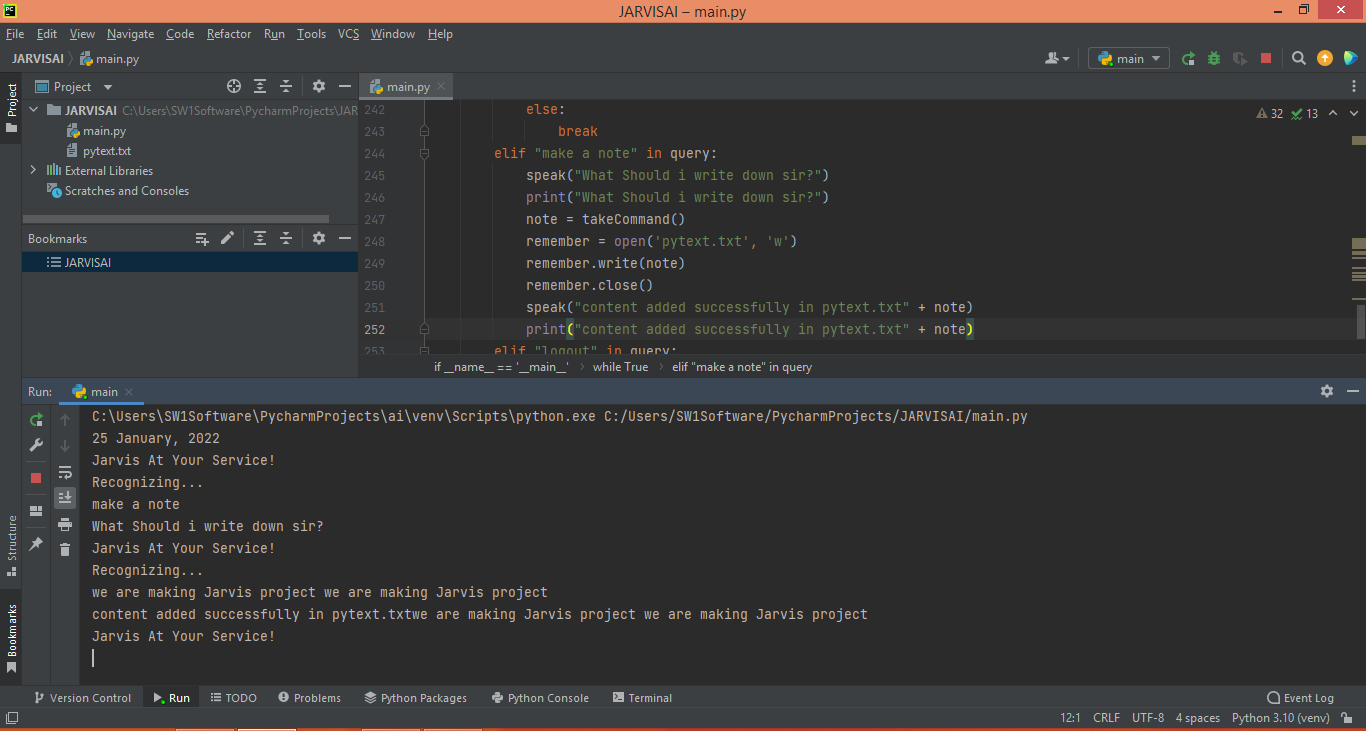


If we give command as “Amazon” it opens the amazon.in web page in the chrome.



**Make a note**

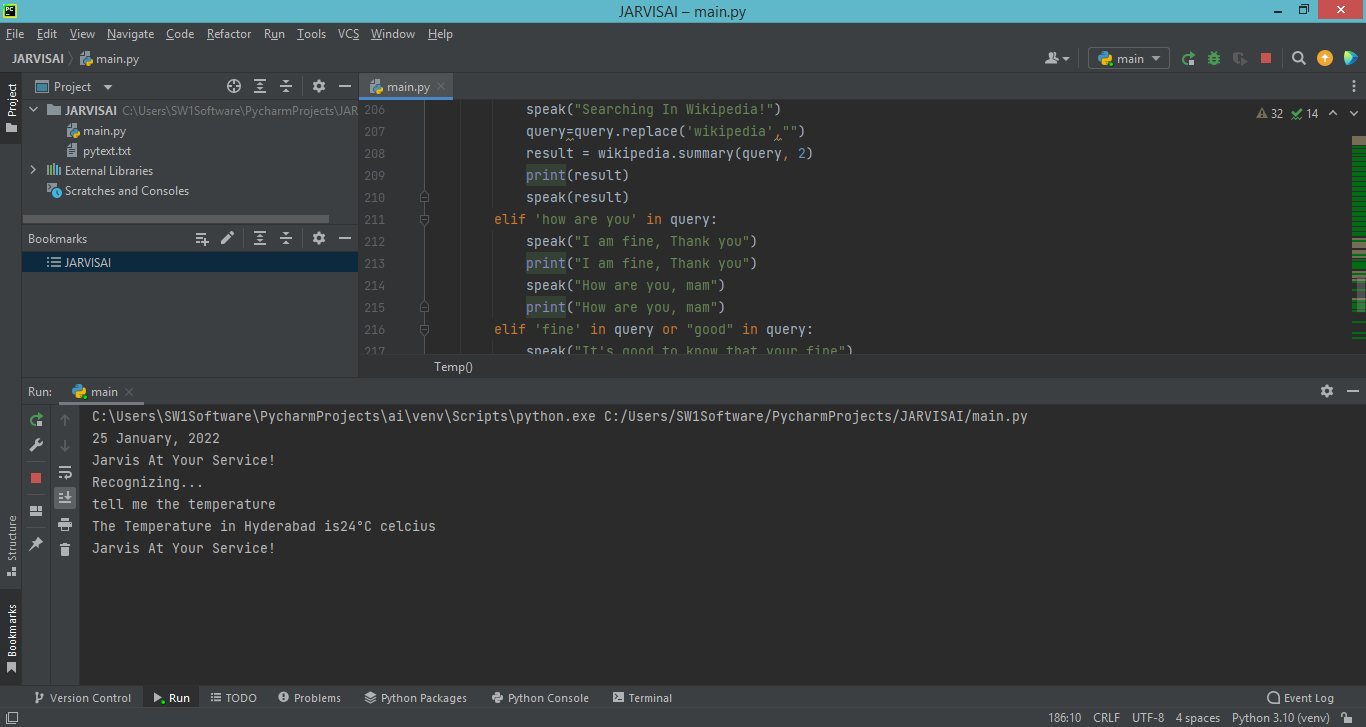
If we want to make a note the Jarvis voice assistant firstly recognize and immediately makes a note as user input.



## Temperature (get weather for a location).

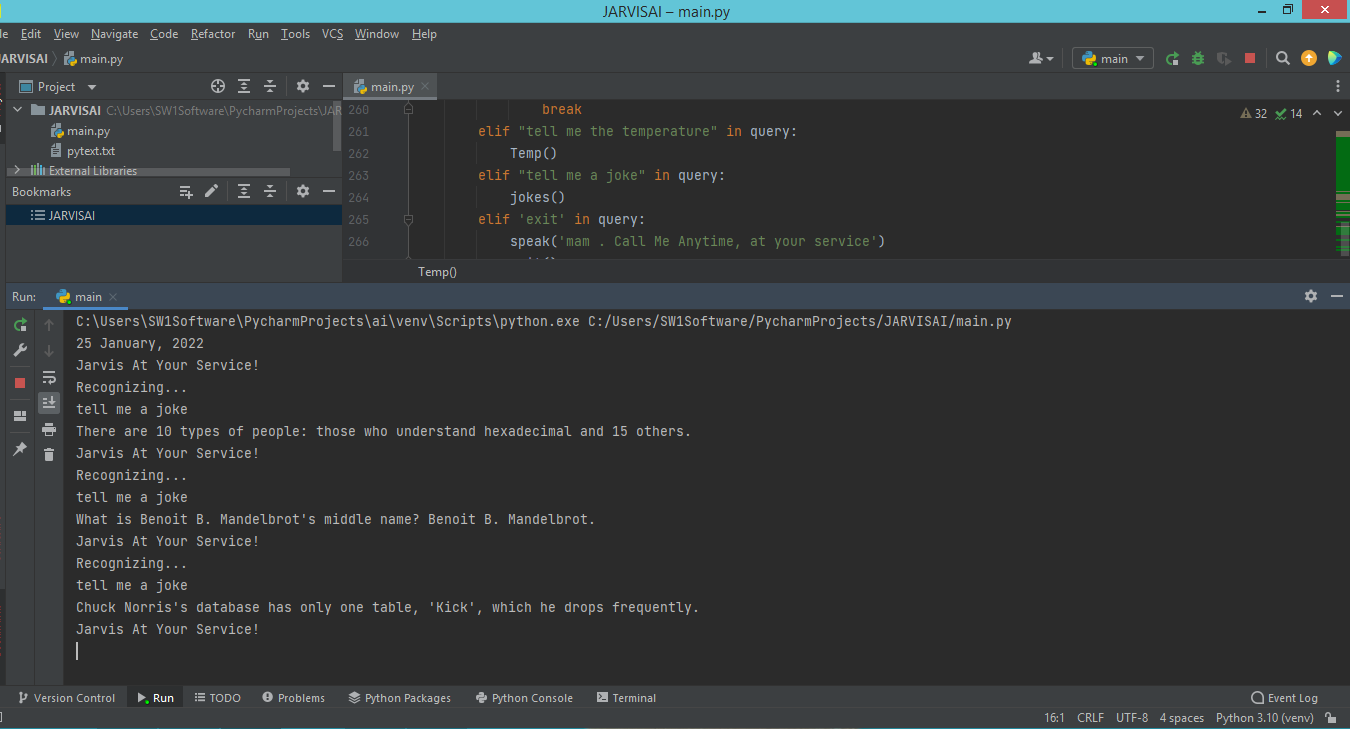
If we give the input as temperature it returns the current temperature in Hyderabad. Getting live weather conditions about a place remains an important task of virtual assistants. It helps the user charter the course of their action. Jarvis addresses this issue with the help of Python.

.



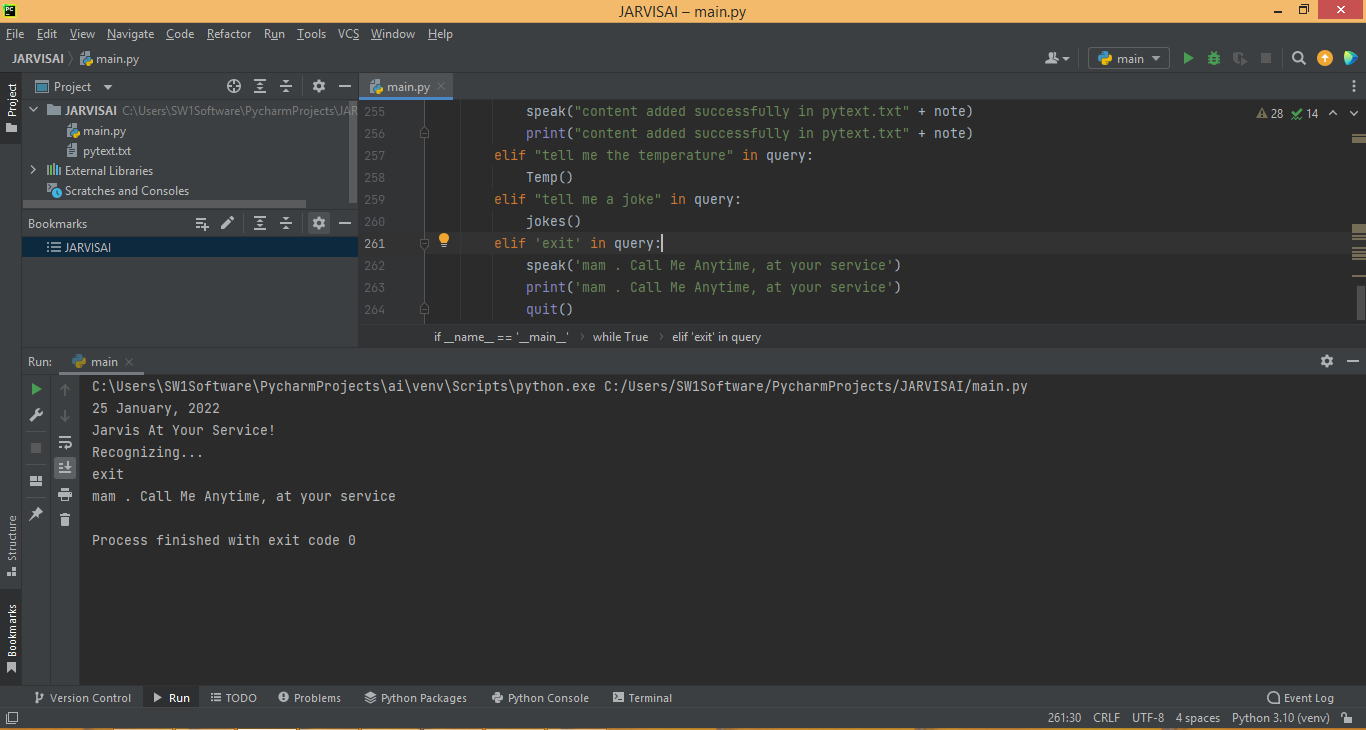
**Joke**

If we want to listen a joke .we can give command as tell me a joke ,the jarvis will recognise command and returns the jokes.



**Exit**

If we to end the communication with the jarvis we can simply use exit command .it displays the message as process finished with exit code.



# CONCLUSION and Future work

. Through this voice assistant, we have automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, opening various apps(instagram,facebook,calculator). The technique was able to authenticate the particular speaker based on the individual information that was included in the voice signal. We aim to make this project a complete server assistant and make it smart enough to act as a replacement for a general server administration. The future plans include integrating Jarvis with mobile using React Native to provide a synchronised experience between the two connected devices.

The current version of our assistant has some good features and is a style well responsive but there's a lot that's yet to be improved. The understanding and reliability of our assistants are often improved more. the longer-term development of the assistant includes merging NLP, Machine learning, and IoT with it. By incorporating these technologies with our assistant, we'll be able to achieve better results. What the virtual assistants are ready to do is way beyond what we've

**FUTURE PROSPECTIVE**

We plan to Integrate Jarvis with mobile using react native, to provide a synchronized experience between the two connected devices. Further, in the long run, Jarvis is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The functionality would be seamless enough to replace the Server Administrator with Jarvis.

# REFERENCES

. https://www.freecodecamp.org/news/python-project-how-to-build-your-own-jarvis-using-python/amp/ 2. https://www.w3schools.com 3. https://www.geeksforgeeks.org/voice-assistant-using-python/amp/

4. Nishitani, Y. Alcohol and Traffic Accidents in Japan. IATSS Res. 2019, 43, 79–83. [CrossRef]

5. Mahata, D.; Narzary, P.K.; Govil, D. Spatio-Temporal Analysis of Road Traffic Accidents in Indian Large Cities. Clin. Epidemiol. Glob. Health 2019, 7, 586–591. [CrossRef]

6. Sheng, H.; Zhao, H.; Huang, J.; Li, N. A Spatio-Velocity Model Based Semantic Event Detection Algorithm for Traffic Surveillance Video. Sci. China Technol. Sci. 2010, 53, 120–125. [CrossRef]

7. Parsa, A.B.; Chauhan, R.S.; Taghipour, H.; Derrible, S.; Mohammadian, A. Applying Deep Learning to Detect Traffic Accidents in Real Time Using Spatiotemporal Sequential Data. arXiv 2019, arXiv:1912.06991.

8. Joshua, S.C.; Garber, N.J. Estimating Truck Accident Rate and Involvements Using Linear and Poisson Regression Models. Transp. Plan. Technol. 1990, 15, 41–58. [CrossRef]

9. Arvin, R.; Kamrani, M.; Khattak, A.J. How Instantaneous Driving Behavior Contributes to Crashes at Intersections: Extracting Useful Information from Connected Vehicle Message Data. Accid. Anal. Prev. 2019, 127, 118–133. [CrossRef] [PubMed] [12] Sh. Ataei et al. ‘Sensor fusion of a railway bridge load test using neural networks’. In: Expert Syst. Appl. 29 (3 2005), pp. 678–683.

[13] A.S. Zaher and S.D.J. McArthur. ‘A Multi-Agent Fault Detection System for Wind Turbine Defect Recognition and Diagnosis’. In: 2007 IEEE Lausanne Powertech. 2007.

[14] Alice E. Smith, David W. Coit and Yun-chia Liang. A Neural Network Approach to Condition Based Maintenance: Case Study of Airport Ground Transportation Vehicles.

[15] Mark Owen. Practical Signal Processing. Cambridge University Press, 2007.