FUTURE INSTITUTE OF ENGINEERING & MANAGEMENT

A PROJECT REPORT

CHATBOT

(In partial fulfilment of the requirements for the award of the degree)

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

UNDER THE GUIDANCE OF:

MAHENDRA DATTA

In association with

(ISO9001:2015)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

(Note: All entries of the proforma of approval should be filled up with appropriate and complete information. Incomplete proforma of approval in any respect will be summarily rejected.)

1. **PROJECT ON:** **CHATBOT**

2. **SUBMITTED BY:** Ankan Das
Aman Yadav
Himanshu Verma
Akashdip Chaterjee

3. **NAME OF THEGUIDE:** **Mr. MAHENDRA DUTTA**

4. **ADDRESS:** Ardent Computech Pvt. Ltd
(An ISO 9001:2015 Certified)
SDF Building, Module #132, Ground Floor, Salt Lake
City, GP Block, Sector V, Kolkata, West Bengal, 700091

PROJECT VERSION CONTROL HISTORY:

Version	Primary Author	Description of Version	Date of Completion
Final	Ankan Das Aman Yadav Himanshu Verma Akashdip Chaterjee	Project Report	

Ankan Das

AkashDip Chattopadhyay

Aman Yadav Himanshu

Mahendra Dutta.

Signature of Students

Date:

Signature of Approver

Date:

MR.MAHENDRA DUTTA

For Office Use Only.

PROJECT PROPOSAL EVALUATOR:

Approved **Not Approved**

DECLARATION

We hereby declare that the project work being presented in the project proposal entitled **"CHATBOT"** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** at **ARDENT COMPUTECH PVT. LTD, SALTLAKE, KOLKATA, WEST BENGAL**, is an authentic work carried out under the guidance of **MR. MAHENDRA DUTTA**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

DATE:

NAME OF THE STUDENTS: ANKAN DAS

AMAN YADAV

HIMANSHU VERMA

AKASHDIP CHATERJEE

SIGNATURE OF THE STUDENTS: Ankan Das

AkashDip Chattopadhyay

Aman Yadav

Himanshu

Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091.

CERTIFICATE

This is to certify that this proposal of minor project entitled **"CHATBOT"** is a record of bonafide work, carried out by **HIMANSHU VERMA, AMAN YADAV, ANKAN DAS, AKASHDIP CHATTERJEE** under my guidance at **ARDENT COMPUTECH PVT LTD.** In my opinion, the report in its present form is in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** and as per regulations of the **ARDENT®.** To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

GUIDE / SUPERVISOR

Mahendra Dutta.

MR. MAHENDRA DUTTA

(Project Engineer)

Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

ACKNOWLEDGEMENT

Success of any project depends largely on the encouragement and guidelines of many others. I take this sincere opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project work.

I would like to show our greatest appreciation to ***Mr. MAHENDRA DUTTA,*** Project Engineer at Ardent, Kolkata. I always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

Words are inadequate in offering our thanks to the other **trainees, project assistants and other members** at ***Ardent Computech Pvt. Ltd*****.** for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

ABSTRACT

Chatbots can be described as software that can chat with people using artificial intelligence. This software is used to perform tasks such as quickly responding to users, informing them, helping to purchase products and providing better service to customers. In this paper, we present the general working principle and the basic concepts of artificial intelligence based chatbots and related concepts as well as their applications in various sectors such as telecommunication, banking, health, customer call centres and e-commerce. Additionally, the results of an example chatbot for donation service developed for telecommunication service provider are presented using the proposed architecture.

Chatbots are programs that work on Artificial Intelligence (AI) & Machine Learning Platform. Chatbot has become more popular in business groups right now as it can reduce customer service costs and handles multiple users at a time. But yet to accomplish many tasks there is a need to make chatbots as efficient as possible. In this project, we provide the design of a chatbot, which provides a genuine and accurate answer for any query using Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA) with python platform.

OBJECTIVE OF THE PROJECT

Digitization is transforming society into a “mobile-first” population. As messaging applications grow in popularity, chatbots are increasingly playing an important role in this mobility-driven transformation. Intelligent conversational chatbots are often interfaces for mobile applications and are changing the way businesses and customers interact.

Chatbots allow businesses to connect with customers in a personal way without the expense of human representatives. Chatbots provide a personal alternative to a written FAQ or guide and can even triage questions, including handing off a customer issue to a live person if the issue becomes too complex for the chatbot to resolve. Chatbots have become popular as a time and money saver for businesses and an added convenience for customers.

CONTENTS

OVERVIEW

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and has fewer syntactical constructions than other languages.

Python is interpreted: Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to Perl and PHP.

Python is Interactive: You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented: Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language: Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

PYTHON

HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, UNIX shell, and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

FEATURES OF PYTHON

Easy-to-learn: Python has few Keywords, simple structure and clearly defined syntax. This allows a student to pick up the language quickly.

Easy-to-Read: Python code is more clearly defined and visible to the eyes.

Easy -to-Maintain: Python's source code is fairly easy-to-maintain.

A broad standard library: Python's bulk of the library is very portable and cross platform compatible on UNIX, Windows, and Macintosh.

Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable: Python can run on the wide variety of hardware platforms and has the same interface on all platforms.

Extendable: You can add low level modules to the python interpreter. These modules enables programmers to add to or customize their tools to be more efficient.

Databases: Python provides interfaces to all major commercial databases.

GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable: Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- It support functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte code for building large applications.
- It provides very high level dynamic datatypes and supports dynamic type checking.

- It supports automatic garbage collections.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA and JAVA.

ENVIRONMENT SETUP

Open a terminal window and type "python" to find out if it is already installed and which version is installed.

- UNIX (Solaris, Linux, FreeBSD, AIX, HP/UX, SunOS, IRIX, etc.)
- Win 9x/NT/2000
- Macintosh (Intel, PPC, 68K)
- OS/2
- DOS (multiple versions)
- PalmOS
- Nokia mobile phones
- Windows CE
- Acorn/RISC OS

BASIC SYNTAX OF PYTHON PROGRAM

Type the following text at the Python prompt and press the Enter –

```
>>> print "Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**.
However in Python version 2.4.3, this produces the following result –

```
Hello, Python!
```

PYTHON IDENTIFIERS

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).
Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language.

PYTHON KEYWORDS

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

And, exec, not
Assert, finally, or
Break, for, pass
Class, from, print

continue, global, raise
def, if, return
del, import, try
elif, in, while
else, is, with
except, lambda, yield

LINES & INDENTATION

Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.
The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. **For example –**

```
if True:
print "True"
else:
print "False"
```

COMMAND LINE ARGUMENTS

Many programs can be run to provide you with some basic information about how they should be run. Python enables you to do this with -h −

```
$ python-h
usage: python [option]...[-c cmd|-m mod | file |-][arg]...

Options and arguments (and corresponding environment variables):

-c cmd: program passed in as string(terminates option list)

-d      : debug output from parser (also PYTHONDEBUG=x)

-E      : ignore environment variables (such as PYTHONPATH)

-h      : print this help message and exit [ etc.]
```

VARIABLE TYPES

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

ASSIGNING VALUES TO VARIABLES

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

```
counter=10            # An integer assignment
weight=10.60          # A floating point
name="Ardent"          # A string
```

MULTIPLE ASSIGNMENT

Python allows you to assign a single value to several variables simultaneously. For example −
a = b = c = 1
a,b,c = 1,2,"hello"

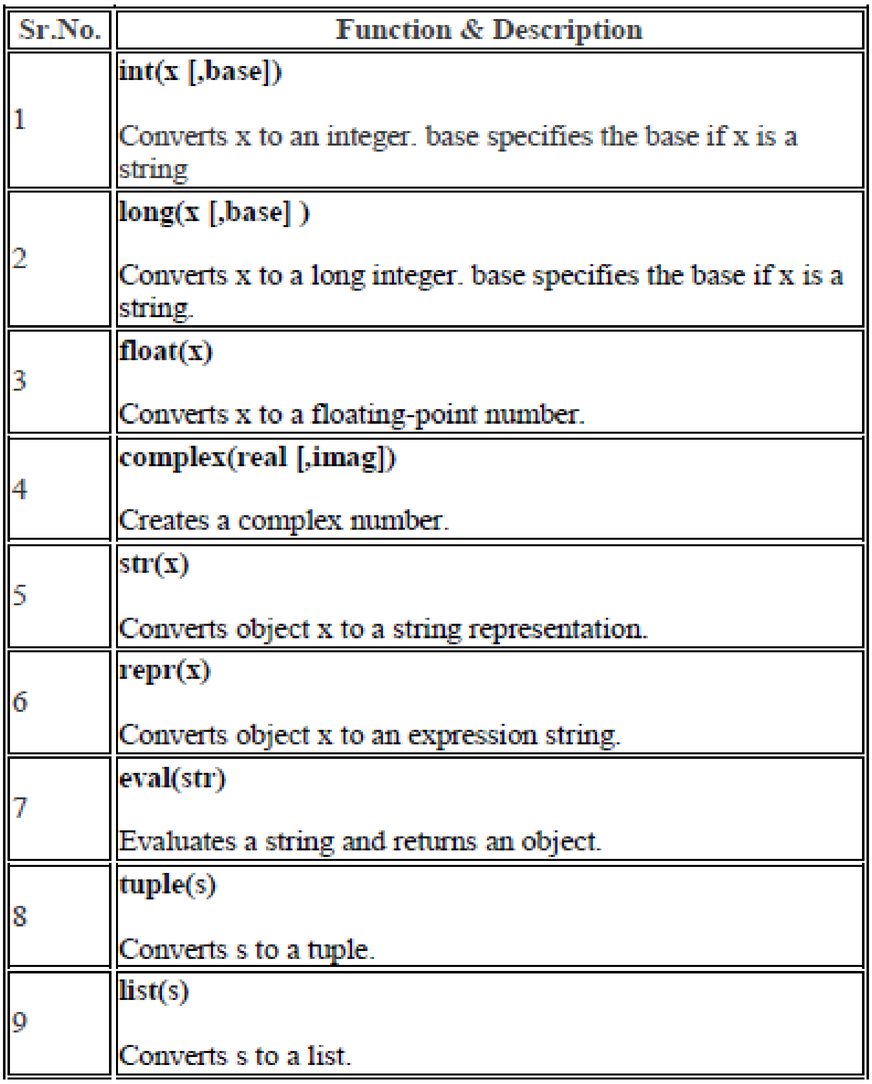
STANDARD DATA TYPES

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has five standard data types −

□ String
□ List
□ Tuple
□ Dictionary
□ Number

DATA TYPE CONVERSION

Sometimes, you may need to perform conversions between the built-in types. To convert between types, you simply use the type name as a function.
There are several built-in functions to perform conversion from one data type to another.



Sr.No.	Function & Description
1	**int(x [,base])** Converts x to an integer. base specifies the base if x is a string
2	**long(x [,base])** Converts x to a long integer. base specifies the base if x is a string.
3	**float(x)** Converts x to a floating-point number.
4	**complex(real [,imag])** Creates a complex number.
5	**str(x)** Converts object x to a string representation.
6	**repr(x)** Converts object x to an expression string.
7	**eval(str)** Evaluates a string and returns an object.
8	**tuple(s)** Converts s to a tuple.
9	**list(s)** Converts s to a list.

FUNCTIONS

DEFINING A FUNCTION:

- def function name(parameters):

```
    "function_docstring"
    function suite
    return [expression]
```

PASS BY REFERENCE VS PASS BY VALUE

All parameters (arguments) in the Python language are passed by reference. It means if you change what a parameter refers to within a function, the change also reflects back in the calling function. For example –

# Function definition is here

```
def change me(mylist):
        "This changes a passed list into this function"
        mylist.append([1,2,3,4]);
        print"Values inside the function: ",mylist
        return
```

# Now you can call changeme function

```
mylist=[10,20,30];
change me(mylist);
print" Values outside the function: ",mylist
```

Here, we are maintaining reference of the passed object and appending values in the same object. So, this would produce the following result −

```
Values inside the function: [10, 20, 30, [1, 2, 3, 4]]
Values outside the function: [10, 20, 30, [1, 2, 3, 4]]
```

GLOBAL VS. LOCAL VARIABLES

Variables that are defined inside a function body have a local scope, and those defined outside have a global scope . For Example-

```
total=0;                    # This is global variable.
```

# Function definition is here

```
def sum( arg1, arg2 ):
```

# Add both the parameters and return them."

```
total= arg1 + arg2;      # Here total is local variable.
print"Inside the function local total: ", total
return total;
```

```
# Now you can call sum function

sum(10,20);
Print”Outside the function global total: ", total
```

When the above code is executed, it produces the following result −

```
Inside the function local total: 30
Outside the function global total: 0
```

MODULES

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

The Python code for a module named *aname* normally resides in a file named *aname.py*. Here's an example of a simple module, support.py

```
def print_func( par ):
        print"Hello : ", par
        return
```

The *import* Statement

You can use any Python source file as a module by executing an import statement in some other Python source file. The *import* has the following syntax –

```
Import module1 [, module2 [… moduleN]
```

PACKAGES

A package is a hierarchical file directory structure that defines a single Python application environment that consists of modules and sub packages and sub-subpackages, and so on.
Consider a file *Pots.py* available in *Phone* directory. This file has following line of source code –

```
def Pots ():
        print "I'm Pots Phone"
```

Similar way, we have another two files having different functions with the same name as above –

□ *Phone/Isdn.py* file having function Isdn ()

□ *Phone/G3.py* file having function G3 ()

Now, create one more file __init__.py in *Phone* directory −

□ Phone/__init__.py

To make all of your functions available when you've imported Phone, you need to put explicit import statements in __init__.py as follows −

```
from Pots import Pots
from Isdn import Isdn
from G3 import
```

ARTIFICIAL INTELLIGENCE

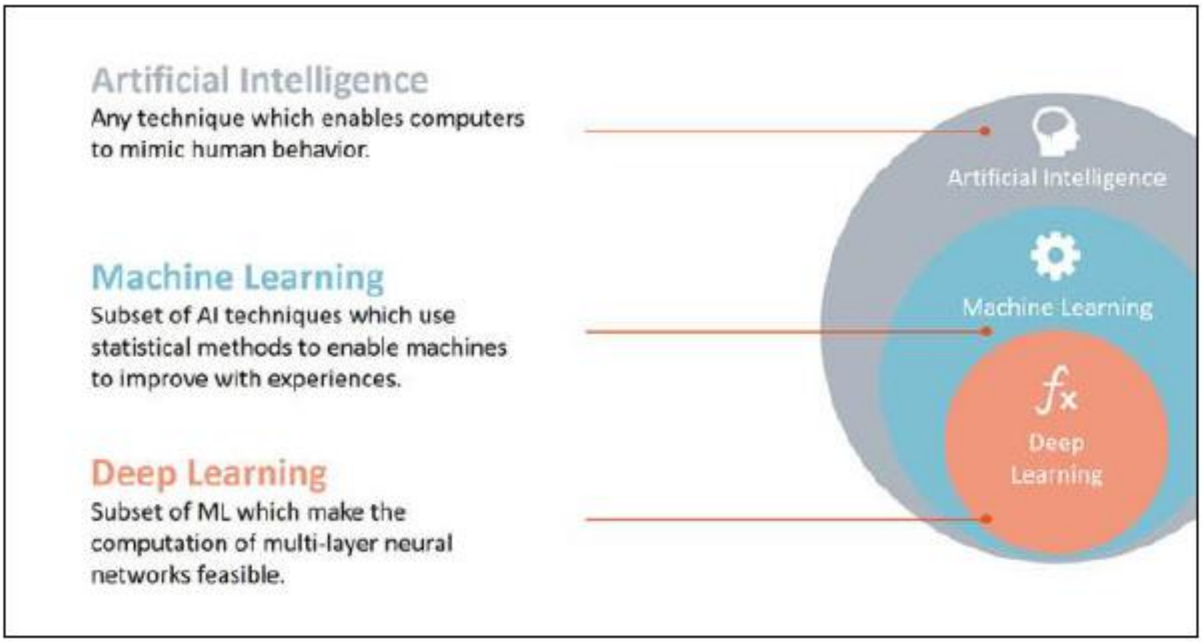
INTRODUCTION:

According to the father of Artificial Intelligence, John McCarthy, it is *“The science and engineering of making intelligent machines, especially intelligent computer programs”*.

Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

The development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.



Artificial Intelligence
Any technique which enables computers to mimic human behavior.

Machine Learning
Subset of AI techniques which use statistical methods to enable machines to improve with experiences.

Deep Learning
Subset of ML which make the computation of multi-layer neural networks feasible.

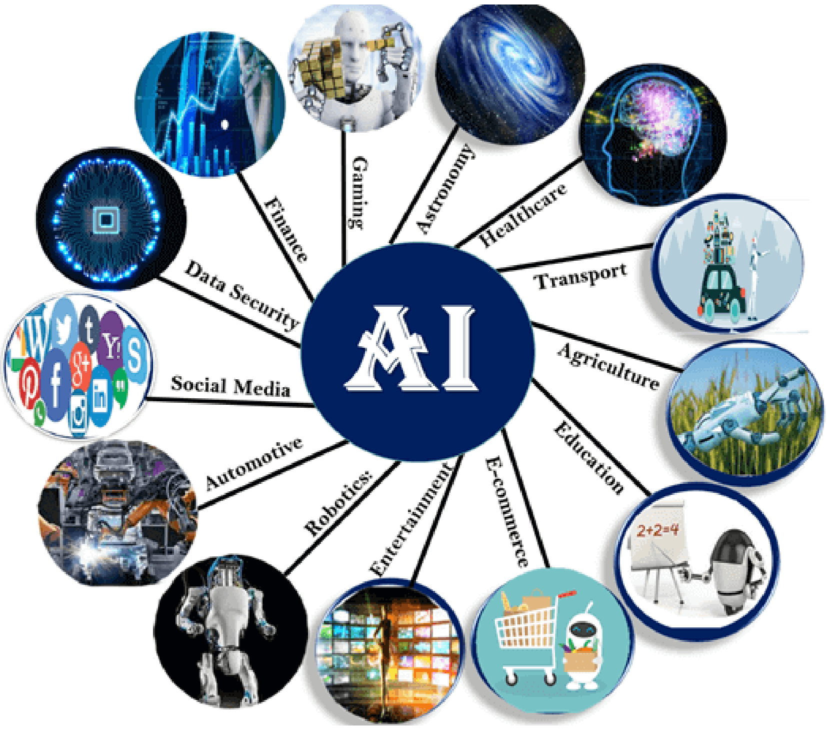
GOALS OF AI

To Create Expert Systems − The systems which exhibit intelligent behaviour, learn, demonstrate, explain, and advice its users.
To Implement Human Intelligence in Machines − Creating systems that understand, think, learn, and behave like humans.

APPLICATIONS OF AI

AI has been dominant in various fields such as: -

Gaming − AI plays crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machine can think of large number of possible positions based on heuristic knowledge.



Natural Language Processing − It is possible to interact with the computer that understands natural language spoken by humans.

Expert Systems − There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanation and advice to the users.

Vision Systems − These systems understand, interpret, and comprehend visual input on the computer.

For example: A spying aeroplane takes photographs, which are used to figure out spatial information

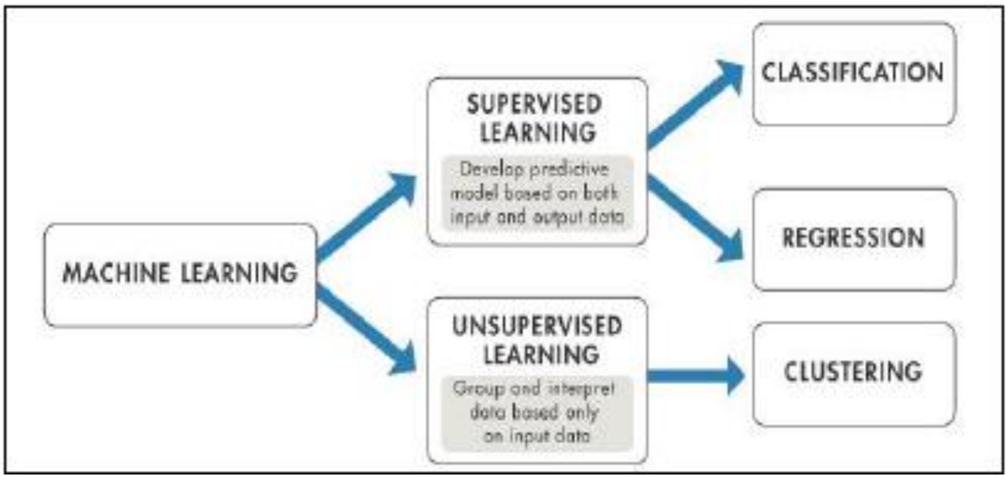
1. Or map of the areas.
2. Doctors use clinical expert system to diagnose the patient.
3. Police use computer software that can recognize the face of criminal with the stored portrait made by forensic artist.

Speech Recognition − Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it. It can handle different accents, slang words, noise in the background, change in human's noise due to cold, etc.

Handwriting Recognition − The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus. It can recognize the shapes of the letters and convert it into editable text.

Intelligent Robots − Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment.

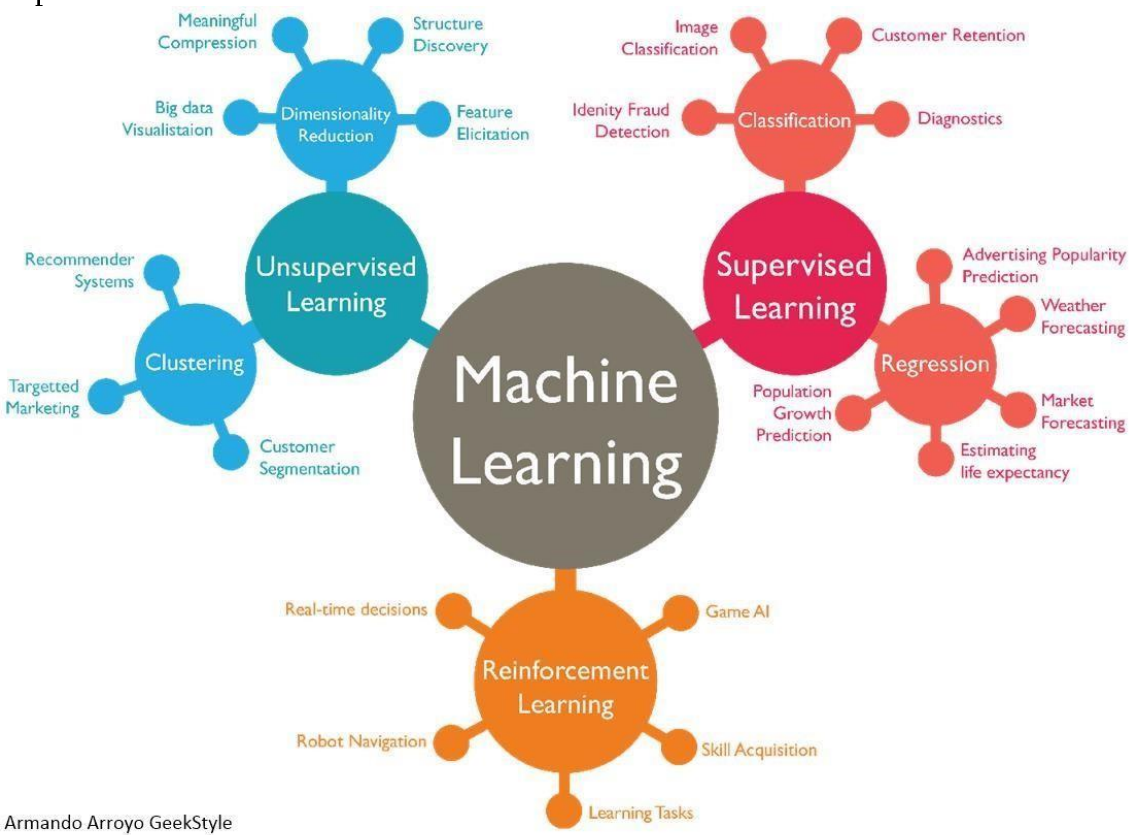
Machine Learning − Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed.
Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data.


MACHINE LEARNING
SUPERVISED LEARNING
Develop predictive model based on both input and output data
UNSUPERVISED LEARNING
Group and interpret data based only on input data
CLASSIFICATION
REGRESSION
CLUSTERING

INTRODUCTION TO MACHINE LEARNING

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed.

Arthur Samuel, an American pioneer in the field of computer gaming and artificial intelligence, coined the term "Machine Learning" in 1959 while at IBM. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data.



Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system:-

1. **Supervised learning** is the machine learning task of inferring a function from *labelled training data*.[1] The training data consist of a set of *training examples*. In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value.
2. **Unsupervised learning** is the machine learning task of inferring a function to describe hidden structure from "unlabelled" data (a classification or categorization is not included in the observations). Since the examples given to the learner are unlabelled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm—which is one way of distinguishing unsupervised learning from supervised learning and reinforcement learning.

NUMPY:

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin.

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars.

NUMPY ARRAY:

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank*.

For example, the coordinates of a point in 3D space [1, 2, 1] is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[1., 0., 0.],
 [ 0., 1., 2.]]
```

NumPy's array class is called *ndarray*. It is also known by the alias.

SLICING NUMPY ARRAY:

```
Import numpy as np

a = np.array ([[1, 2, 3],[3,4,5],[4,5,6]])

print 'Our array is:'
Print a
print '\n'
print 'The items in the second column are:'
print a[...,1]
print '\n'
print 'The items in the second row are:'
print a[1...]
print '\n'
print 'The items columns 1 onwards are:'
print a [...,1:]
```

OUTPUT:

```
Our array is:
[[1 2 3]
[3 4 5]
[4 5 6]]
The items in the second column are:
[2 4 5]
The items in the second row are:
[3 4 5]
The items column 1 onwards are:
[[2 3]
[4 5]
[5 6]]
```

SCIPY:

modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack.

THE SCIPY LIBRARY/PACKAGE:

The SciPy package of key algorithms and functions core to Python's scientific computing capabilities. Available sub-packages include:

- **constants:** physical constants and conversion factors (since version 0.7.0)
- **cluster:** hierarchical clustering, vector quantization, K-means
- **fftpack:** Discrete Fourier Transform algorithms
- **integrate:** numerical integration routines
- **interpolate:** interpolation tools
- **io:** data input and output
- **lib:** Python wrappers to external libraries
- **linalg:** linear algebra routines
- **misc:** miscellaneous utilities (e.g. image reading/writing)
- **ndimage:** various functions for multi-dimensional image processing
- **optimize:** optimization algorithms including linear programming
- **signal:** signal processing tools
- **sparse:** sparse matrix and related algorithms
- **spatial:** KD-trees, nearest neighbours, distance functions
- **special:** special functions
- **stats:** statistical functions
- **weave:** tool for writing C/C++ code as Python multiline strings

DATA STRUCTURES:

The basic data structure used by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for linear algebra, Fourier transforms and random number generation, but not with the generality of the equivalent functions in SciPy. NumPy can also be used as an efficient multi-dimensional container of data with arbitrary data-types. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases. Older versions of SciPy used Numeric as an array type, which is now deprecated in favour of the newer NumPy array code.

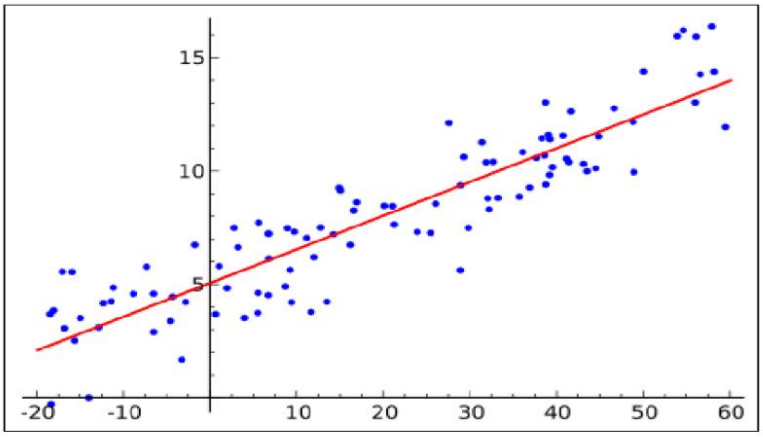
SCIKIT-LEARN:

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.[4] The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010[5]. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012.

REGRESSION ANALYSIS

In statistical modelling, **regression analysis** is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.



Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer casual relationships between the independent and dependent variables. However this can lead to illusions or false relationships, so caution is advisable

LINEAR REGRESSION

Linear regression is a linear approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X. The case of one explanatory variable is called *simple linear regression*. For more than one explanatory variable, the process is called *multiple linear regression*.

In linear regression, the relationships are modelled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called *linear models*.

LOGISTIC REGRESSION

Logistic regression, or logit regression, or logit model [1] is a regression model where the dependent variable (DV) is categorical. This article covers the case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression. In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

POLYNOMIAL REGRESSION

Polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an n^{th} degree polynomial in x.

Polynomial regression fits a nonlinear relationship between the value of x and the corresponding conditional mean of y , denoted E(y | x), and has been used to describe nonlinear phenomena such as the growth rate of tissues, the distribution of carbon isotopes in lake sediments, and the progression of disease epidemics.

Although *polynomial regression* fits a nonlinear model to the data, as a statistical estimation problem it is linear, in the sense that the regression function E(y | x) is linear in the unknown parameters that are estimated from the data.

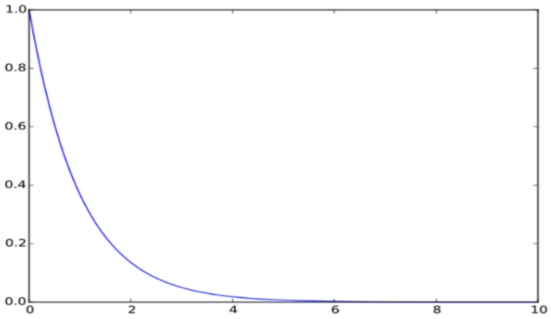
MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter,wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged .SciPy makes use of matplotlib.

EXAMPLE:

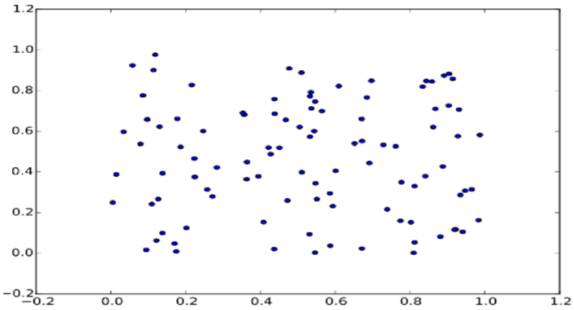
1. LINE PLOT

```
>>>importmatplotlib.pyplotasplt
>>>importnumpyasnp
>>> a =np.linspace(0,10,100)
>>> b =np.exp(-a)
>>>plt.plot (a,b)
>>>plt.show ()
```



2. SCATTER PLOT

```
>>>importmatplotlib.pyplotasplt
>>>fromnumpy.randomimport rand
>>> a =rand(100)
>>> b =rand(100)
>>>plt.scatter(a, b)
>>>plt.show ()
```



PANDAS

In computer programming, **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. "Panel data", an econometrics term for multidimensional, structured data sets.

LIBRARY FEATURES:

- Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and sub setting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.

- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation.

CLUSTERING

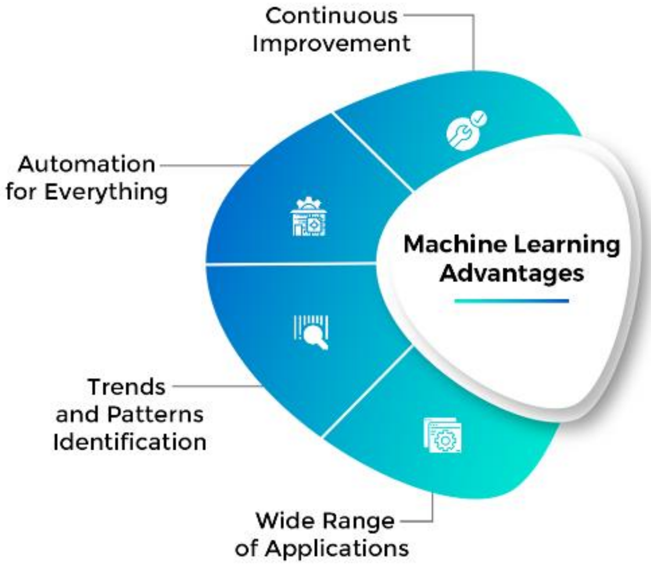
Cluster analysis or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, and computer graphics.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem.

The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data pre-processing and model parameters until the resul achieves the desired properties.

ADVANTAGES OF ML:

- It gives fast and real time predictions to problems.
- Efficiently utilizes theresources.
- Helps in automation of different tasks.
- It is used a lot in different sectors of life like business, medicine, sports etc.
- Helps interpret previous behaviour of model.
- Easily identifies trends and patterns.
- No human intervention needed (automation)
- Continuous Improvement.
- Handling multi-dimensional and multi-variety data


Continuous Improvement
Automation for Everything
Machine Learning Advantages
Trends and Patterns Identification
Wide Range of Applications

CHATBOT

INTRODUCTION:

A chatbot is an automated software program that interacts with humans. A chatbot is merely a computer program that fundamentally simulates human conversations. A chatbot that functions through AI and machine learning have an artificial neural network inspired by the neural nodes of the human brain. Chatbots are programs that can do talk like human conversations very easily. For example, Facebook has a machine learning chatbot that creates a platform for companies to interact with their consumers through the Facebook Messenger application. In 2016, chatbots became too popular on Messenger. By the consequences is noted that 2016 was the entire year of chatbots. The software industry is mainly oriented on chatbots. Thousands of chatbots are invented on start-ups and used by the businesses to improve their customer service, keeping them hanging by a kind communication. According to research, nowadays chatbots are used to solve a number of business tasks across many industries like E-Commerce, Insurance, Banking, Healthcare, Finance, Legal, Telecom, Logistics, Retail, Auto, Leisure, Travel, Sports, Entertainment, Media and many others. Thus that was the moment to look at the chatbots as a new technology in the communication field. Nowadays various companies are using chatbots to answer quickly and efficiently some frequented asking questions from their own customers.

AIML and LSA are used for creating chatbots. Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA) are used for developing chatbots, which are used to define general pattern-based queries. This pattern can also be used to give random responses for the same query in the chatbot. LSA is a Latent Semantic Analysis technology in python, which is utilized to discover likenesses between words as vector representation. So that the unanswered queries by AIML will be viewed as a reply by LSA.

CHATBOTS: ARE THEY REALLY USEFUL?

If the chatbot is powered by artificial intelligence, it provides a better conversation experience. With AI technology and cognitive technology, it can understand user content, purpose, reasons, while learning and communicating intelligently with people based on programmed programs.

These technologies enable your applications to see, hear, interpret, and interact more humanely. Therefore, integrating chatbots with cognitive services and equipping them with customized data analytics capabilities can provide your enterprise with rich insights and discoveries to make business decisions easily.

A well-built AI-powered chatbot should be able to understand the meaning from randomly unstructured data - which means any raw data that lies somewhere in your systems. It can help you transform customer engagement, improve decision making, and build self-learning capabilities that can greatly benefit business processes using this data.

ARTIFICIAL INTELLIGENCE MARKUP LANGUAGE

Extensible Markup Language (XML) is the base for the derivation of Artificial Intelligence Markup Language (AIML). It has a class of data object called an AIML object that describes the behaviour of computer programs. It consists of units or tag called topics and categories. In AIML, categories are basic units of knowledge. There each category consists of a pattern that contains input and template which contain the answer of chatbot based on queries. To build a Chatbot, mainly a flexible, easy to understand and universal language is needed which will be AIML. AIML, a derivative of XML, is one of the widely used approaches that satisfy the requirements based on general queries. AIML represents the knowledge put into Chatbots and is based on the software technology developed for A.L.I.C.E. (the Artificial Linguistic Internet Computer Entity). It has the ability to characterize the type of data object and describe partial conductance of the programs that it processes. These objects consist of two units: topics and categories. Thus, the data contained in these categories are either parsed or unparsed.

The purpose of the AIML language is to simplify the job of conversational modelling, in a "stimulus-response" process. It is also a mark-up language based on XML and depends on tags which are the identifiers that make snippets of codes to send commands into the Chatbot. The data object class is defined in AIML as an AIML object, and the responsibility of these objects is modelling conversational patterns. Each AIML object is the language tag that associates with a language command using patterns. The general structure of AIML objects is put forward by List of parameters the most important object among the AIML objects is category, pattern, and template. The task of the category tag is defining the various patterns and their answer-based templates. The pattern tag identifies the input

from the user and the task of template tag is to respond to the specific user input, these are the most frequent tags and the bases to design AIML Chatbots with an intelligent response to natural language speech conversations. Let's see the structure of category, pattern, and template object which is shown below:

```
<category>
    <pattern>User Input</pattern>
         <template>
              Corresponding Response to input
        </template>
</category>
```

SYSTEM DESIGN

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a given system to satisfy specified requirements. Systems design could be the application of various systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and system designing.

A chatbot is a computer program, which is designed to simulate a conversation with human users using patterns, especially over the internet. They are our online assistants that offer different services through chatting over the internet. To build artificial intelligence chatbots through Python, you will require an AIML package (Artificial Intelligence Markup Language). First, we need to create a standard start-up file without any pattern and load amil b in the kernel. Add random response patterns that would make dialogue interesting.

Now, to code your own AIML files, look for some files which are available beforehand. For example, browse all among files from the Alice Bot website. The start-up file we will be creating will act as a separate entity. As a result of which, we will have more AIML files without a source code modification. The program will start running when there are enough AIML files for loading. This was an introduction to how to make AI chatbot using Python. Now, let’s proceed further and see which particular library can be implemented for building an AI Chatbot.

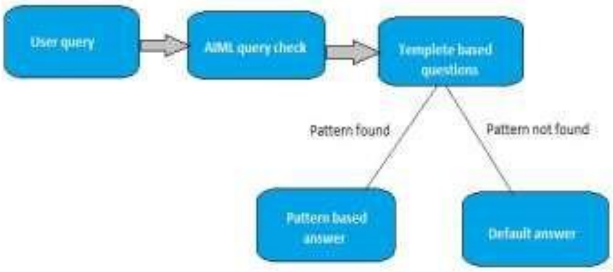


Fig: System Architecture

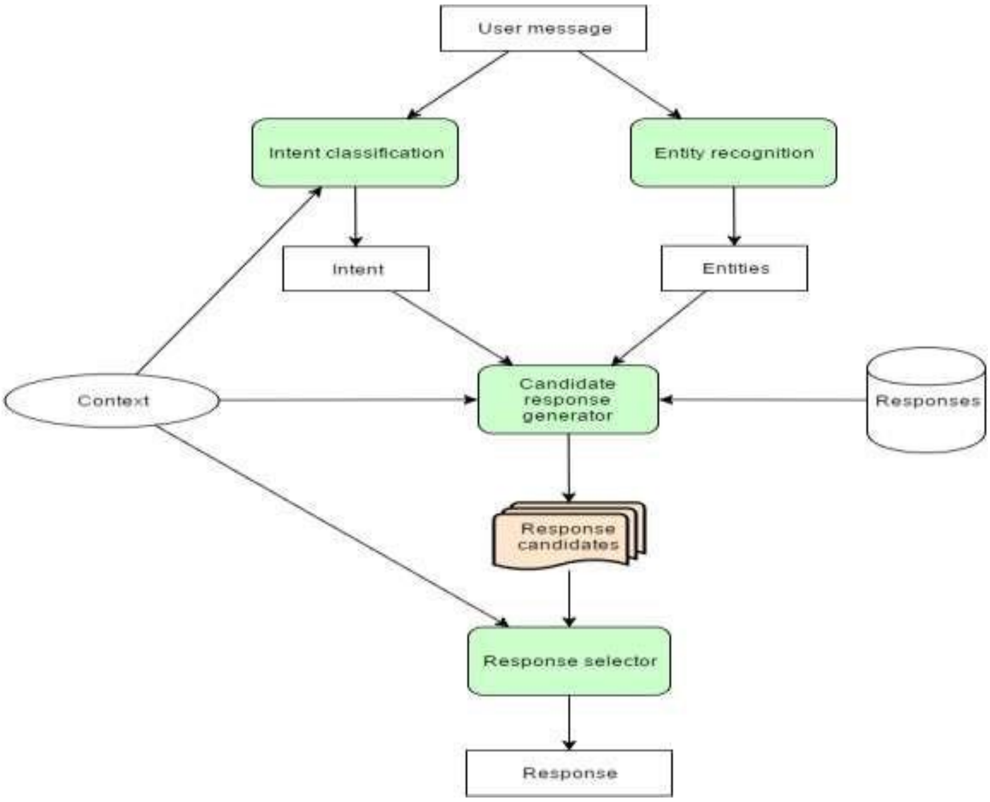


Fig: System Module

PROPOSED SYSTEM

In this work, we have developed an interactive chatbot using the Flask framework in python, and the workflow of the proposed framework is shown in the figure given below. User discussion, as a rule, begins with the simple welcome or general questions. User inquiries are first taken care of by AIML check, to check whether the entered inquiry is AIML script or not. AIML is characterized by general inquiries, queries, and welcome which is replied by utilizing AIML formats.

Once the bot-user types in the query in the chatbot, the AIML developed chatbot will identify the category that contains the query pattern. Here the bot-user is expected to type in the query in a predefined pattern. Once the query pattern is matched, the template of the category that contains the response is sent back to the bot-user.

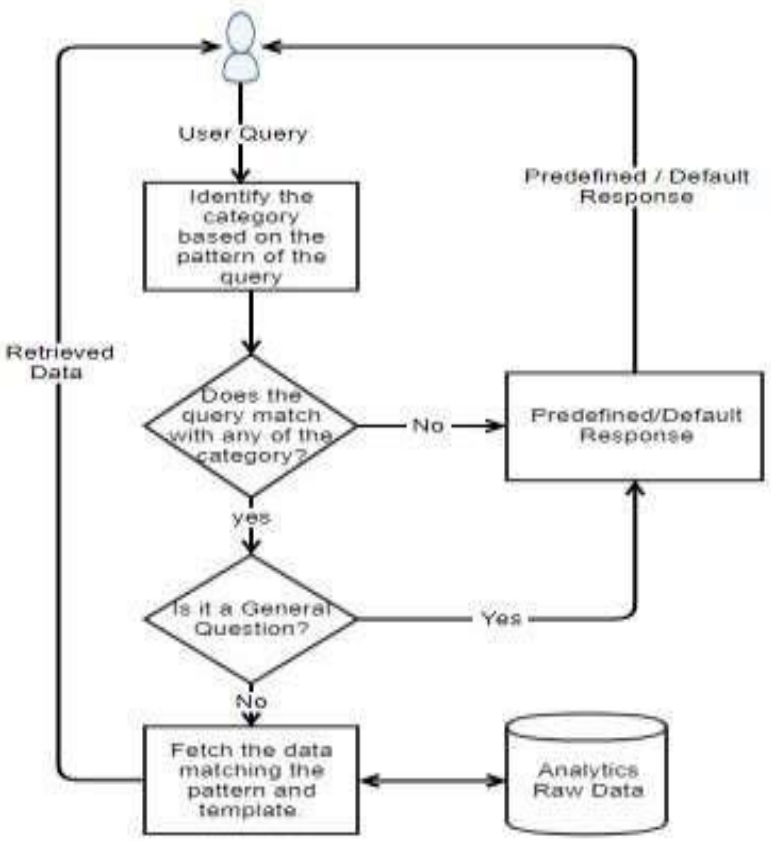

User Query
Identify the category based on the pattern of the query
Predefined / Default Response
Retrieved Data
Does the query match with any of the category?
No
Predefined/Default Response
yes
Is it a General Question?
Yes
No
Fetch the data matching the pattern and template
Analytics Raw Data

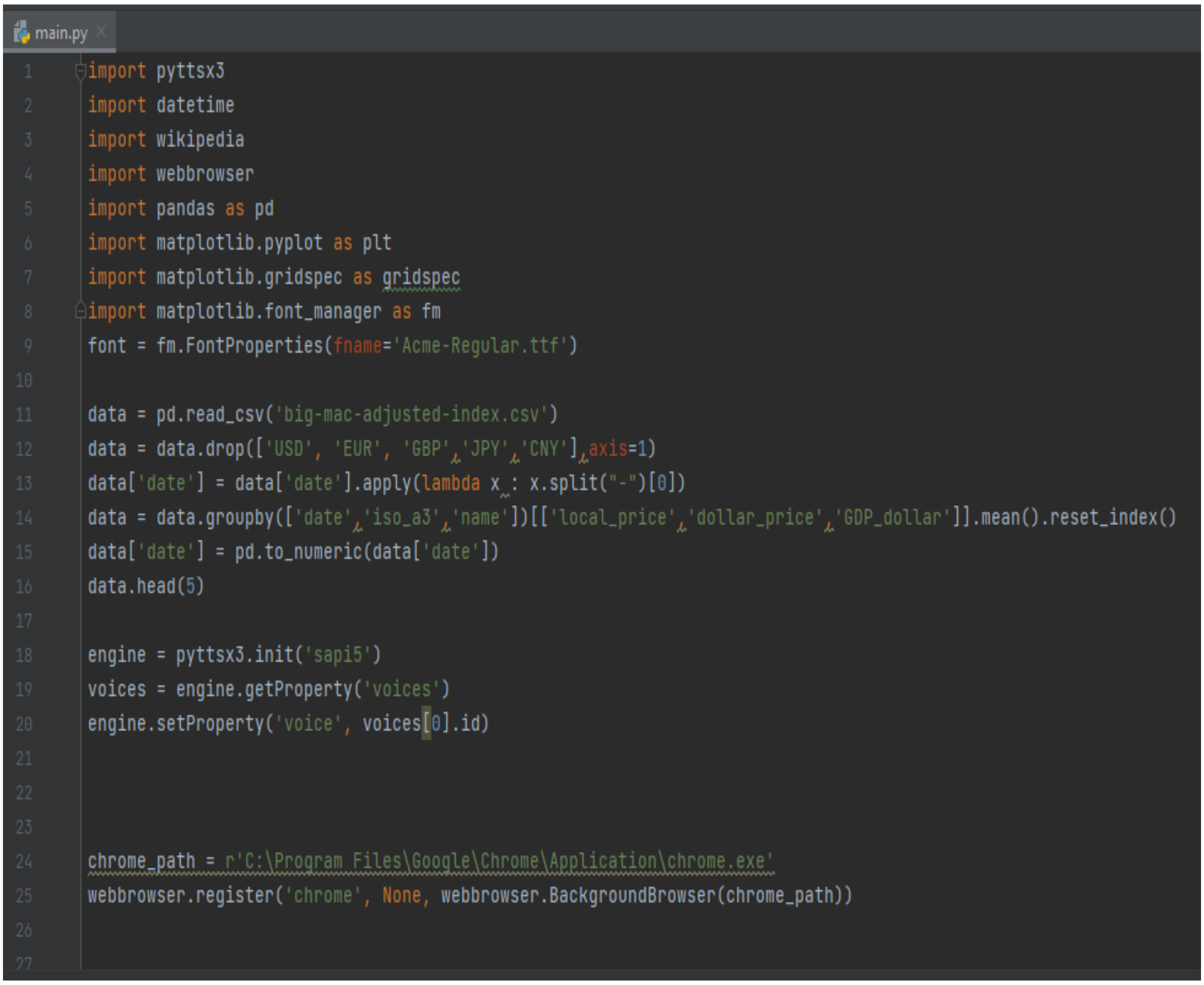
Fig: Proposed Model

IMPLEMENTATION

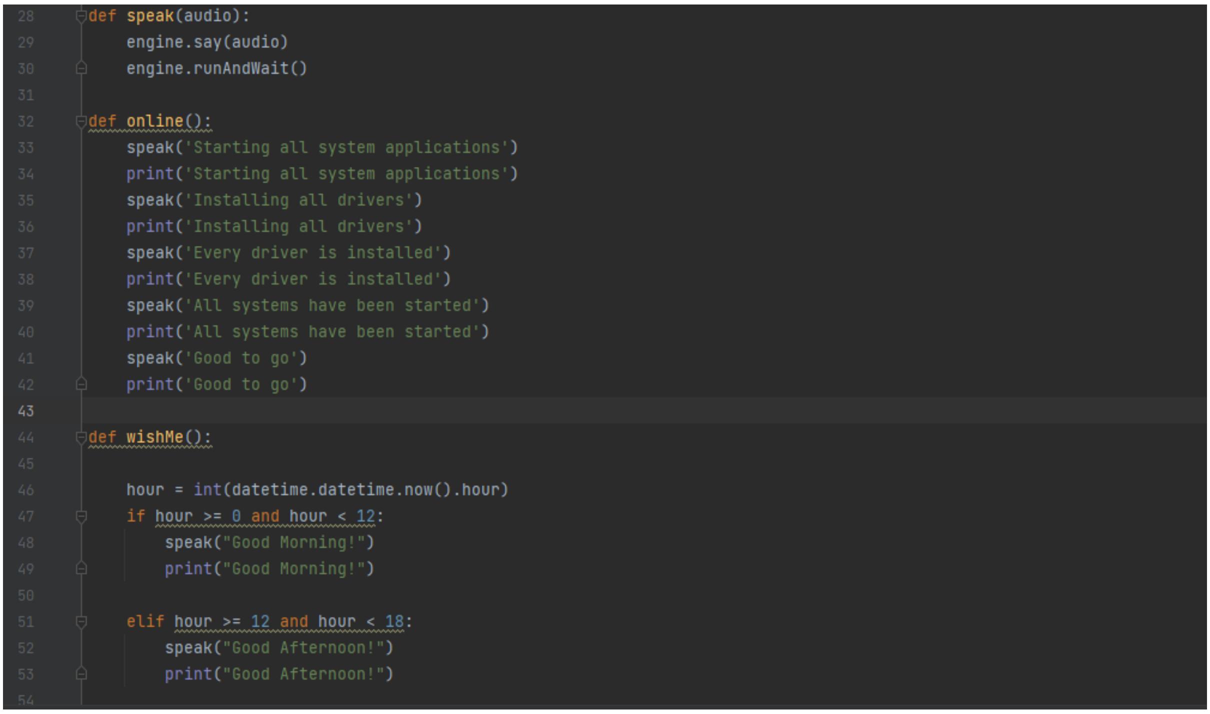
We collected the data set from Kaggle and studied it. After that we used switch case to encounter the different types of questions asked by the user. Using matplotlib we plotted the data set. The modules that we used were-

- Pyttsx3
- Datetime
- Wikipedia
- Pandas
- Matplotlin

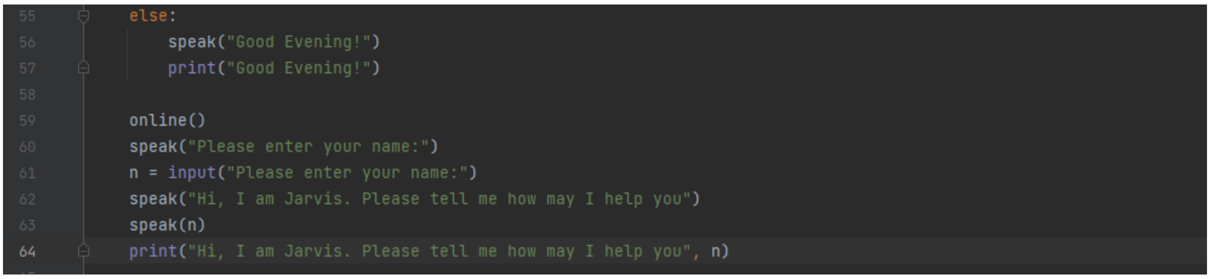
This section covers the design and implementation of the bot, which contains the design of the PYTHON module.



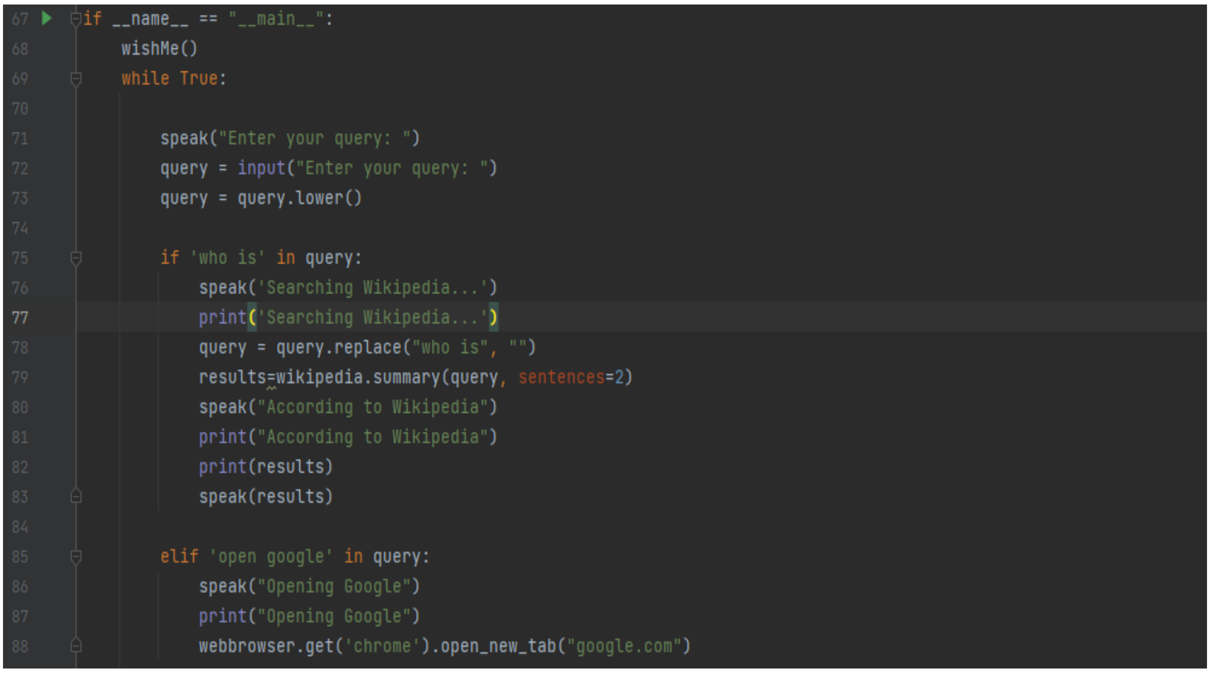
```python
import pyttsx3
import datetime
import wikipedia
import webbrowser
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import matplotlib.font_manager as fm
font = fm.FontProperties(fname='Acme-Regular.ttf')

data = pd.read_csv('big-mac-adjusted-index.csv')
data = data.drop(['USD', 'EUR', 'GBP','JPY','CNY'],axis=1)
data['date'] = data['date'].apply(lambda x : x.split("-")[0])
data = data.groupby(['date','iso_a3','name'])[['local_price','dollar_price','GDP_dollar']].mean().reset_index()
data['date'] = pd.to_numeric(data['date'])
data.head(5)

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)



chrome_path = r'C:\Program Files\Google\Chrome\Application\chrome.exe'
webbrowser.register('chrome', None, webbrowser.BackgroundBrowser(chrome_path))

```



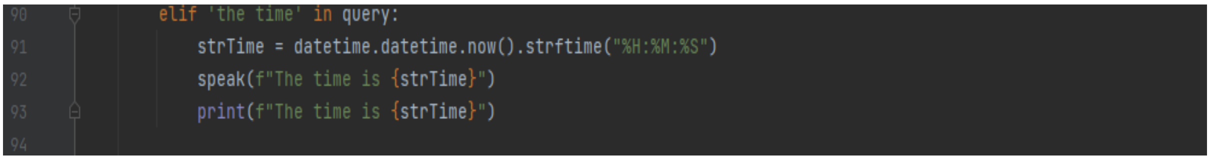
```python
def speak(audio):
    engine.say(audio)
    engine.runAndWait()

def online():
    speak('Starting all system applications')
    print('Starting all system applications')
    speak('Installing all drivers')
    print('Installing all drivers')
    speak('Every driver is installed')
    print('Every driver is installed')
    speak('All systems have been started')
    print('All systems have been started')
    speak('Good to go')
    print('Good to go')

def wishMe():

    hour = int(datetime.datetime.now().hour)
    if hour >= 0 and hour < 12:
        speak("Good Morning!")
        print("Good Morning!")

    elif hour >= 12 and hour < 18:
        speak("Good Afternoon!")
        print("Good Afternoon!")
```



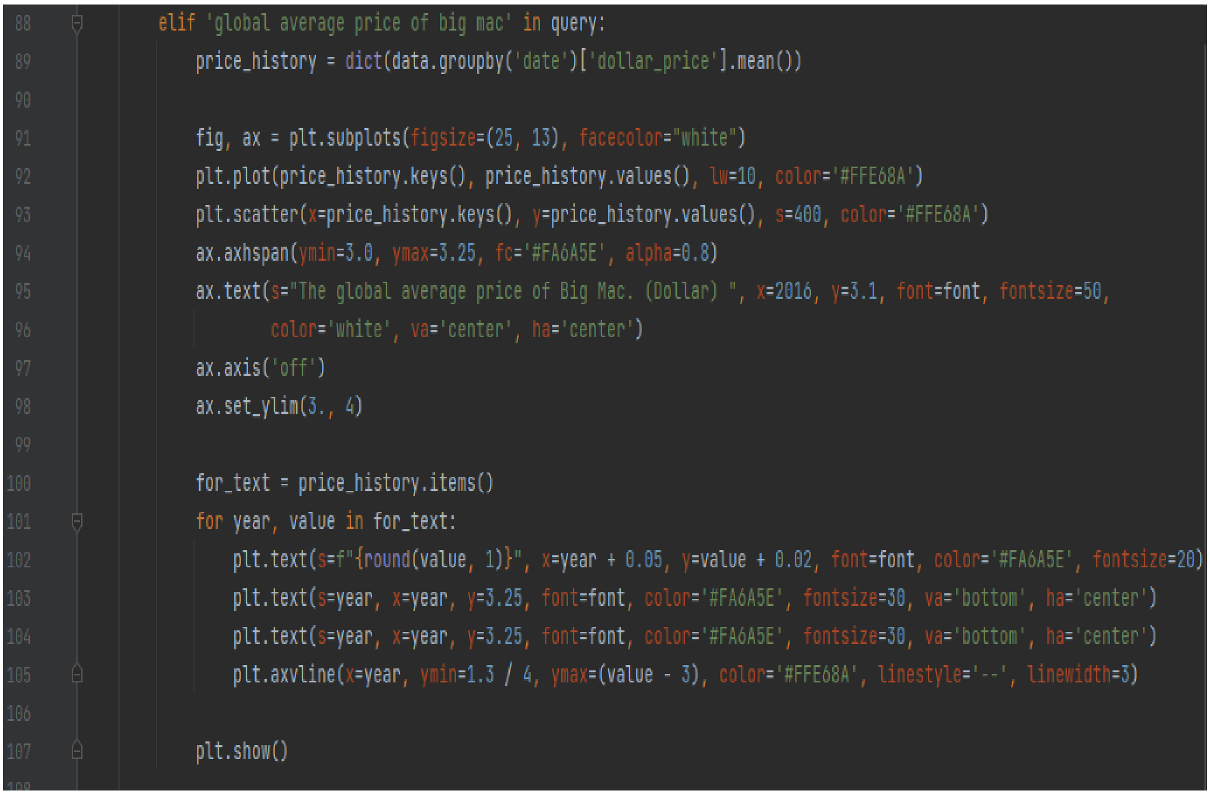
```python
    else:
        speak("Good Evening!")
        print("Good Evening!")

    online()
    speak("Please enter your name:")
    n = input("Please enter your name:")
    speak("Hi, I am Jarvis. Please tell me how may I help you")
    speak(n)
    print("Hi, I am Jarvis. Please tell me how may I help you", n)
```



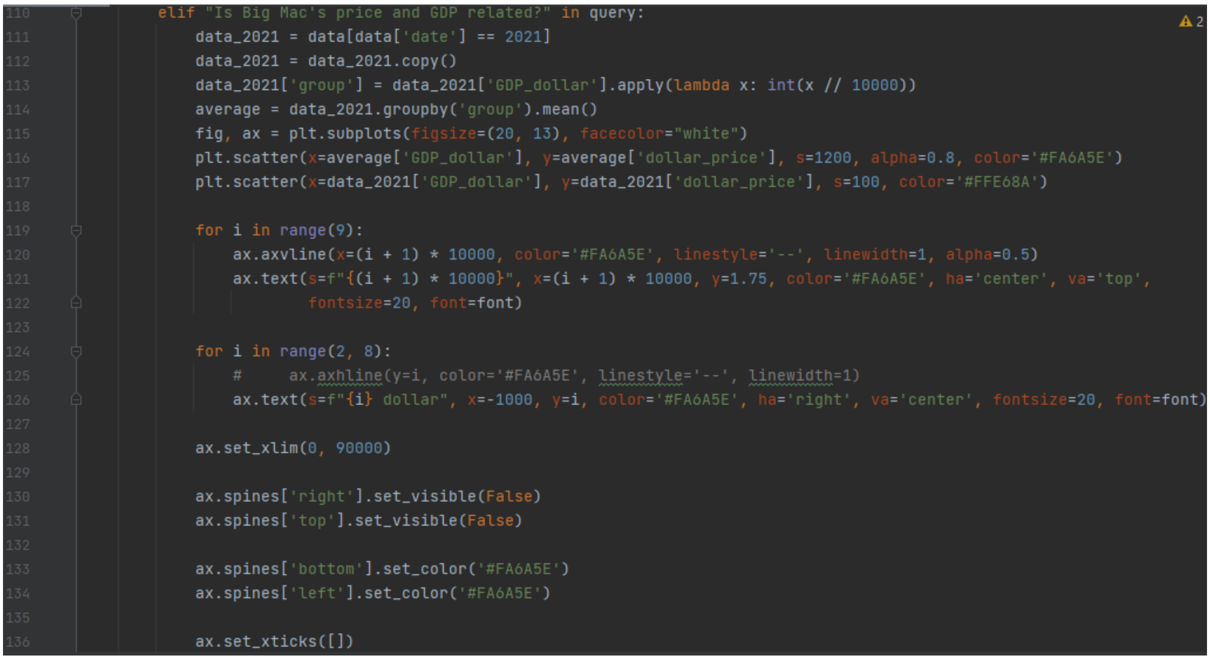
```python
if __name__ == "__main__":
    wishMe()
    while True:

        speak("Enter your query: ")
        query = input("Enter your query: ")
        query = query.lower()

        if 'who is' in query:
            speak('Searching Wikipedia...')
            print('Searching Wikipedia...')
            query = query.replace("who is", "")
            results=wikipedia.summary(query, sentences=2)
            speak("According to Wikipedia")
            print("According to Wikipedia")
            print(results)
            speak(results)

        elif 'open google' in query:
            speak("Opening Google")
            print("Opening Google")
            webbrowser.get('chrome').open_new_tab("google.com")
```



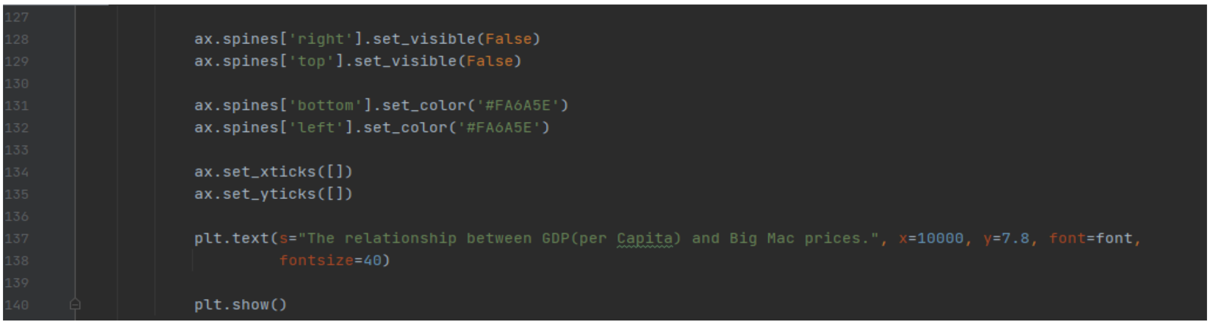
```python
    elif 'the time' in query:
        strTime = datetime.datetime.now().strftime("%H:%M:%S")
        speak(f"The time is {strTime}")
        print(f"The time is {strTime}")
```



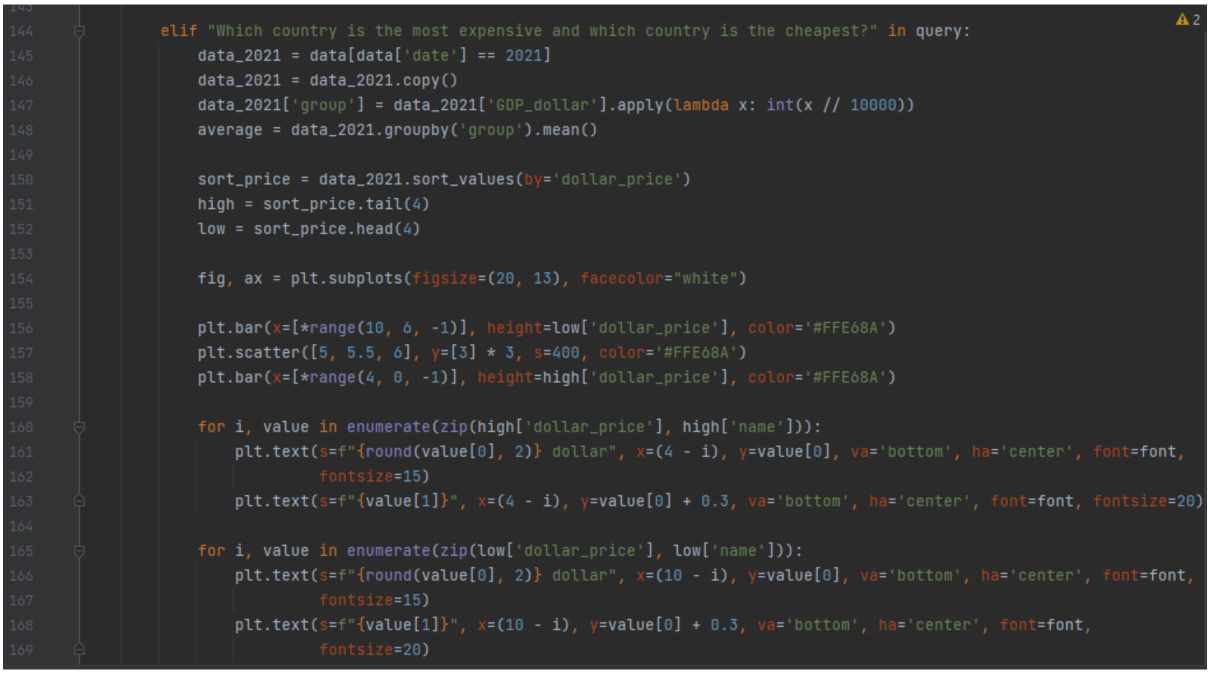
```python
    elif 'global average price of big mac' in query:
        price_history = dict(data.groupby('date')['dollar_price'].mean())

        fig, ax = plt.subplots(figsize=(25, 13), facecolor="white")
        plt.plot(price_history.keys(), price_history.values(), lw=10, color='#FFE68A')
        plt.scatter(x=price_history.keys(), y=price_history.values(), s=400, color='#FFE68A')
        ax.axhspan(ymin=3.0, ymax=3.25, fc='#FA6A5E', alpha=0.8)
        ax.text(s="The global average price of Big Mac. (Dollar) ", x=2016, y=3.1, font=font, fontsize=50,
                color='white', va='center', ha='center')
        ax.axis('off')
        ax.set_ylim(3., 4)

        for_text = price_history.items()
        for year, value in for_text:
            plt.text(s=f"{round(value, 1)}", x=year + 0.05, y=value + 0.02, font=font, color='#FA6A5E', fontsize=20)
            plt.text(s=year, x=year, y=3.25, font=font, color='#FA6A5E', fontsize=30, va='bottom', ha='center')
            plt.text(s=year, x=year, y=3.25, font=font, color='#FA6A5E', fontsize=30, va='bottom', ha='center')
            plt.axvline(x=year, ymin=1.3 / 4, ymax=(value - 3), color='#FFE68A', linestyle='--', linewidth=3)

        plt.show()
```



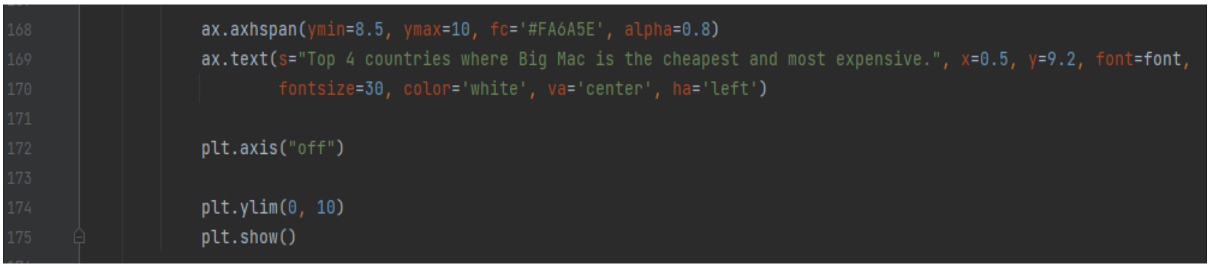
```python
    elif "Is Big Mac's price and GDP related?" in query:
        data_2021 = data[data['date'] == 2021]
        data_2021 = data_2021.copy()
        data_2021['group'] = data_2021['GDP_dollar'].apply(lambda x: int(x // 10000))
        average = data_2021.groupby('group').mean()
        fig, ax = plt.subplots(figsize=(20, 13), facecolor="white")
        plt.scatter(x=average['GDP_dollar'], y=average['dollar_price'], s=1200, alpha=0.8, color='#FA6A5E')
        plt.scatter(x=data_2021['GDP_dollar'], y=data_2021['dollar_price'], s=100, color='#FFE68A')

        for i in range(9):
            ax.axvline(x=(i + 1) * 10000, color='#FA6A5E', linestyle='--', linewidth=1, alpha=0.5)
            ax.text(s=f"{(i + 1) * 10000}", x=(i + 1) * 10000, y=1.75, color='#FA6A5E', ha='center', va='top',
                    fontsize=20, font=font)

        for i in range(2, 8):
            #     ax.axhline(y=i, color='#FA6A5E', linestyle='--', linewidth=1)
            ax.text(s=f"{i} dollar", x=-1000, y=i, color='#FA6A5E', ha='right', va='center', fontsize=20, font=font)

        ax.set_xlim(0, 90000)

        ax.spines['right'].set_visible(False)
        ax.spines['top'].set_visible(False)

        ax.spines['bottom'].set_color('#FA6A5E')
        ax.spines['left'].set_color('#FA6A5E')

        ax.set_xticks([])
```



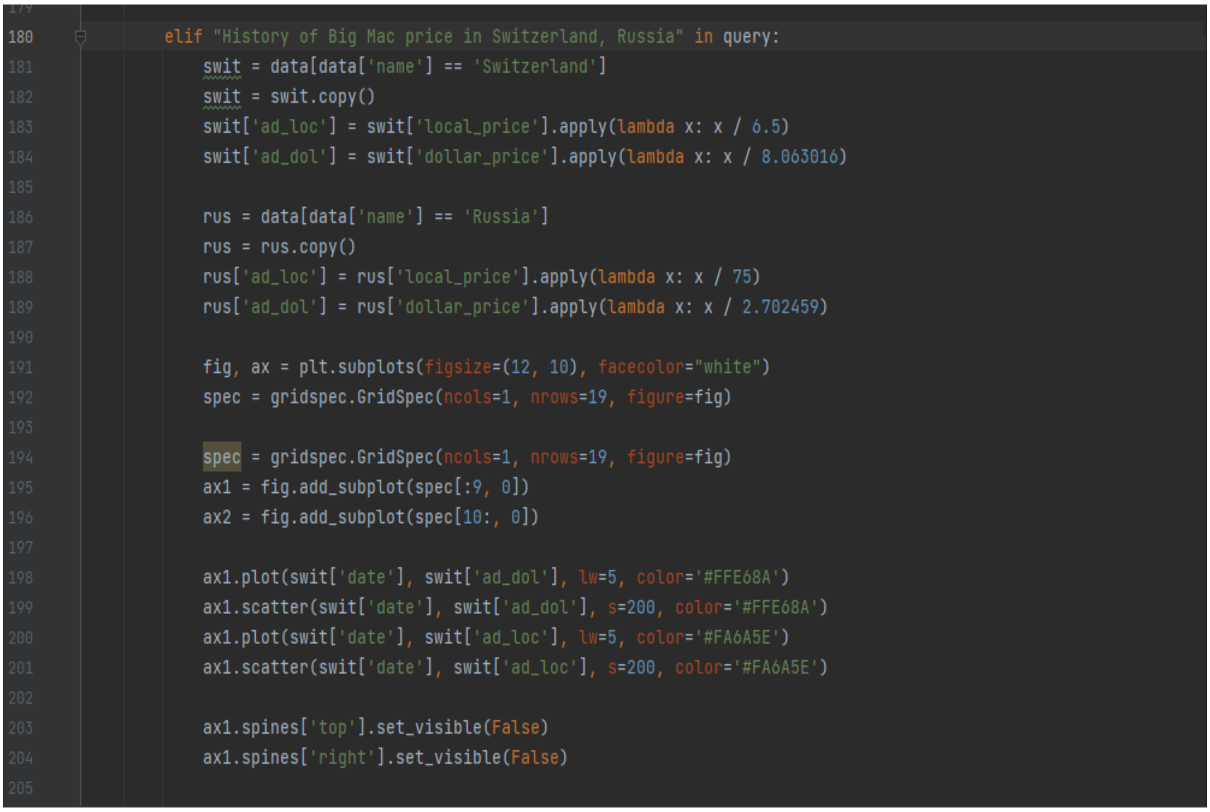
```
        ax.spines['right'].set_visible(False)
        ax.spines['top'].set_visible(False)

        ax.spines['bottom'].set_color('#FA6A5E')
        ax.spines['left'].set_color('#FA6A5E')

        ax.set_xticks([])
        ax.set_yticks([])

        plt.text(s="The relationship between GDP(per Capita) and Big Mac prices.", x=10000, y=7.8, font=font,
                 fontsize=40)

        plt.show()
```



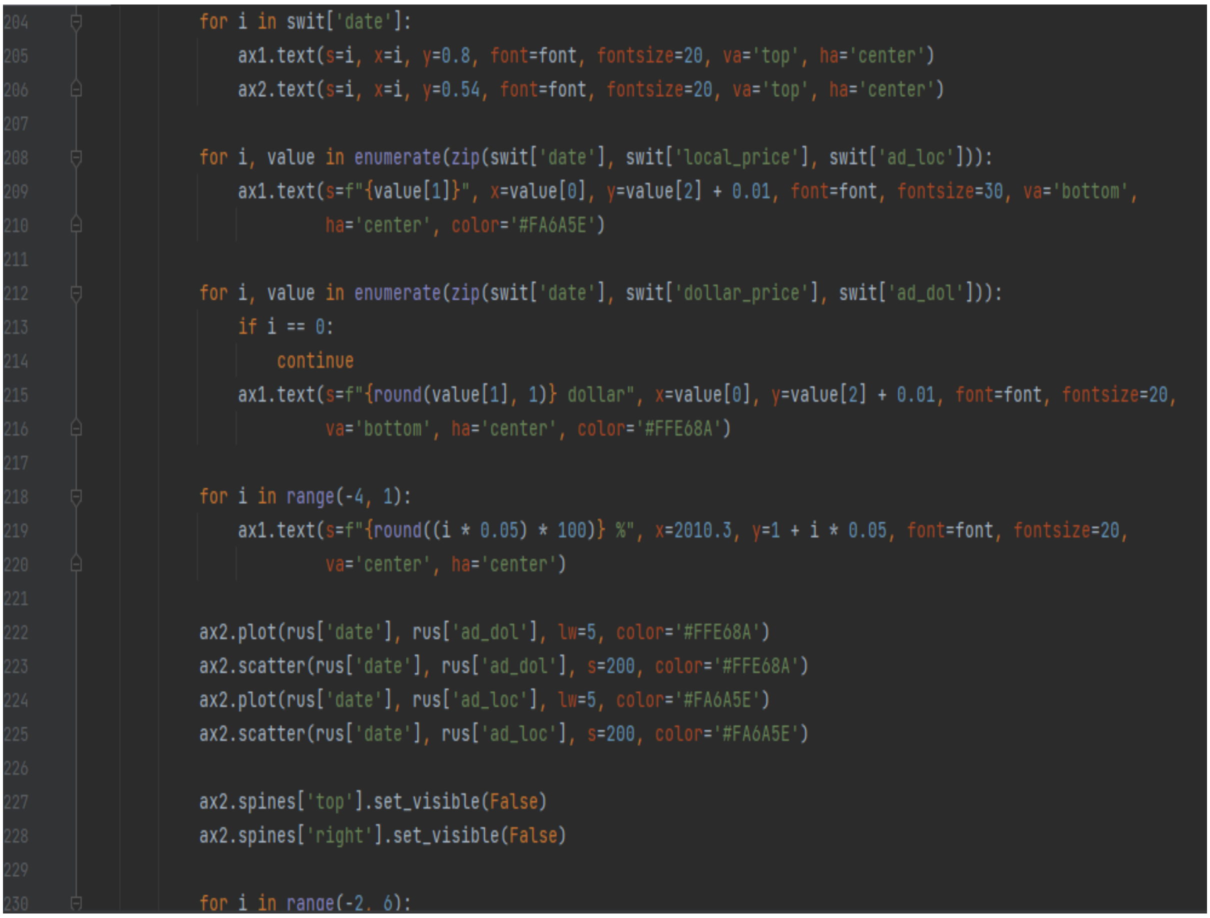
```
    elif "Which country is the most expensive and which country is the cheapest?" in query:
        data_2021 = data[data['date'] == 2021]
        data_2021 = data_2021.copy()
        data_2021['group'] = data_2021['GDP_dollar'].apply(lambda x: int(x // 10000))
        average = data_2021.groupby('group').mean()

        sort_price = data_2021.sort_values(by='dollar_price')
        high = sort_price.tail(4)
        low = sort_price.head(4)

        fig, ax = plt.subplots(figsize=(20, 13), facecolor="white")

        plt.bar(x=[*range(10, 6, -1)], height=low['dollar_price'], color='#FFE68A')
        plt.scatter([5, 5.5, 6], y=[3] * 3, s=400, color='#FFE68A')
        plt.bar(x=[*range(4, 0, -1)], height=high['dollar_price'], color='#FFE68A')

        for i, value in enumerate(zip(high['dollar_price'], high['name'])):
            plt.text(s=f"{round(value[0], 2)} dollar", x=(4 - i), y=value[0], va='bottom', ha='center', font=font,
                     fontsize=15)
            plt.text(s=f"{value[1]}", x=(4 - i), y=value[0] + 0.3, va='bottom', ha='center', font=font, fontsize=20)

        for i, value in enumerate(zip(low['dollar_price'], low['name'])):
            plt.text(s=f"{round(value[0], 2)} dollar", x=(10 - i), y=value[0], va='bottom', ha='center', font=font,
                     fontsize=15)
            plt.text(s=f"{value[1]}", x=(10 - i), y=value[0] + 0.3, va='bottom', ha='center', font=font,
                     fontsize=20)
```



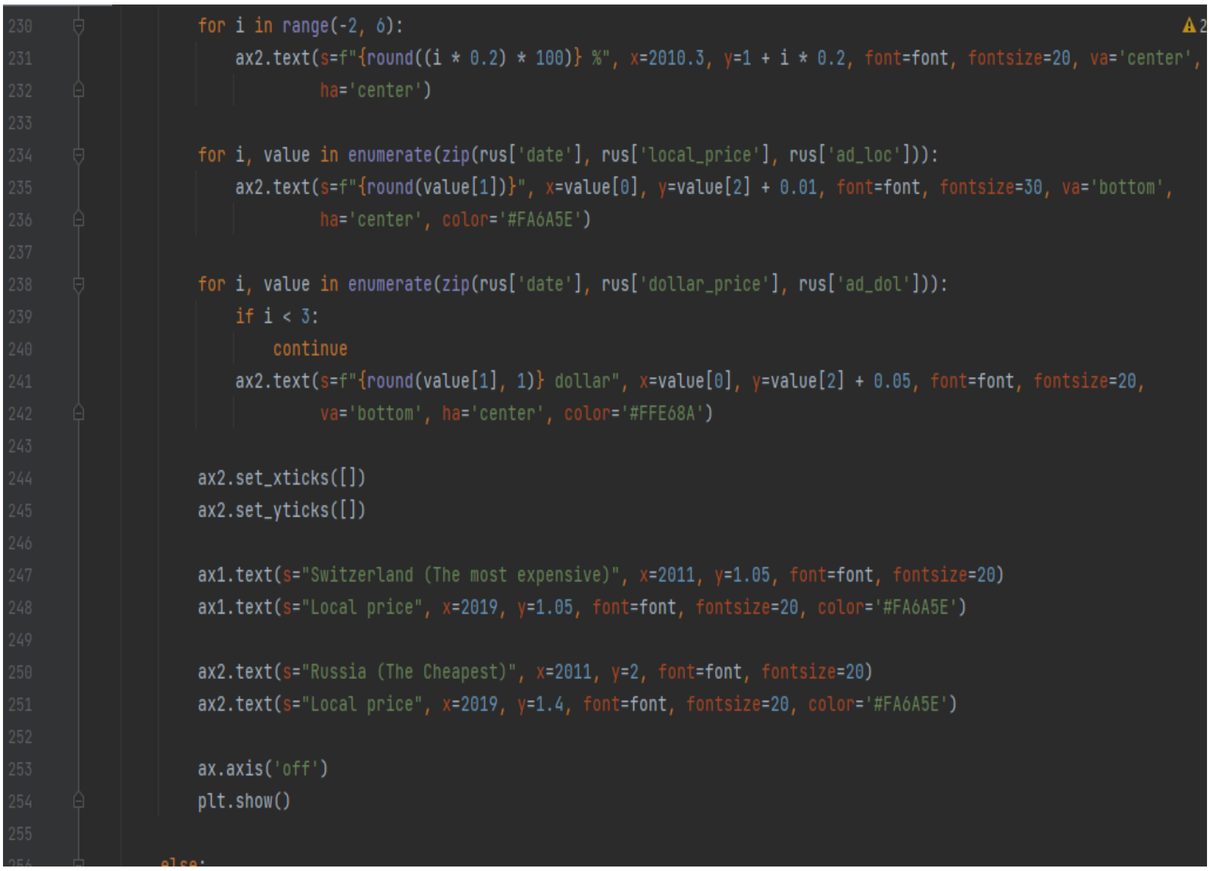
```
        ax.axhspan(ymin=8.5, ymax=10, fc='#FA6A5E', alpha=0.8)
        ax.text(s="Top 4 countries where Big Mac is the cheapest and most expensive.", x=0.5, y=9.2, font=font,
                fontsize=30, color='white', va='center', ha='left')

        plt.axis("off")

        plt.ylim(0, 10)
        plt.show()
```



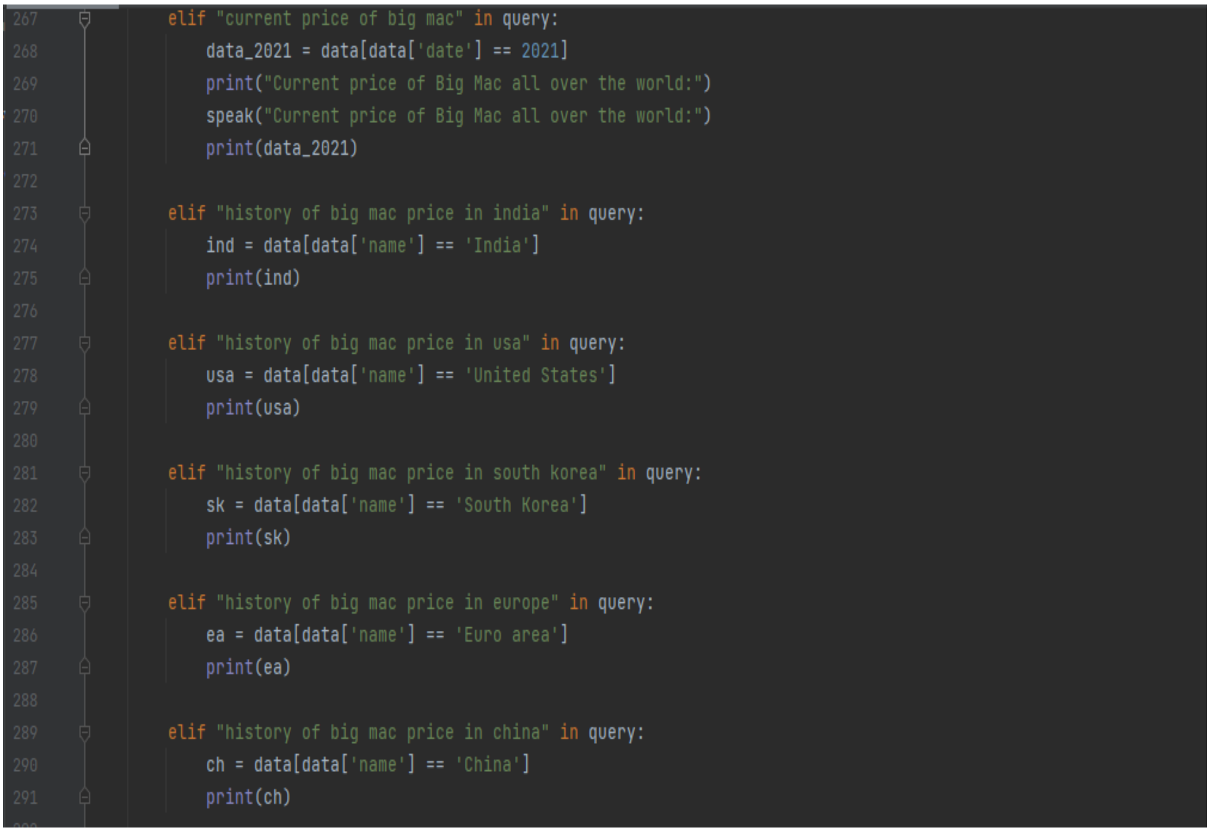
```
    elif "History of Big Mac price in Switzerland, Russia" in query:
        swit = data[data['name'] == 'Switzerland']
        swit = swit.copy()
        swit['ad_loc'] = swit['local_price'].apply(lambda x: x / 6.5)
        swit['ad_dol'] = swit['dollar_price'].apply(lambda x: x / 8.063016)

        rus = data[data['name'] == 'Russia']
        rus = rus.copy()
        rus['ad_loc'] = rus['local_price'].apply(lambda x: x / 75)
        rus['ad_dol'] = rus['dollar_price'].apply(lambda x: x / 2.702459)

        fig, ax = plt.subplots(figsize=(12, 10), facecolor="white")
        spec = gridspec.GridSpec(ncols=1, nrows=19, figure=fig)

        spec = gridspec.GridSpec(ncols=1, nrows=19, figure=fig)
        ax1 = fig.add_subplot(spec[:9, 0])
        ax2 = fig.add_subplot(spec[10:, 0])

        ax1.plot(swit['date'], swit['ad_dol'], lw=5, color='#FFE68A')
        ax1.scatter(swit['date'], swit['ad_dol'], s=200, color='#FFE68A')
        ax1.plot(swit['date'], swit['ad_loc'], lw=5, color='#FA6A5E')
        ax1.scatter(swit['date'], swit['ad_loc'], s=200, color='#FA6A5E')

        ax1.spines['top'].set_visible(False)
        ax1.spines['right'].set_visible(False)

```



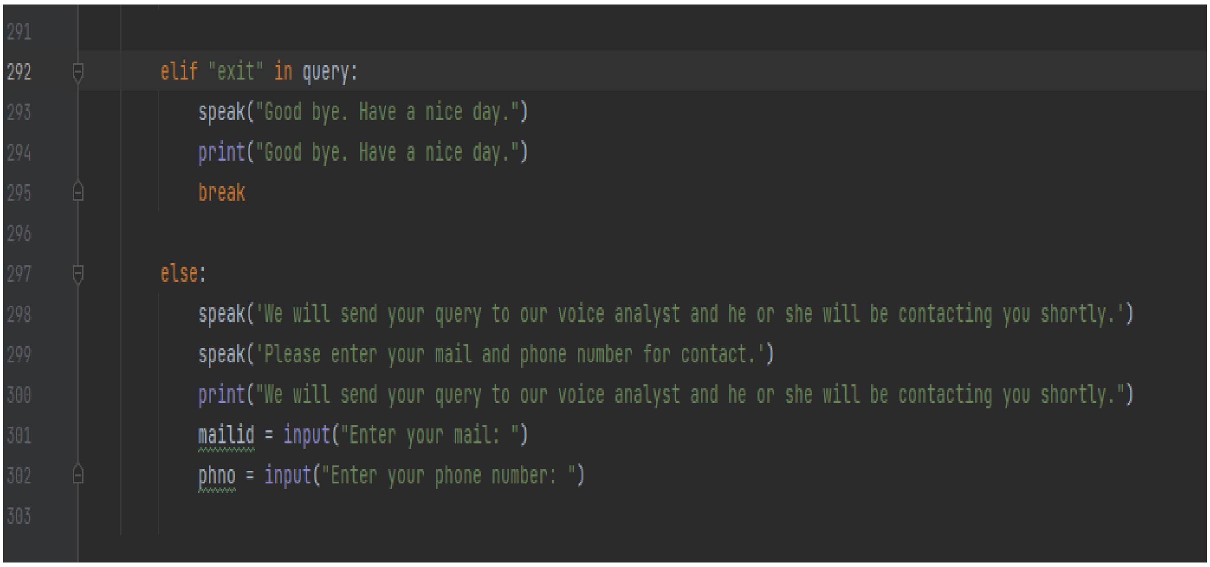
```
        for i in swit['date']:
            ax1.text(s=i, x=i, y=0.8, font=font, fontsize=20, va='top', ha='center')
            ax2.text(s=i, x=i, y=0.54, font=font, fontsize=20, va='top', ha='center')

        for i, value in enumerate(zip(swit['date'], swit['local_price'], swit['ad_loc'])):
            ax1.text(s=f"{value[1]}", x=value[0], y=value[2] + 0.01, font=font, fontsize=30, va='bottom',
                     ha='center', color='#FA6A5E')

        for i, value in enumerate(zip(swit['date'], swit['dollar_price'], swit['ad_dol'])):
            if i == 0:
                continue
            ax1.text(s=f"{round(value[1], 1)} dollar", x=value[0], y=value[2] + 0.01, font=font, fontsize=20,
                     va='bottom', ha='center', color='#FFE68A')

        for i in range(-4, 1):
            ax1.text(s=f"{round((i * 0.05) * 100)} %", x=2010.3, y=1 + i * 0.05, font=font, fontsize=20,
                     va='center', ha='center')

        ax2.plot(rus['date'], rus['ad_dol'], lw=5, color='#FFE68A')
        ax2.scatter(rus['date'], rus['ad_dol'], s=200, color='#FFE68A')
        ax2.plot(rus['date'], rus['ad_loc'], lw=5, color='#FA6A5E')
        ax2.scatter(rus['date'], rus['ad_loc'], s=200, color='#FA6A5E')

        ax2.spines['top'].set_visible(False)
        ax2.spines['right'].set_visible(False)

        for i in range(-2, 6):
```



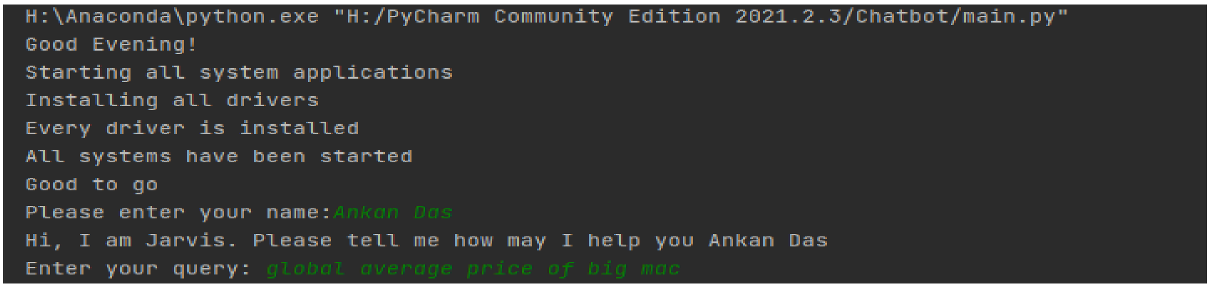
```
        for i in range(-2, 6):
            ax2.text(s=f"{round((i * 0.2) * 100)} %", x=2010.3, y=1 + i * 0.2, font=font, fontsize=20, va='center',
                     ha='center')

        for i, value in enumerate(zip(rus['date'], rus['local_price'], rus['ad_loc'])):
            ax2.text(s=f"{round(value[1])}", x=value[0], y=value[2] + 0.01, font=font, fontsize=30, va='bottom',
                     ha='center', color='#FA6A5E')

        for i, value in enumerate(zip(rus['date'], rus['dollar_price'], rus['ad_dol'])):
            if i < 3:
                continue
            ax2.text(s=f"{round(value[1], 1)} dollar", x=value[0], y=value[2] + 0.05, font=font, fontsize=20,
                     va='bottom', ha='center', color='#FFE68A')

        ax2.set_xticks([])
        ax2.set_yticks([])

        ax1.text(s="Switzerland (The most expensive)", x=2011, y=1.05, font=font, fontsize=20)
        ax1.text(s="Local price", x=2019, y=1.05, font=font, fontsize=20, color='#FA6A5E')

        ax2.text(s="Russia (The Cheapest)", x=2011, y=2, font=font, fontsize=20)
        ax2.text(s="Local price", x=2019, y=1.4, font=font, fontsize=20, color='#FA6A5E')

        ax.axis('off')
        plt.show()

    else:
```



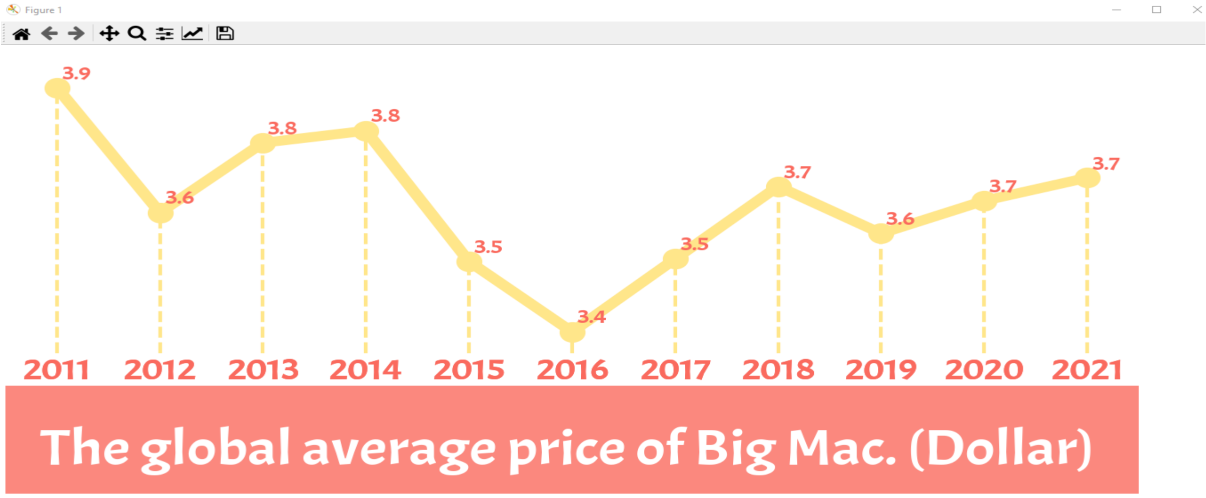
```
    elif "current price of big mac" in query:
        data_2021 = data[data['date'] == 2021]
        print("Current price of Big Mac all over the world:")
        speak("Current price of Big Mac all over the world:")
        print(data_2021)

    elif "history of big mac price in india" in query:
        ind = data[data['name'] == 'India']
        print(ind)

    elif "history of big mac price in usa" in query:
        usa = data[data['name'] == 'United States']
        print(usa)

    elif "history of big mac price in south korea" in query:
        sk = data[data['name'] == 'South Korea']
        print(sk)

    elif "history of big mac price in europe" in query:
        ea = data[data['name'] == 'Euro area']
        print(ea)

    elif "history of big mac price in china" in query:
        ch = data[data['name'] == 'China']
        print(ch)
```

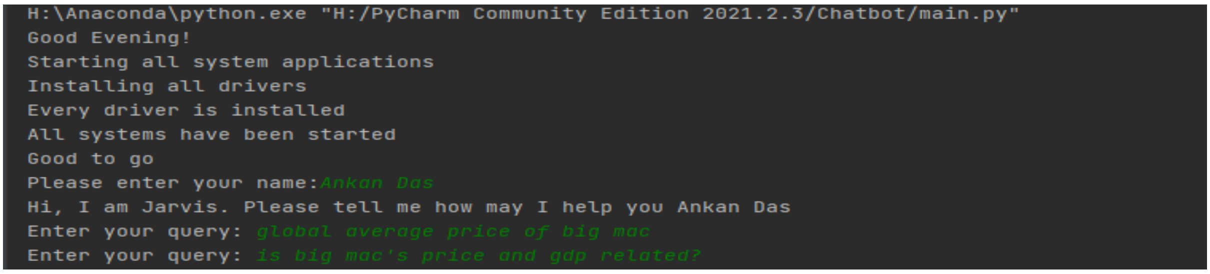


```
        elif "exit" in query:
            speak("Good bye. Have a nice day.")
            print("Good bye. Have a nice day.")
            break

        else:
            speak('We will send your query to our voice analyst and he or she will be contacting you shortly.')
            speak('Please enter your mail and phone number for contact.')
            print("We will send your query to our voice analyst and he or she will be contacting you shortly.")
            mailid = input("Enter your mail: ")
            phno = input("Enter your phone number: ")

```

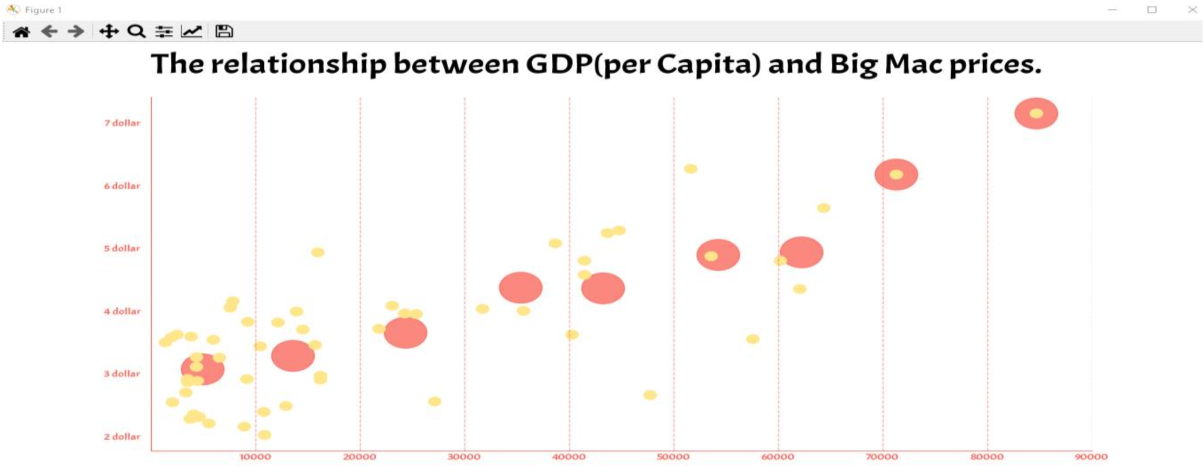


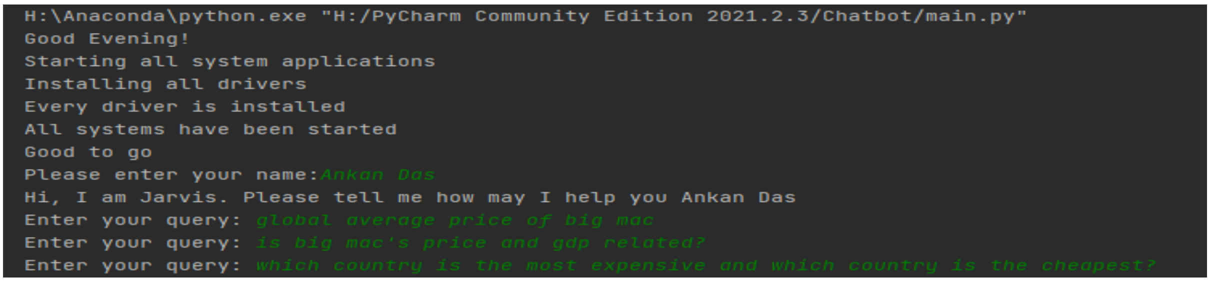
```
H:\Anaconda\python.exe "H:/PyCharm Community Edition 2021.2.3/Chatbot/main.py"
Good Evening!
Starting all system applications
Installing all drivers
Every driver is installed
All systems have been started
Good to go
Please enter your name:Ankan Das
Hi, I am Jarvis. Please tell me how may I help you Ankan Das
Enter your query: global average price of big mac
```



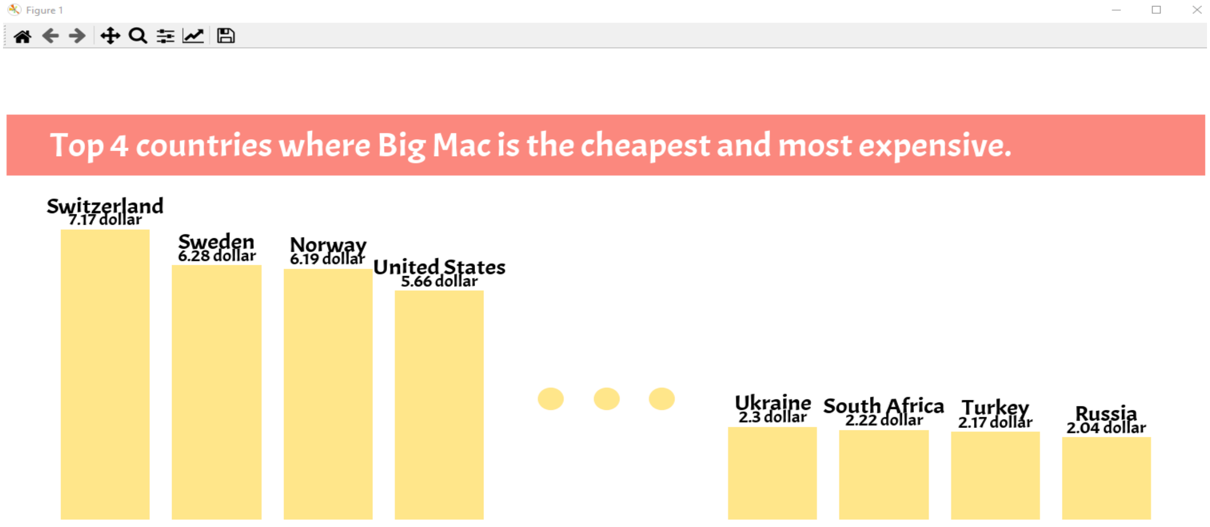


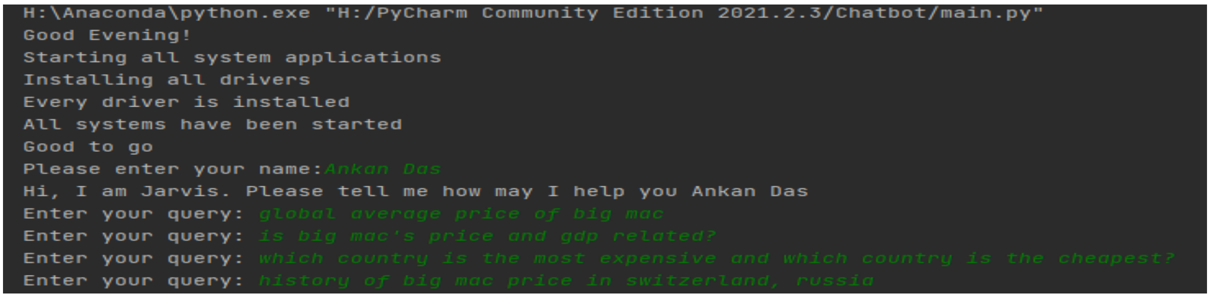
```
H:\Anaconda\python.exe "H:/PyCharm Community Edition 2021.2.3/Chatbot/main.py"
Good Evening!
Starting all system applications
Installing all drivers
Every driver is installed
All systems have been started
Good to go
Please enter your name:Ankan Das
Hi, I am Jarvis. Please tell me how may I help you Ankan Das
Enter your query: global average price of big mac
Enter your query: is big mac's price and gdp related?
```



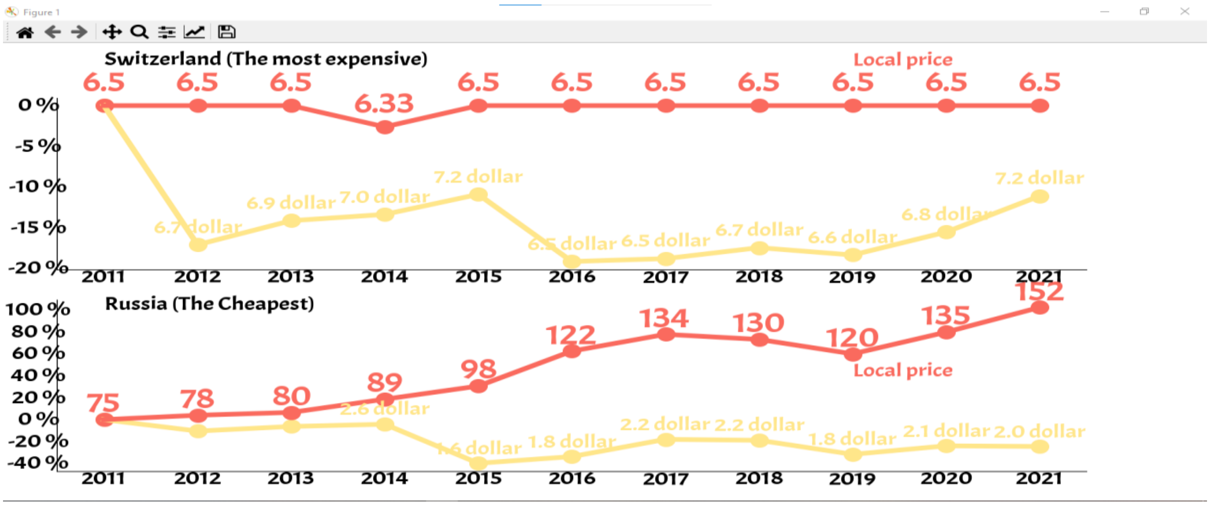


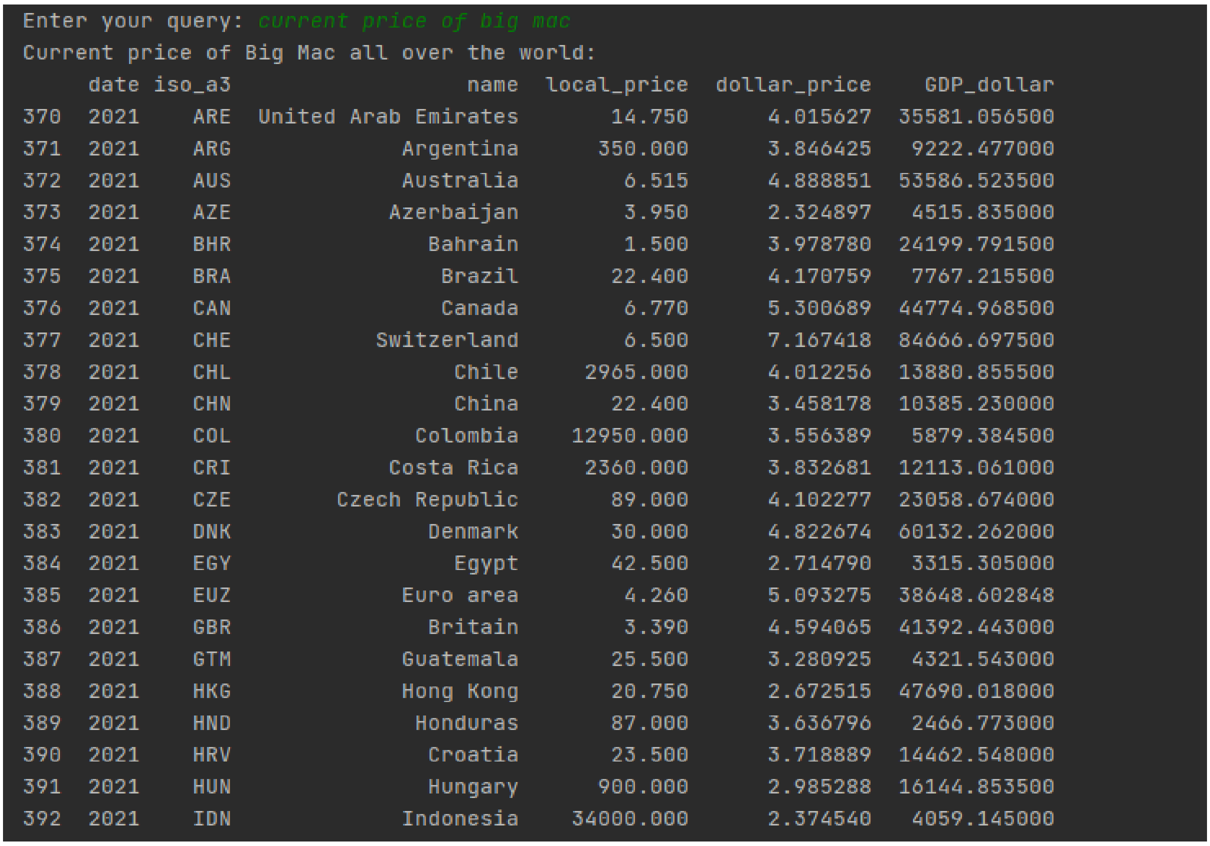
```
H:\Anaconda\python.exe "H:/PyCharm Community Edition 2021.2.3/Chatbot/main.py"
Good Evening!
Starting all system applications
Installing all drivers
Every driver is installed
All systems have been started
Good to go
Please enter your name:Ankan Das
Hi, I am Jarvis. Please tell me how may I help you Ankan Das
Enter your query: global average price of big mac
Enter your query: is big mac's price and gdp related?
Enter your query: which country is the most expensive and which country is the cheapest?
```

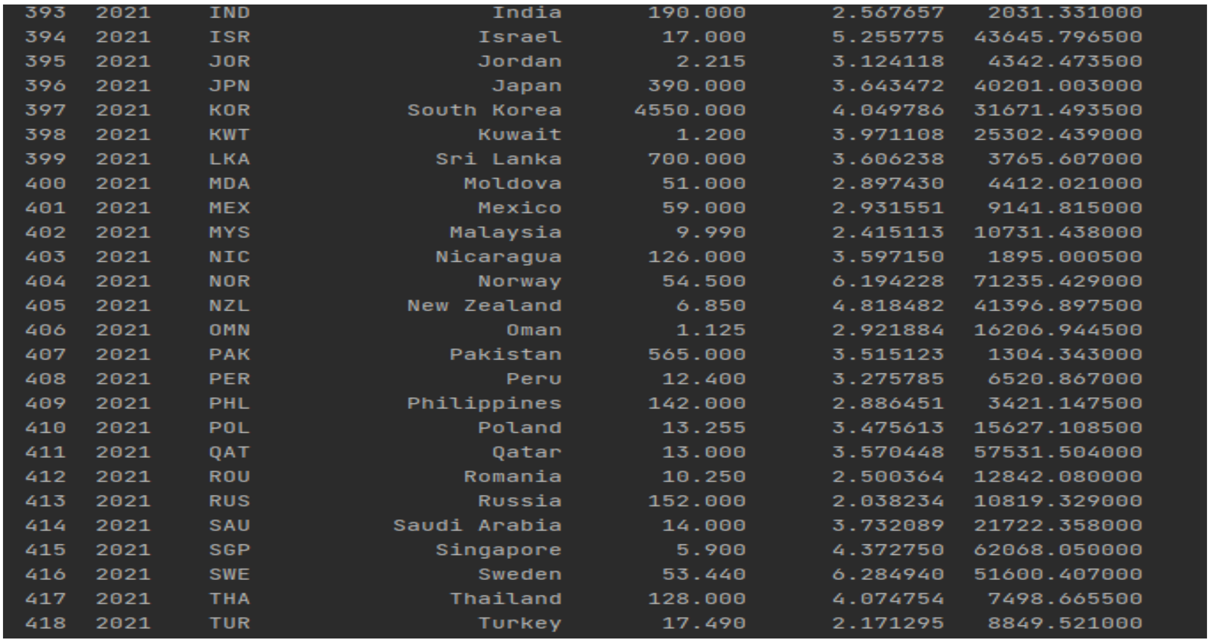




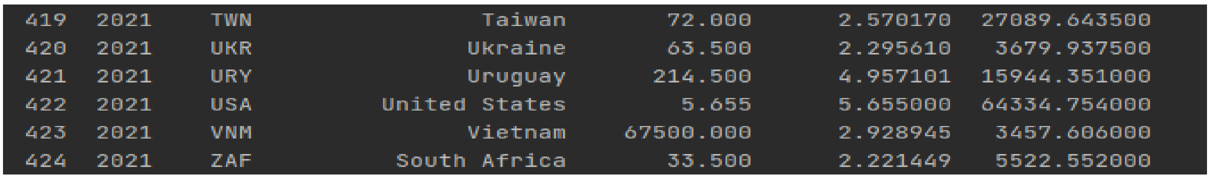
```
H:\Anaconda\python.exe "H:/PyCharm Community Edition 2021.2.3/Chatbot/main.py"
Good Evening!
Starting all system applications
Installing all drivers
Every driver is installed
All systems have been started
Good to go
Please enter your name:Ankan Das
Hi, I am Jarvis. Please tell me how may I help you Ankan Das
Enter your query: global average price of big mac
Enter your query: is big mac's price and gdp related?
Enter your query: which country is the most expensive and which country is the cheapest?
Enter your query: history of big mac price in switzerland, russia
```



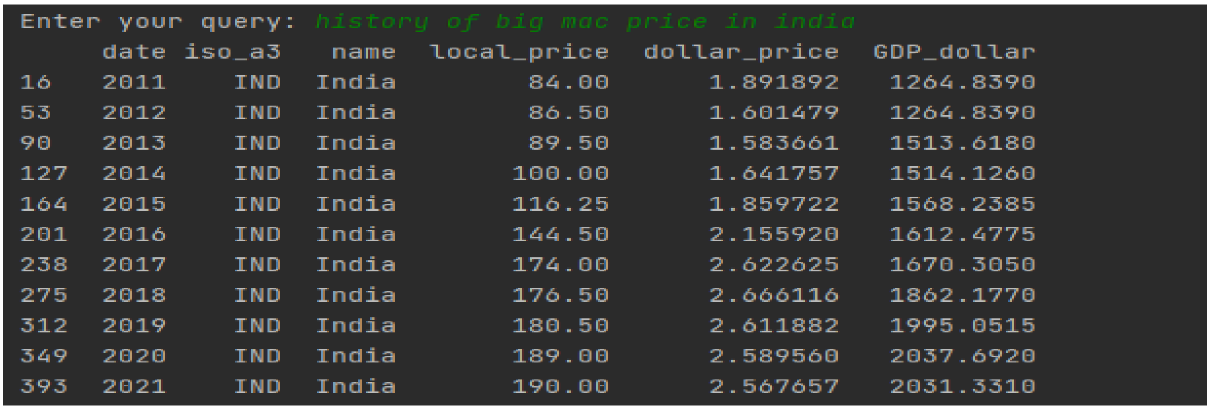

```
Enter your query: current price of big mac
Current price of Big Mac all over the world:
     date iso_a3                  name  local_price  dollar_price     GDP_dollar
370  2021    ARE  United Arab Emirates       14.750      4.015627   35581.056500
371  2021    ARG             Argentina      350.000      3.846425    9222.477000
372  2021    AUS             Australia        6.515      4.888851   53586.523500
373  2021    AZE            Azerbaijan        3.950      2.324897    4515.835000
374  2021    BHR               Bahrain        1.500      3.978780   24199.791500
375  2021    BRA                Brazil       22.400      4.170759    7767.215500
376  2021    CAN                Canada        6.770      5.300689   44774.968500
377  2021    CHE           Switzerland        6.500      7.167418   84666.697500
378  2021    CHL                 Chile     2965.000      4.012256   13880.855500
379  2021    CHN                 China       22.400      3.458178   10385.230000
380  2021    COL              Colombia    12950.000      3.556389    5879.384500
381  2021    CRI            Costa Rica     2360.000      3.832681   12113.061000
382  2021    CZE        Czech Republic       89.000      4.102277   23058.674000
383  2021    DNK               Denmark       30.000      4.822674   60132.262000
384  2021    EGY                 Egypt       42.500      2.714790    3315.305000
385  2021    EUZ             Euro area        4.260      5.093275   38648.602848
386  2021    GBR               Britain        3.390      4.594065   41392.443000
387  2021    GTM             Guatemala       25.500      3.280925    4321.543000
388  2021    HKG             Hong Kong       20.750      2.672515   47690.018000
389  2021    HND              Honduras       87.000      3.636796    2466.773000
390  2021    HRV               Croatia       23.500      3.718889   14462.548000
391  2021    HUN               Hungary      900.000      2.985288   16144.853500
392  2021    IDN             Indonesia    34000.000      2.374540    4059.145000
```



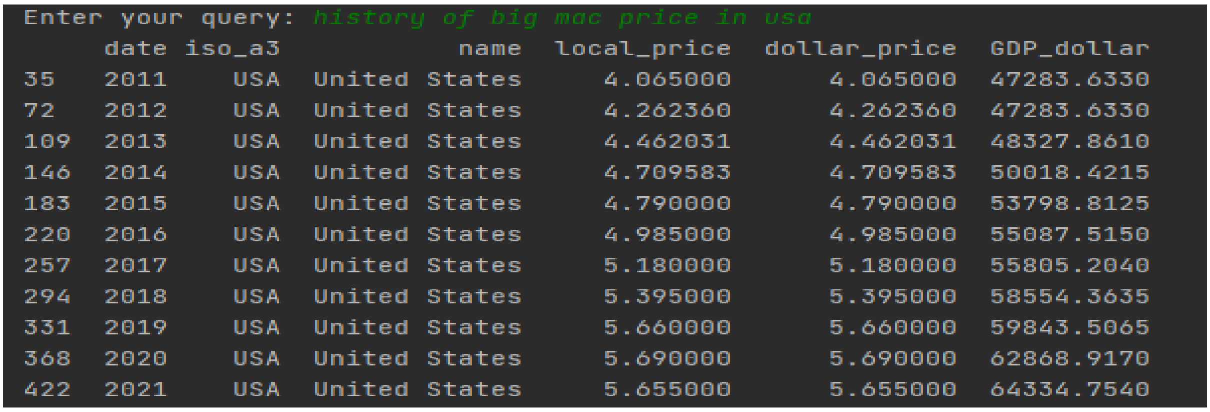
```
393  2021    IND              India     190.000      2.567657   2031.331000
394  2021    ISR             Israel      17.000      5.255775  43645.796500
395  2021    JOR             Jordan       2.215      3.124118   4342.473500
396  2021    JPN              Japan     390.000      3.643472  40201.003000
397  2021    KOR        South Korea    4550.000      4.049786  31671.493500
398  2021    KWT             Kuwait       1.200      3.971108  25302.439000
399  2021    LKA          Sri Lanka     700.000      3.606238   3765.607000
400  2021    MDA            Moldova      51.000      2.897430   4412.021000
401  2021    MEX             Mexico      59.000      2.931551   9141.815000
402  2021    MYS           Malaysia       9.990      2.415113  10731.438000
403  2021    NIC          Nicaragua     126.000      3.597150   1895.000500
404  2021    NOR             Norway      54.500      6.194228  71235.429000
405  2021    NZL        New Zealand       6.850      4.818482  41396.897500
406  2021    OMN               Oman       1.125      2.921884  16206.944500
407  2021    PAK           Pakistan     565.000      3.515123   1304.343000
408  2021    PER               Peru      12.400      3.275785   6520.867000
409  2021    PHL        Philippines     142.000      2.886451   3421.147500
410  2021    POL             Poland      13.255      3.475613  15627.108500
411  2021    QAT              Qatar      13.000      3.570448  57531.504000
412  2021    ROU            Romania      10.250      2.500364  12842.080000
413  2021    RUS             Russia     152.000      2.038234  10819.329000
414  2021    SAU       Saudi Arabia      14.000      3.732089  21722.358000
415  2021    SGP          Singapore       5.900      4.372750  62068.050000
416  2021    SWE             Sweden      53.440      6.284940  51600.407000
417  2021    THA           Thailand     128.000      4.074754   7498.665500
418  2021    TUR             Turkey      17.490      2.171295   8849.521000
```



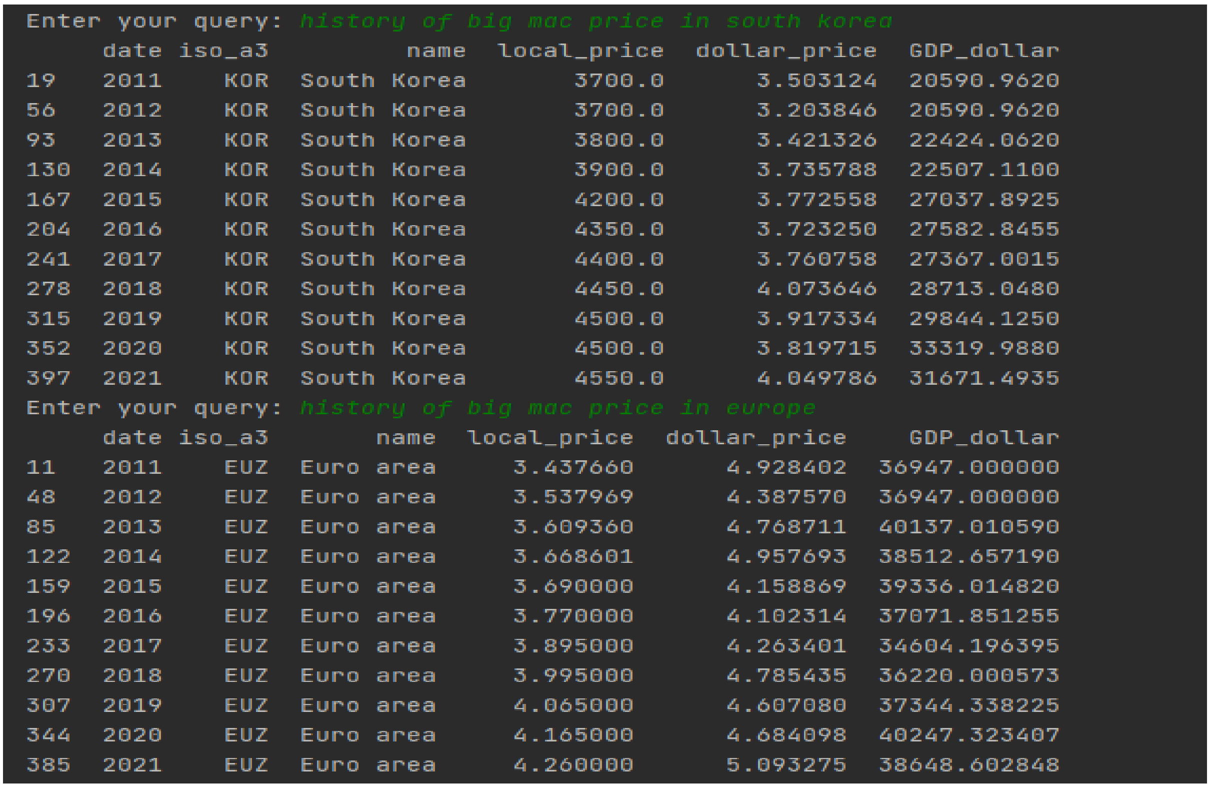
```
419  2021    TWN              Taiwan      72.000      2.570170  27089.643500
420  2021    UKR             Ukraine      63.500      2.295610   3679.937500
421  2021    URY             Uruguay     214.500      4.957101  15944.351000
422  2021    USA       United States       5.655      5.655000  64334.754000
423  2021    VNM             Vietnam   67500.000      2.928945   3457.606000
424  2021    ZAF        South Africa      33.500      2.221449   5522.552000
```



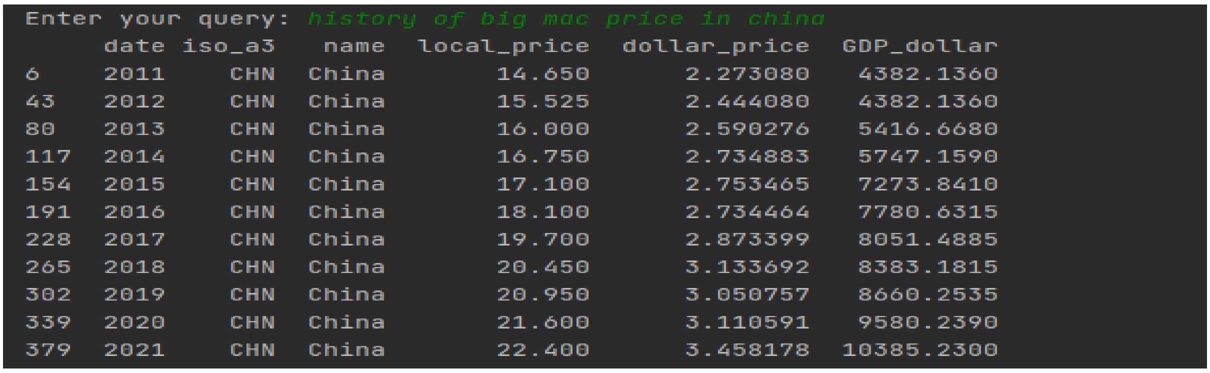
```
Enter your query: history of big mac price in india
     date iso_a3   name  local_price  dollar_price  GDP_dollar
16   2011    IND  India        84.00      1.891892   1264.8390
53   2012    IND  India        86.50      1.601479   1264.8390
90   2013    IND  India        89.50      1.583661   1513.6180
127  2014    IND  India       100.00      1.641757   1514.1260
164  2015    IND  India       116.25      1.859722   1568.2385
201  2016    IND  India       144.50      2.155920   1612.4775
238  2017    IND  India       174.00      2.622625   1670.3050
275  2018    IND  India       176.50      2.666116   1862.1770
312  2019    IND  India       180.50      2.611882   1995.0515
349  2020    IND  India       189.00      2.589560   2037.6920
393  2021    IND  India       190.00      2.567657   2031.3310
```



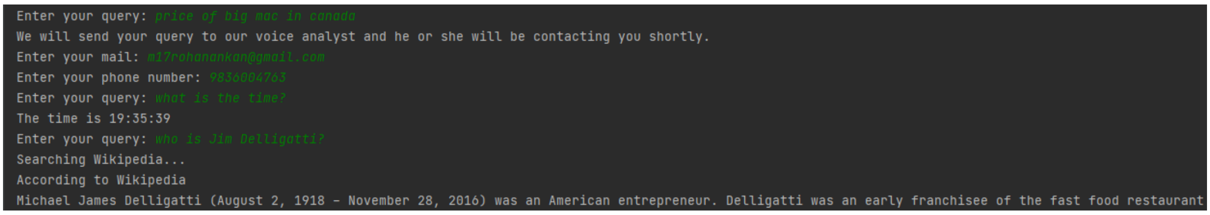
```
Enter your query: history of big mac price in usa
     date iso_a3           name  local_price  dollar_price  GDP_dollar
35   2011    USA  United States     4.065000      4.065000  47283.6330
72   2012    USA  United States     4.262360      4.262360  47283.6330
109  2013    USA  United States     4.462031      4.462031  48327.8610
146  2014    USA  United States     4.709583      4.709583  50018.4215
183  2015    USA  United States     4.790000      4.790000  53798.8125
220  2016    USA  United States     4.985000      4.985000  55087.5150
257  2017    USA  United States     5.180000      5.180000  55805.2040
294  2018    USA  United States     5.395000      5.395000  58554.3635
331  2019    USA  United States     5.660000      5.660000  59843.5065
368  2020    USA  United States     5.690000      5.690000  62868.9170
422  2021    USA  United States     5.655000      5.655000  64334.7540
```



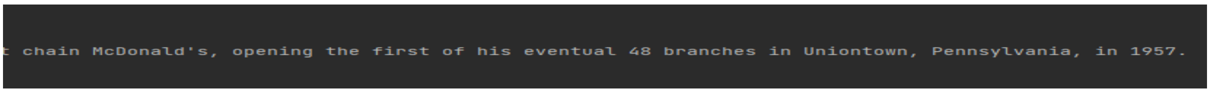
```
Enter your query: history of big mac price in south korea
     date iso_a3         name  local_price  dollar_price  GDP_dollar
19   2011    KOR  South Korea       3700.0      3.503124  20590.9620
56   2012    KOR  South Korea       3700.0      3.203846  20590.9620
93   2013    KOR  South Korea       3800.0      3.421326  22424.0620
130  2014    KOR  South Korea       3900.0      3.735788  22507.1100
167  2015    KOR  South Korea       4200.0      3.772558  27037.8925
204  2016    KOR  South Korea       4350.0      3.723250  27582.8455
241  2017    KOR  South Korea       4400.0      3.760758  27367.0015
278  2018    KOR  South Korea       4450.0      4.073646  28713.0480
315  2019    KOR  South Korea       4500.0      3.917334  29844.1250
352  2020    KOR  South Korea       4500.0      3.819715  33319.9880
397  2021    KOR  South Korea       4550.0      4.049786  31671.4935
Enter your query: history of big mac price in europe
     date iso_a3       name  local_price  dollar_price     GDP_dollar
11   2011    EUZ  Euro area     3.437660      4.928402  36947.000000
48   2012    EUZ  Euro area     3.537969      4.387570  36947.000000
85   2013    EUZ  Euro area     3.609360      4.768711  40137.010590
122  2014    EUZ  Euro area     3.668601      4.957693  38512.657190
159  2015    EUZ  Euro area     3.690000      4.158869  39336.014820
196  2016    EUZ  Euro area     3.770000      4.102314  37071.851255
233  2017    EUZ  Euro area     3.895000      4.263401  34604.196395
270  2018    EUZ  Euro area     3.995000      4.785435  36220.000573
307  2019    EUZ  Euro area     4.065000      4.607080  37344.338225
344  2020    EUZ  Euro area     4.165000      4.684098  40247.323407
385  2021    EUZ  Euro area     4.260000      5.093275  38648.602848
```



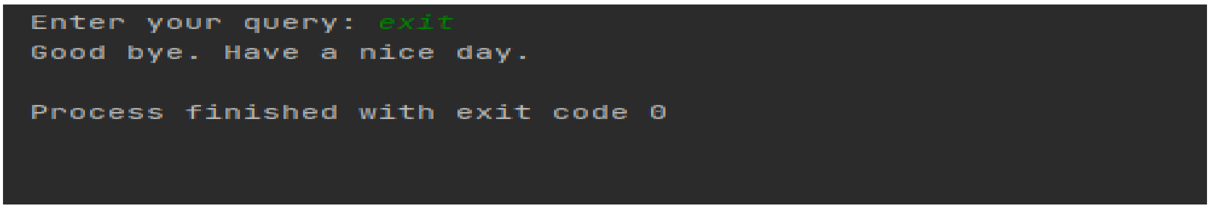
```
Enter your query: history of big mac price in china
     date iso_a3   name  local_price  dollar_price  GDP_dollar
6    2011    CHN  China       14.650      2.273080   4382.1360
43   2012    CHN  China       15.525      2.444080   4382.1360
80   2013    CHN  China       16.000      2.590276   5416.6680
117  2014    CHN  China       16.750      2.734883   5747.1590
154  2015    CHN  China       17.100      2.753465   7273.8410
191  2016    CHN  China       18.100      2.734464   7780.6315
228  2017    CHN  China       19.700      2.873399   8051.4885
265  2018    CHN  China       20.450      3.133692   8383.1815
302  2019    CHN  China       20.950      3.050757   8660.2535
339  2020    CHN  China       21.600      3.110591   9580.2390
379  2021    CHN  China       22.400      3.458178  10385.2300
```



```
Enter your query: price of big mac in canada
We will send your query to our voice analyst and he or she will be contacting you shortly.
Enter your mail: m17rohanankan@gmail.com
Enter your phone number: 9836004763
Enter your query: what is the time?
The time is 19:35:39
Enter your query: who is Jim Delligatti?
Searching Wikipedia...
According to Wikipedia
Michael James Delligatti (August 2, 1918 – November 28, 2016) was an American entrepreneur. Delligatti was an early franchisee of the fast food restaurant
```



```
t chain McDonald's, opening the first of his eventual 48 branches in Uniontown, Pennsylvania, in 1957.
```



```
Enter your query: exit
Good bye. Have a nice day.

Process finished with exit code 0
```

CONCLUSION

In this project, we have introduced a chatbot that is able to interact with users. This chatbot can answer queries in the textual user input. For this purpose, AIML with program-o has been used. The chatbot can answer only those questions which he has the answer in its AIML dataset. So, to increase the knowledge of the chatbot, we can add the APIs of Wikipedia, Weather Forecasting Department, Sports, News, Government and a lot more. In such cases, the user will be able to talk and interact with the chatbot in any kind of domain. Using APIs like Weather, Sports, News and Government Services, the chatbot will be able to answer the questions outside of its dataset and which are currently happening in the real world.

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

FUTURE SCOPE

The data taken was limited. The analysis of the results signifies that the integration of multidimensional data along with different classification, feature selection and dimensionality reduction techniques can provide auspicious tools for inference in this domain.

Further research in this field should be carried out for the better performance of the classification techniques so that it can predict on more variables.

I'm intending how to parametrize our classification techniques hence to achieve high accuracy. I'm looking into many datasets and how further Machine Learning algorithms can be used to characterize chatbot. I want to reduce the error rates with maximum accuracy.

The error can be minimized as well using other algorithms.

REFERENCES

1. Bayan Abu Shawar and Eric Atwell, 2007 “Chatbots: Are they Really Useful?”
2. LDV Forum - GLDV Journal for Computational Linguistics and Language
3. Technology.
4. Bringing chatbots into education: Towards natural language negotiation of open
5. learner models. Know. -Based Syst. 20, 2 (Mar. 2007), 177-185.
6. Intelligent Tutoring Systems: Prospects for Guided Practice and Efficient Learning. Whitepaper for the Army's Science of Learning Workshop, Hampton,
7. VA. Aug 1-3, 2006.

9. http://en.wikipedia.org/wiki/Chatterbot

10. Kaggle