**VIRTUAL ASSISTANT**

1. **INTRODUCTION**

In today’s era almost all tasks are digitalized. We have Smartphone in hands and it is nothing less than having world at your finger tips. These days we aren’t even using fingers. We just speak of the task and it is done. There exist systems where we can say Text Dad, “I’ll be late today.” And the text is sent. That is the task of a Virtual Assistant. It also supports specialized task such as booking a flight, or finding cheapest book online from various e-commerce sites and then providing an interface to book an order are helping automate search, discovery and online order operations.

Virtual Assistants are software programs that help you ease your day to day tasks, such as showing weather report, creating reminders, making shopping lists etc. They can take commands via text (online chat bots) or by voice. Voice based intelligent assistants need an invoking word or wake word to activate the listener, followed by the command. For my project the wake word is JIA. We have so many virtual assistants, such as Apple’s Siri, Amazon’s Alexa and Microsoft’s Cortana. For this project, wake word was chosen JIA.

This system is designed to be used efficiently on desktops. Personal assistant software improves user productivity by managing routine tasks of the user and by providing information from online sources to the user. JIA is effortless to use. Call the wake word ‘JIA’ followed by the command. And within seconds, it gets executed.

Voice searches have dominated over text search. Web searches conducted via mobile devices have only just overtaken those carried out using a computer and the analysts are already predicting that 50% of searches will be via voice by 2020.Virtual assistants are turning out to be smarter than ever. Allow your intelligent assistant to make email work for you. Detect intent, pick out important information, automate processes, and deliver personalized responses.

**1.1 BACKGROUND**

There already exist a number of desktop virtual assistants. A few examples of current virtual assistants available in market are discussed in this section along with the tasks they can provide and their drawbacks.

**SIRI from Apple**

SIRI is personal assistant software that interfaces with the user thru voice interface, recognizes commands and acts on them. It learns to adapt to user’s speech and thus improves voice recognition over time. It also tries to converse with the user when it does not identify the user request.

It integrates with calendar, contacts and music library applications on the device and also integrates with GPS and camera on the device. It uses location, temporal, social and task based contexts, to personalize the agent behavior specifically to the user at a given point of time.

**Supported Tasks**

* Call someone from my contacts list
* Launch an application on my iPhone
* Send a text message to someone
* Set up a meeting on my calendar for 9am tomorrow
* Set an alarm for 5am tomorrow morning
* Play a specific song in my iTunes library
* Enter a new note

**Drawback**

SIRI does not maintain a knowledge database of its own and its understanding comes from the information captured in domain models and data models.

**ReQall**

ReQall is personal assistant software that runs on smartphones running Apple iOS or Google Android operating system. It helps user to recall notes as well as tasks within a location and time context. It records user inputs and converts them into commands, and monitors current stack of user tasks to proactively suggest actions while considering any changes in the environment. It also presents information based on the context of the user, as well as filter information to the user based on its learned understanding of the priority of that information.

**Supported Tasks**

* Reminders
* Email
* Calendar, Google Calendar
* Outlook
* Evernote
* Facebook, LinkedIn
* News Feeds

**Drawback**

Will take some time to put all of the to-do items in – you could spend more time putting the entries in than actually doing the revision.

**1.2 OBJECTIVES**

Main objective of building personal assistant software (a virtual assistant) is using semantic data sources available on the web, user generated content and providing knowledge from knowledge databases. The main purpose of an intelligent virtual assistant is to answer questions that users may have. This may be done in a business environment, for example, on the business website, with a chat interface. On the mobile platform, the intelligent virtual assistant is available as a call-button operated service where a voice asks the user “What can I do for you?” and then responds to verbal input.

Virtual assistants can tremendously save you time. We spend hours in online research and then making the report in our terms of understanding. JIA can do that for you. Provide a topic for research and continue with your tasks while JIA does the research. Another difficult task is to remember test dates, birthdates or anniversaries. It comes with a surprise when you enter the class and realize it is class test today. Just tell JIA in advance about your tests and she reminds you well in advance so you can prepare for the test.

One of the main advantages of voice searches is their rapidity. In fact, voice is reputed to be four times faster than a written search: whereas we can write about 40 words per minute, we are capable of speaking around 150 during the same period of time15. In this respect, the ability of personal assistants to accurately recognize spoken words is a prerequisite for them to be adopted by consumers.

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

* 1. **PURPOSE, SCOPE AND APPILCABILITY**

**Purpose**

Purpose of virtual assistant is to being capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news. Virtual assistants enable users to speak natural language voice commands in order to operate the device and its apps.

There is an increased overall awareness and a higher level of comfort demonstrated specifically by millennial consumers. In this ever-evolving digital world where speed, efficiency, and convenience are constantly being optimized, it’s clear that we are moving towards less screen interaction.

**Scope**

Voice assistants will continue to offer more *individualized* experiences as they get better at differentiating between voices. Developers will also need to focus on maintaining a user experience that is consistent within the coming years as complexity becomes more of a concern. This is because the visual interface with voice assistants is missing. Users simply cannot see or touch a voice interface.

**Applicability**

The mass adoption of artificial intelligence in users’ everyday lives is also fueling the shift towards voice. The number of IoT devices such as smart thermostats and speakers are giving voice assistants more utility in a connected user’s life. Smart speakers are the number one way we are seeing voice being used. Many industry experts even predict that nearly every application will integrate voice technology in some way in the next 5 years.

The use of virtual assistants can also enhance the system of IoT (Internet of Things). Twenty years from now, Microsoft and its competitors will be offering personal digital assistants that will offer the services of a full-time employee usually reserved for the rich and famous.

1. **SURVEY OF TECHNOLOGY**

**Python**

Python is OOPs (Object Oriented Programming) based, high level, interpreted programming language. It is a robust, highly useful language focused on rapid application development (RAD). Python helps in easy writing and execution of codes. Python can implement the same logic with as much as 1/5th code as compared to other OOPs languages.

Python provides a huge list of benefits to all. The usage of Python is such that it cannot be limited to only one activity. Its growing popularity has allowed it to enter into some of the most popular and complex processes like Artificial Intelligence (AI), Machine Learning (ML), natural language processing, data science etc. Python has a lot of libraries for every need of this project. For JIA, libraries used are speechrecognition to recognize voice, Pyttsx for text to speech, selenium for web automation etc.

Python is reasonably efficient. Efficiency is usually not a problem for small examples. If your Python code is not efficient enough, a general procedure to improve it is to find out what is taking most the time, and implement just that part more efficiently in some lower-level language. This will result in much less programming and more efficient code (because you will have more time to optimize) than writing everything in a low-level language.

**Pyttsx**

Pyttsx stands for Python Text to Speech. It is a cross-platform Python wrapper for text-to-speech synthesis. It is a Python package supporting common text-to-speech engines on Mac OS X, Windows, and Linux. It works for both Python2.x and 3.x versions. Its main advantage is that it works offline.

**Speech Recognition**

This is a library for performing speech recognition, with support for several engines and APIs, online and offline*.* It supports APIs like Google Cloud Speech API, IBM Speech to Text, Microsoft Bing Voice Recognition etc.

**SQLite**

SQLite is a capable library, providing an in-process relational database for efficient storage of small-to-medium-sized data sets. It supports most of the common features of SQL with few exceptions. Best of all, most Python users do not need to install anything to get started working with SQLite, as the standard library in most distributions ships with the sqlite3 module.

SQLite runs embedded in memory alongside your application, allowing you to easily extend SQLite with your own Python code. SQLite provides quite a few hooks, a reasonable subset of which are implemented by the standard library database driver.

1. **REQUIREMENT AND ANALYSIS**

System Analysis is about complete understanding of existing systems and finding where the existing system fails. The solution is determined to resolve issues in the proposed system. It defines the system. The system is divided into smaller parts. Their functions and inter relation of these modules are studied in system analysis. The complete analysis is followed below.

**3.1 Problem definition**

Usually, user needs to manually manage multiple sets of applications to complete one task. For example, a user trying to make a travel plan needs to check for airport codes for nearby airports and then check travel sites for tickets between combinations of airports to reach the destination. There is need of a system that can manage tasks effortlessly.

We already have multiple virtual assistants. But we hardly use it. There are number of people who have issues in voice recognition. These systems can understand English phrases but they fail to recognize in our accent. Our way of pronunciation is way distinct from theirs. Also, they are easy to use on mobile devices than desktop systems. There is need of a virtual assistant that can understand English in Indian accent and work on desktop system.

When a virtual assistant is not able to answer questions accurately, it’s because it lacks the proper context or doesn’t understand the intent of the question. Its ability to answer questions relevantly only happens with rigorous optimization, involving both humans and machine learning. Continuously ensuring solid quality control strategies will also help manage the risk of the virtual assistant learning undesired bad behaviors. They require large amount of information to be fed in order for it to work efficiently.

Virtual assistant should be able to model complex task dependencies and use these models to recommend optimized plans for the user. It needs to be tested for finding optimum paths when a task has multiple sub-tasks and each sub-task can have its own sub-tasks. In such a case there can be multiple solutions to paths, and the it should be able to consider user preferences, other active tasks, priorities in order to recommend a particular plan.

**3.2 REQUIREMENT SPECIFICATION**

Personal assistant software is required to act as an interface into the digital world by understanding user requests or commands and then translating into actions or recommendations based on agent’s understanding of the world.

JIA focuses on relieving the user of entering text input and using voice as primary means of user input. Agent then applies voice recognition algorithms to this input and records the input. It then use this input to call one of the personal information management applications such as task list or calendar to record a new entry or to search about it on search engines like Google, Bing or Yahoo etc. Focus is on capturing the user input through voice, recognizing the input and then executing the tasks if the agent understands the task. Software takes this input in natural language, and so makes it easier for the user to input what he or she desires to be done.

Voice recognition software enables hands free use of the applications, lets users to query or command the agent through voice interface. This helps users to have access to the agent while performing other tasks and thus enhances value of the system itself. JIA also have ubiquitous connectivity through Wi-Fi or LAN connection, enabling distributed applications that can leverage other APIs exposed on the web without a need to store them locally.

Virtual assistants must provide a wide variety of services. These include:

* Providing information such as weather, facts from e.g. Wikipedia etc.
* Set an alarm or make to-do lists and shopping lists.
* Remind you of birthdays and meetings.
* Play music from streaming services such as Saavn and Gaana.
* Play videos, TV shows or movies on televisions, streaming from e.g. Netflix or Hotstar.
* Book tickets for shows, travel and movies.

**Feasibility Study**

Feasibility study can help you determine whether or not you should proceed with your project. It is essential to evaluate cost and benefit. It is essential to evaluate cost and benefit of the proposed system. Five types of feasibility study are taken into consideration.

1. **Technical feasibility:** It includes finding out technologies for the project, both hardware and software. For virtual assistant, user must have microphone to convey their message and a speaker to listen when system speaks. These are very cheap now a days and everyone generally possess them. Besides, system needs internet connection. While using JIA, make sure you have a steady internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.
2. **Operational feasibility:** It is the ease and simplicity of operation of proposed system. System does not require any special skill set for users to operate it. In fact, it is designed to be used by almost everyone. Kids who still don’t know to write can read out problems for system and get answers.
3. **Economical feasibility:** Here, we find the total cost and benefit of the proposed system over current system. For this project, the main cost is documentation cost. User also would have to pay for microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, JIA won’t cost too much.
4. **Organizational feasibility:** This shows the management and organizational structure of the project. This project is not built by a team. The management tasks are all to be carried out by a single person. That won’t create any management issues and will increase the feasibility of the project.
5. **Cultural feasibility:** It deals with compatibility of the project with cultural environment. Virtual assistant is built in accordance with the general culture. The project is named JIA so as to represent Indian culture without undermining local beliefs.

This project is technically feasible with no external hardware requirements. Also it is simple in operation and does not cost training or repairs. Overall feasibility study of the project reveals that the goals of the proposed system are achievable. Decision is taken to proceed with the project.

* 1. **PLANNING AND SCHEDULING**

The project is planned on the keeping in mind that there are resources available online which contributes to the development process like libraries and frameworks. Building a virtual assistant is a task which involves abundance of user requirements. Planning must be done to ensure on time delivery of the project.

**PERT Chart**

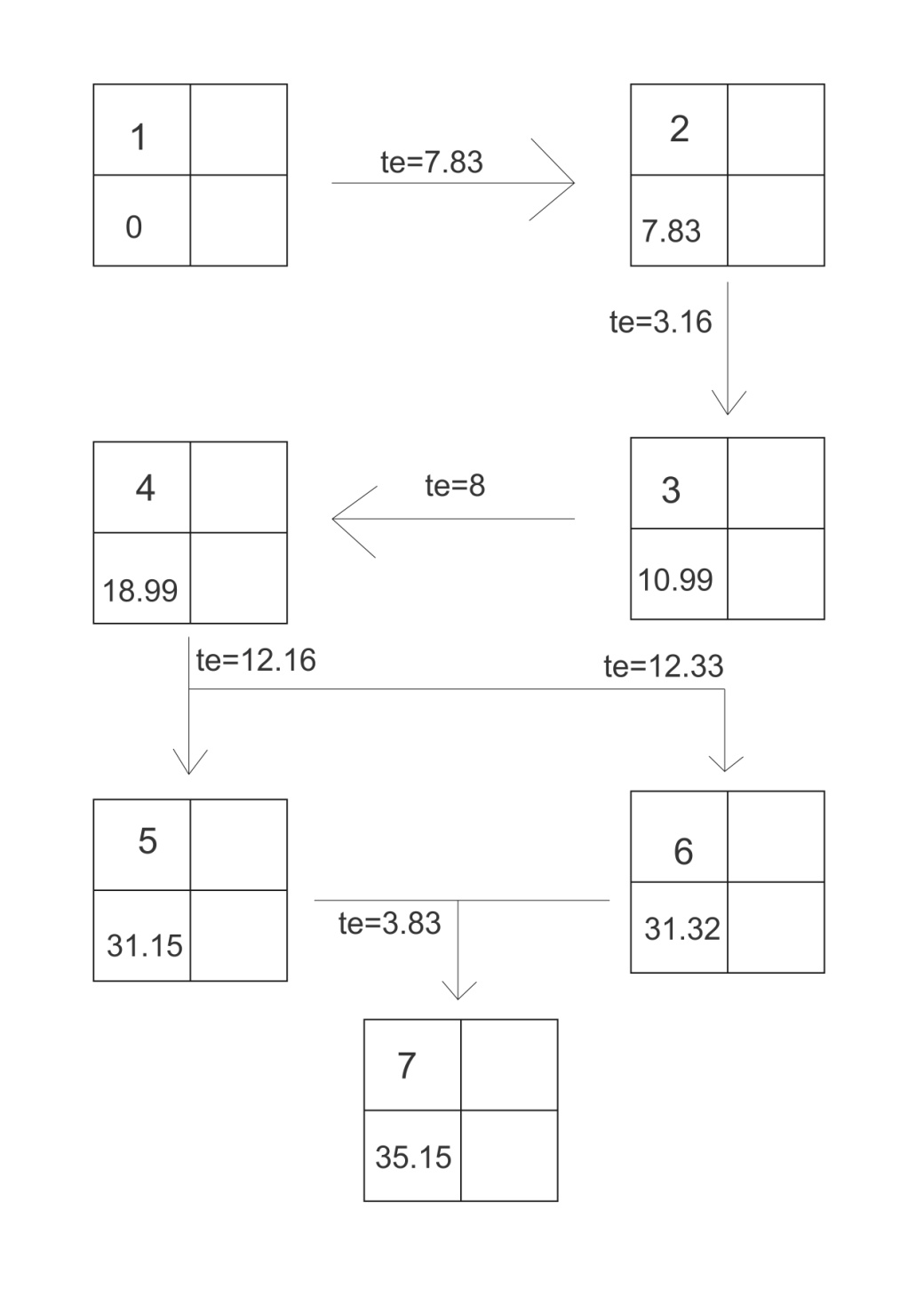
Project Evaluation and Review Technique (PERT) requires three time estimates which are counted in weeks, generally.

* Most-likely time: Time expected by a task to take under normal circumstances.
* Optimistic time: Shortest time which is expected under no risks.
* Pessimistic time: Worst possible time considering risks and failures.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Activity | Optimistic time (a) | Most-likely time (m) | Pessimistic time (b) |
| 1 |  |  |  |  |
| 2 | Requirement Gathering | 7 | 8 | 8 |
| 3 | Requirement Analysis | 3 | 3 | 4 |
| 4 | Design | 8 | 8 | 9 |
| 5 | Coding | 11 | 12 | 14 |
| 6 | Testing | 12 | 12 | 14 |
| 7 | Implementation | 3 | 4 | 4 |

Now, we will calculate te which is expected time calculated by

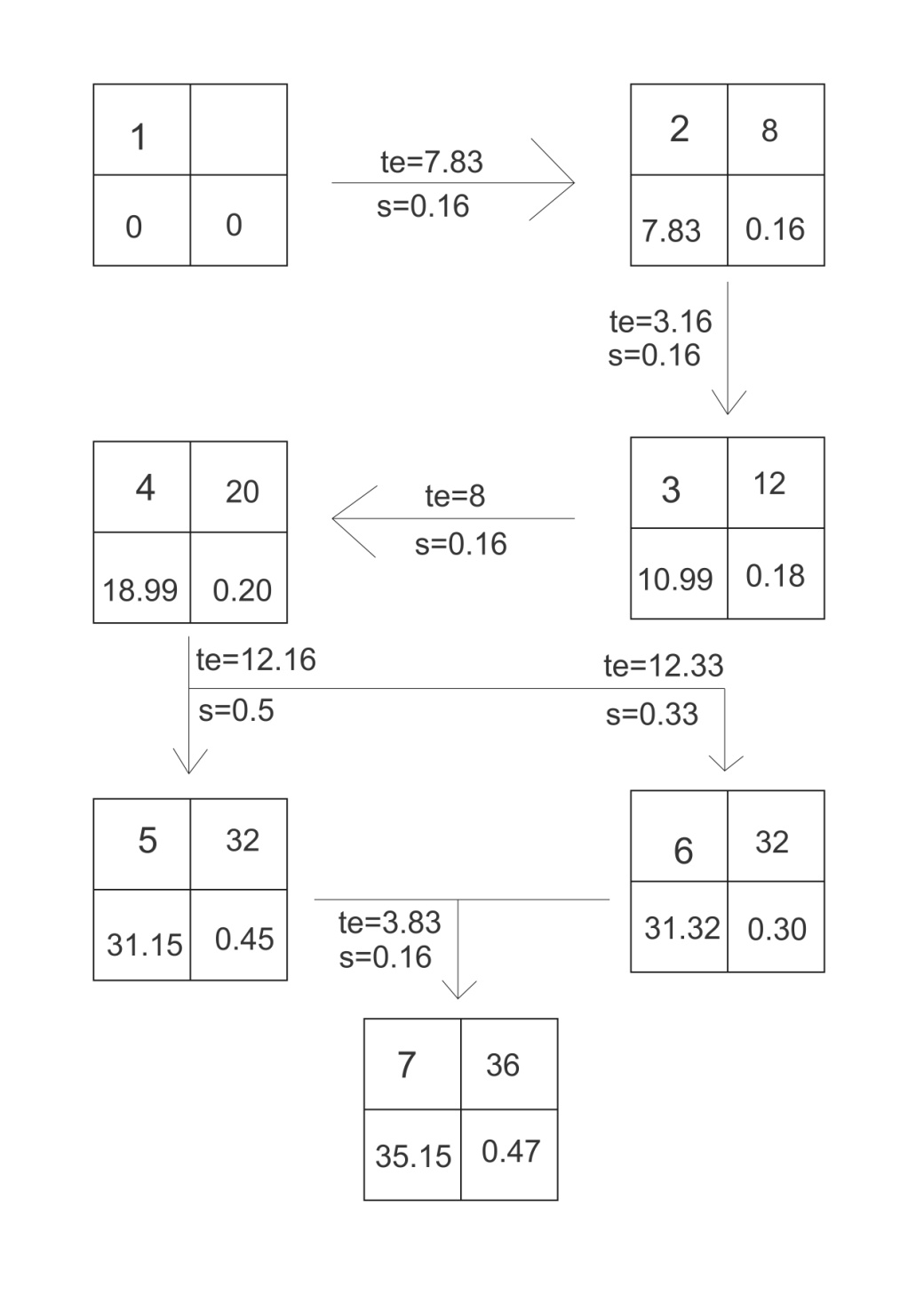
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Optimistic time (a) | Most-likely time (m) | Pessimistic time (b) | Expected time (te) |
| 2 | 7 | 8 | 8 | 7.83 |
| 3 | 3 | 3 | 4 | 3.16 |
| 4 | 8 | 8 | 9 | 8 |
| 5 | 11 | 12 | 14 | 12.16 |
| 6 | 12 | 12 | 14 | 12.33 |
| 7 | 3 | 4 | 4 | 3.83 |



Pert chart after adding expected time.

Calculating degree of uncertainty that is standard deviation s,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr.No | Optimistic  time (a) | Most-likely  time (m) | Pessimistic  time (b) | Expected  time (te) | Standard deviation (s) |
| 2 | 7 | 8 | 8 | 7.83 | 0.16 |
| 3 | 3 | 3 | 4 | 3.16 | 0.16 |
| 4 | 8 | 8 | 9 | 8 | 0.16 |
| 5 | 11 | 12 | 14 | 12.16 | 0.50 |
| 6 | 12 | 12 | 14 | 12.33 | 0.33 |
| 7 | 3 | 4 | 4 | 3.83 | 0.16 |



Pert chart after adding target date and standard deviation.

For the events that have target date T, we will calculate their *z* values,

Calculate corresponding probabilities of not meeting the target date by using graph of equivalent *z* values.

|  |  |  |
| --- | --- | --- |
| Sr.No | *z* value | Probability Percentage |
| 2 | 1.06 | 14% |
| 3 | 5.61 | 0% |
| 4 | 5.05 | 0% |
| 5 | 1.88 | 3.01% |
| 6 | 2.26 | 1.19% |
| 7 | 1.81 | 3.5% |

Probability of not meeting the target date is highest in requirement gathering stage. This indicates the process of requirement gathering must be done efficiently considering the time constraint.

**Gantt Chart**

According to PERT chart, it is estimated that project will be concluded in approximately 36 weeks. The Gantt chart is prepared accordingly which is shown in diagram below:



The process of requirement gathering is allotted eight weeks. That will help in reducing the risk of failure in later processes. Take enough time to gather and elicitate requirements.

Coding and testing process are to be done concurrently because of time constraints. As well as fixing the bugs would take time. Doing testing after the development process may prove hectic and time consuming. Four weeks are kept specifically for implementation. This is done keeping in mind risks that may arise after the deployment of project.

* 1. **HARDWARE AND SOFTWARE REQUIREMENTS**

The software is designed to be light-weighted so that it doesn’t be a burden on the machine running it. This system is being build keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirement for virtual assistant.

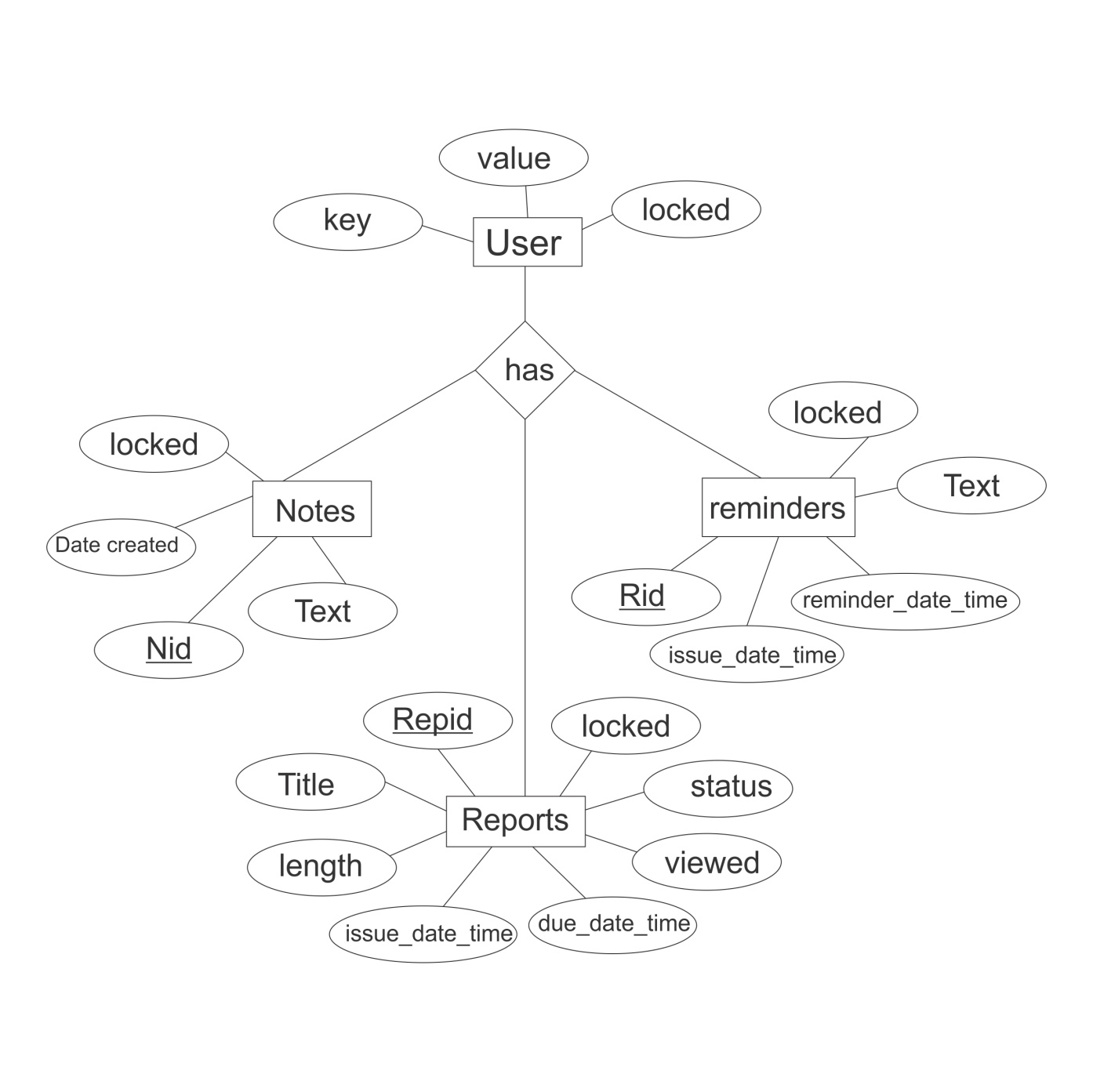
**Hardware:**

* Pentium-pro processor or later.
* RAM 512MB or more.

**Software:**

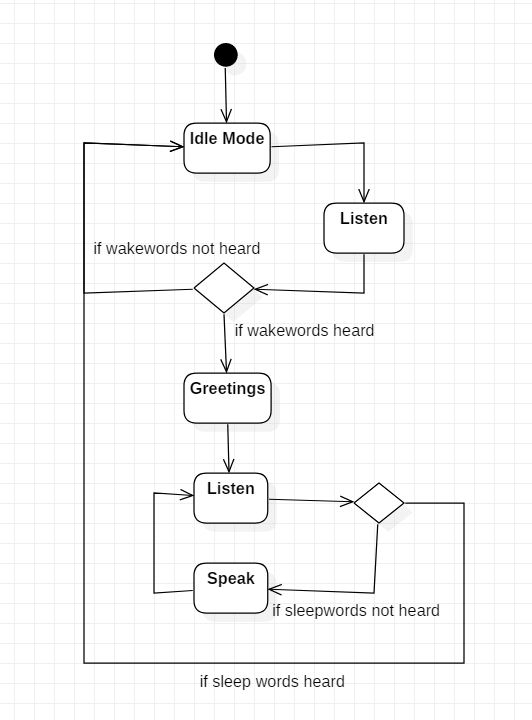
* Windows 7(32-bit) or above.
* Python 2.7 or later
* Chrome Driver
* Selenium Web Automation
* SQLite

1. **SYSTEM DESIGN**
   1. **ER DIAGRAM**



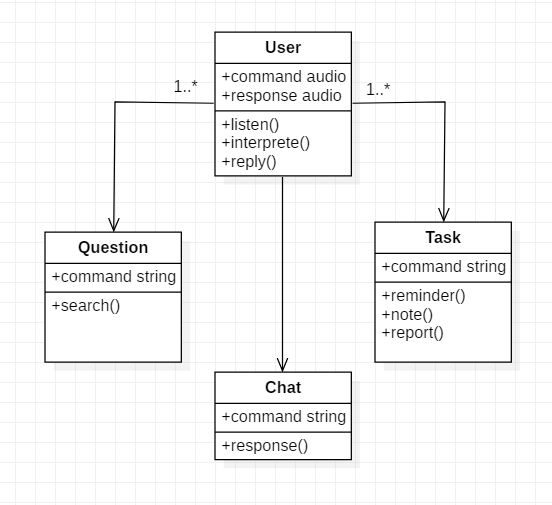
The above diagram shows entities and their relationship for a virtual assistant system. We have a user of a system who can have their keys and values. It can be used to store any information about the user. Say, for key “name” value can be “Jim”. For some keys user might like to keep secure. There he can enable lock and set a password (voice clip).

* 1. **ACTIVITY DIAGRAM**



Initially, the system is in idle mode. As it receives any wake up cal it begins execution. The received command is identified whether it is a questionnaire or a task to be performed. Specific action is taken accordingly. After the Question is being answered or the task is being performed, the system waits for another command. This loop continues unless it receives quit command. At that moment, it goes back to sleep.

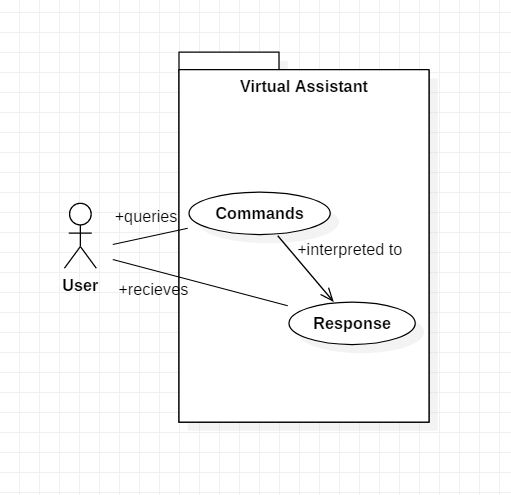
* 1. **CLASS DIAGRAM**



The class user has 2 attributes command that it sends in audio and the response it receives which is also audio. It performs function to listen the user command. Interpret it and then reply or sends back response accordingly. Question class has the command in string form as it is interpreted by interpret class. It sends it to general or about or search function based on its identification.

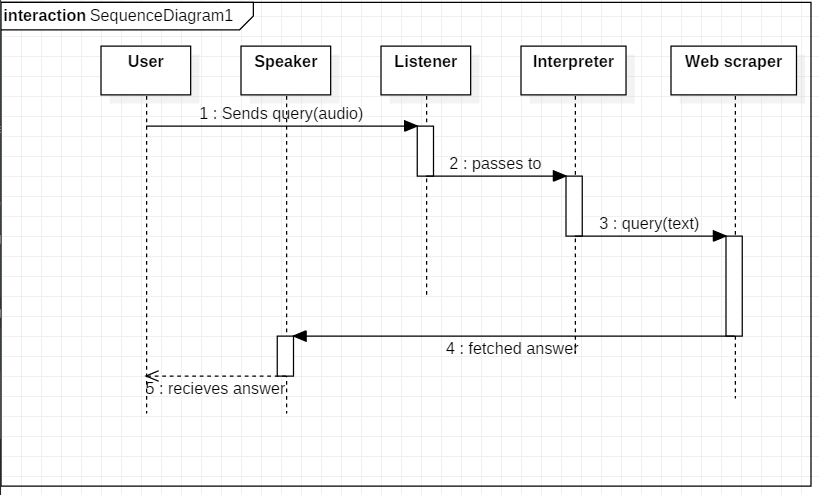
The task class also has interpreted command in string format. It has various functions like reminder, note, mimic, research and reader.

* 1. **USE CASE DIAGRAM**



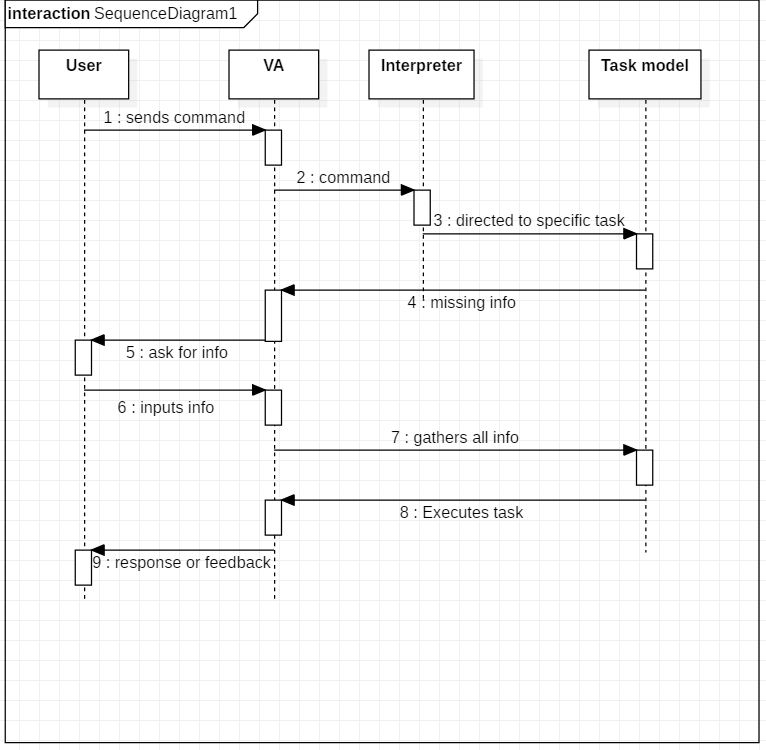
In this project there is only one user. The user queries command to the system. System then interprets it and fetches answer. The response is sent back to the user.

* 1. **SEQUENCE DIAGRAM**
     1. Sequence diagram for Query-Response



The above sequence diagram shows how an answer asked by the user is being fetched from internet. The audio query is interpreted and sent to Web scraper. The web scraper searches and finds the answer. It is then sent back to speaker, where it speaks the answer to user.

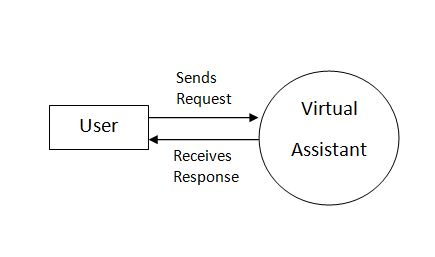
* + 1. Sequence diagram for Task Execution



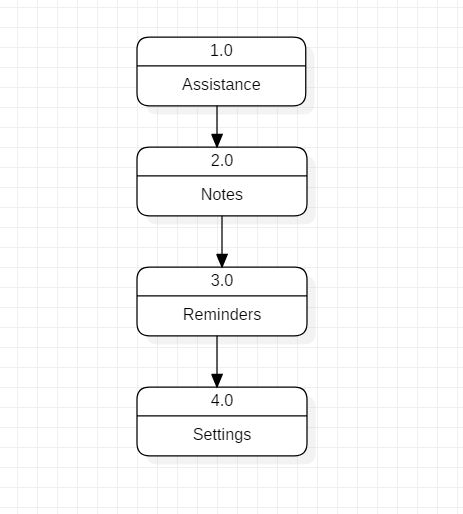
The user sends command to virtual assistant in audio form. The command is passed to the interpreter. It identifies what the user has asked and directs it to task executer. If the task is missing some info, the virtual assistant asks user back about it. The received information is sent back to task and it is accomplished. After execution feedback is sent back to user.

* 1. **DATA FLOW DIAGRAM**

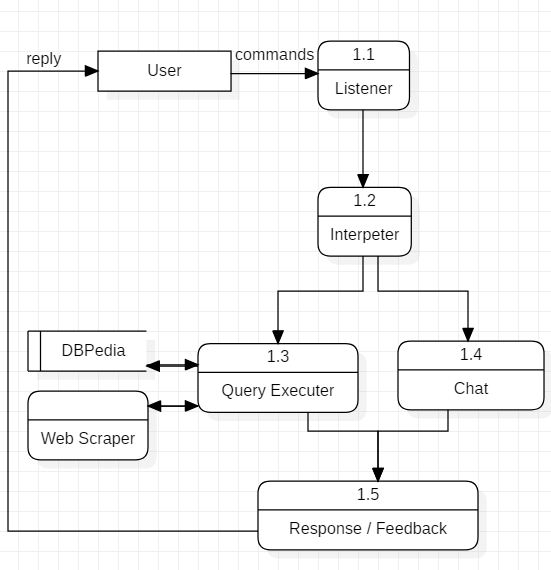
4.6.1 DFD Level 0 (Context Level Diagram)



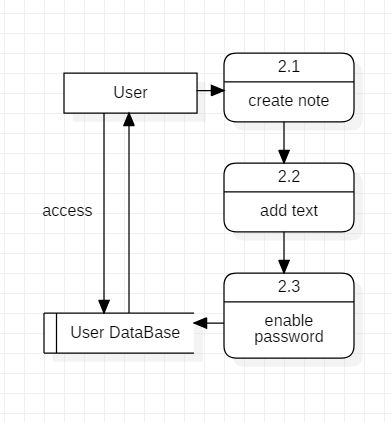
4.6.2 DFD Level 1



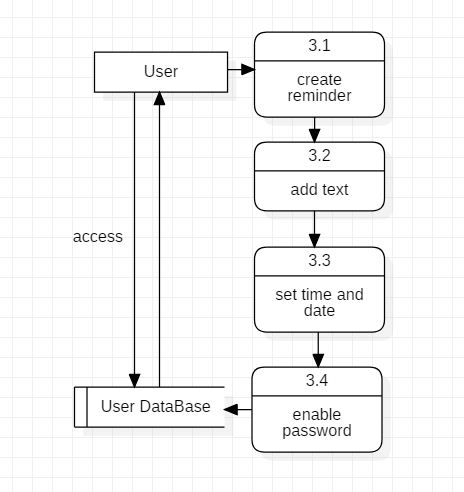
4.6.3 DFD Level 2



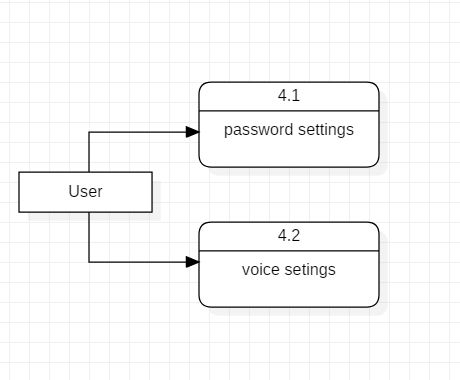
Data Flow in Assistance



Creating notes Data Flow

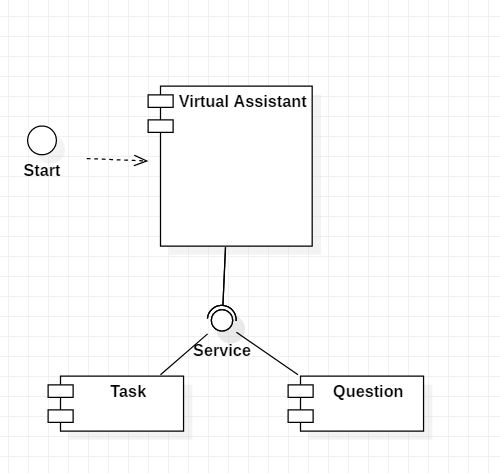


Data Flow in creating reminders



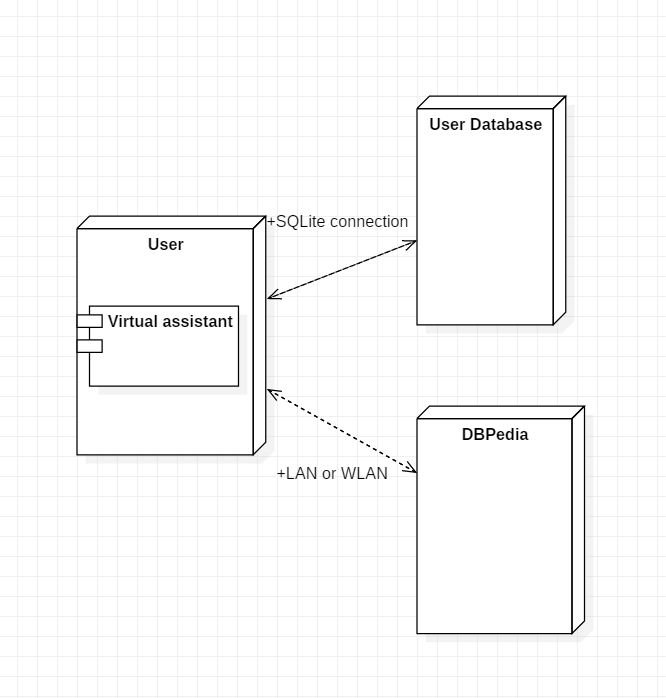
Settings of virtual Assistant

**4.7 COMPONENT DIAGRAM**



The main component here is the Virtual Assistant. It provides two specific service, executing Task or Answering your question.

* 1. **DEPLOYMENT DIAGRAM**



The user interacts with SQLite database using SQLite connection in Python code. The knowledge database DBPedia must be accessed via internet connection. This requires LAN or WLAN / Ethernet network.

* 1. **DATA DICTIONARY**

User

|  |  |
| --- | --- |
| Key | Text |
| Value | Text |
| Locked | Boolean |

Reminder

|  |  |
| --- | --- |
| Rid | Integer PRIMARY KEY |
| Text | Text |
| issue\_date\_time | Date Time |
| Reminder\_date\_time | Date Time |
| Locked | Boolean |

Note

|  |  |
| --- | --- |
| Nid | Integer PRIMARY KEY |
| Title | Text |
| Text | Text |
| Date\_created | Date Time |
| Locked | Boolean |

Report

|  |  |
| --- | --- |
| Repid | Integer PRIMARY KEY |
| Title | Text |
| Length | Integer |
| Issue\_date\_time | Date Time |
| Due\_date\_time | Date Time |
| Status | Text (Done/ Doing) |
| Viewed | Boolean |
| Locked | Boolean |

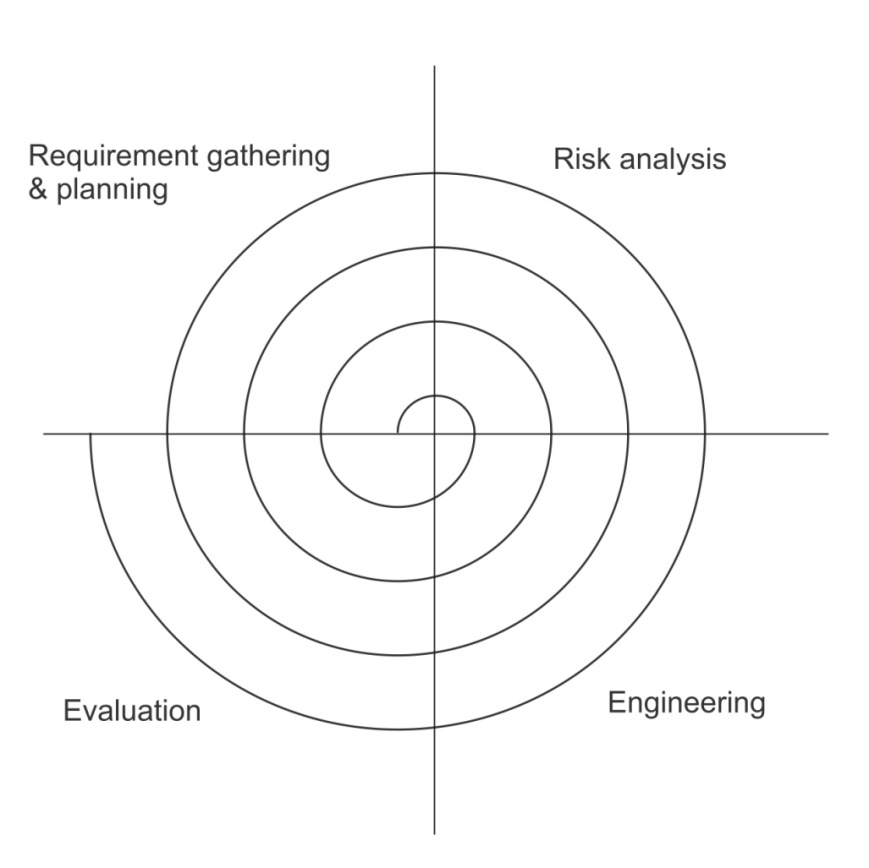
1. **IMPLEMENTATION AND TESTING**

In the Implementation phase, all the tasks are executed as per the project plan developed during the planning phase. However, the project plan may have to be revised periodically to accommodate any changes to the project plan that may arise on account of change requests, risks and various events that may occur during execution of the plan.

* 1. **IMPLEMENTATION APPROACH**

Changes are expected in the product during the development phase as requirements are subject to change. And also it is not possible to anticipate the project risk at the start of the project. Hence I chose **spiral model**. It is another way of looking at waterfall model. The development process proceeds in a spiral where each loop terminates with an evaluation before the next iteration is embarked upon. The main objective of spiral model is to identify risk areas in the project.

For this project, the system is divided into modules. And each module undergoes the spiral phases of requirement gathering and planning, risk analysis, engineering and evaluation.



1. **Requirement gathering and planning includes :**

* Problem identification.
* Deciding product specification.
* Deciding objective of the system.
* Deciding schedule of the system.
* Plan number of iterations required to complete software.

1. **Risk Analysis includes :**

* Identification of risk areas.
* Measuring cost and efforts related to risk.
* Efforts are taken to resolve risk.

1. **Engineering includes :**

* Design the prototype.
* Convert design to code
* Implement the developed system.
* Do the testing.
* Collect feedback

1. **Evaluation includes :**

* Evaluate the product.
* Repeat the loop if required.

Each and every loop goes through testing which makes it easy to recover any error and fix it then and there itself. In this model, we don’t have to start the work from beginning.  The spiral model incorporates the stepwise approach of the Classical Waterfall Model. The spiral model uses the approach of Prototyping Model by building a prototype at the start of each phase as a risk handling technique. Also, the spiral model can be considered as supporting the evolutionary model – the iterations along the spiral can be considered as evolutionary levels through which the complete system is built.

* 1. **CODING DETAILS AND CODE EFFICIENCY**
     1. **Coding details**

Code represents your design in working. It is very important to understand the architecture before you begin writing code. According to the designs in system design chapter, all the loops and conditions are identified well in advance. The code is well written with comments wherever required. It is done so to make sure the code is well organized and maintainable. Here are a few lines of code that symbolizes the use of comments.

#imported stuff

import pyttsx

import speech\_recognition as sr

import Tkinter as tk

#initialisation

engine = pyttsx.init()

engine.setProperty('rate', 120)

#main class

class App(threading.Thread):

def \_\_init\_\_(self, tk\_root):

self.root = tk\_root

threading.Thread.\_\_init\_\_(self)

self.start()

def run(self):

wake()

#begins execution

Root=tk.Tk()

APP = App(root)

root.mainloop()

* + 1. **Code efficiency**

Code efficiency is directly linked with algorithmic efficiency and the speed of runtime execution for software. It is the key element in ensuring high performance. Code efficiency plays a significant role in applications in a high-execution-speed environment where performance and scalability are paramount. One of the recommended best practices in coding is to ensure good code efficiency.

But efficiency in coding isn’t only about creating tight algorithms.  It’s also about being able to reduce waste.  This means waste in terms of how much time you spend fixing problems and also waste in terms of consuming too many computer resources. The more complex you make your code; the more difficult it can be to untangle it.

This project uses the best keywords, data types and variables, and other available programming concepts to implement the related algorithm. Here are a few algorithm pseudo code used in the project.

* **Main Loop**

while (True):

x=listen()

if x in wakewords:

greet()

while (True):

x=listen()

if (x != byewords):

speak(x)

else:

goodbye()

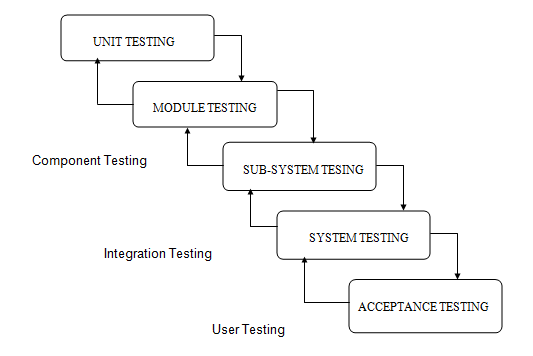
break

The external while loop has no breaking condition. It must be running forever to listen if the user asks something again. As and when the program hears the wakewords, it comes into action. The other while loop runs a listen, takes command or query executes it and wait for another query. This continues until the user is satisfied and says bye.

* 1. **TESTING APPROACH**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive. A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.



* + 1. **Unit Testing**

Unit is the smallest part of software which includes functions, methods, interfaces and classes and can be tested individually for correctness. Unit testing is a testing technique using which individual modules are tested to determine if they are fit for use. It is concerned with functional correctness of the standalone modules.

The main aim is to isolate each unit of the system to identify, analyze and fix the defects. Unit tests are typically written and run by developers to ensure that code meets its design and behaves as intended. A unit provides a strict written contract that the piece of code must satisfy. As a result, it affords several benefits.

All the individual units of this project were tested well before integration as per the test scenario and test cases designed during the design phase. Unit test find problems early in the development cycle.

* + 1. **Integration testing**

Integration testing tests integration or interfaces between components. Integration testing is done by specific integration tester or test team. Integration testing also tests the functionality of the software under review. Integration testing mainly focuses on output protocols and parameters passing between different units, modules / systems. Focus of integration is mainly on low-level design, architecture and construction of software. Integration testing is considered a structural testing.

Integration testing involves the concept of Stubs and Drivers. Stub is a piece of code emulating a called function. In absence of a called function, stub may take care of that part for testing purpose. Driver is a piece of code emulating a calling function. In absence of actual function calling the piece of code under testing, diver tries to work as calling function.

After successful unit testing is achieved, the modules and sub systems of the project were integrated in a sequential manner. So the cause of failure may be analyzed as early as possible. It was found that all the modules are integrated and functioning as required.

* + 1. **System Testing**

System testing represents the final testing done on a system before it is delivered to the customer. It is done on integrated sub systems that make up the entire system, or the final system getting delivered to the customer. System testing validates that the entire system meets its functional / non functional requirements as defined by the customers in software requirements specification. The criteria for system testing may involve in entire domain or selected parts depending on the scope of testing.

System testing is carried out by specialist test team or independent testers. Testers must also deal with incomplete or undocumented requirements. The test environment should correspond to the final target or production environment as much as possible in order to minimize the risk of environment specific failures not being found by testing.

After all the modules and their integration is done. The system was tested as a whole application. Few defects were still encountered which were fixed immediately and regression testing was performed.

* + 1. **Acceptance Testing**

After the system test has corrected almost all the defects, the system will be delivered to the user or customer for acceptance testing. The goal of acceptance testing is to establish confidence in the system. Acceptance testing is most often focused on a validation type testing. This testing is done in two phases; Alpha testing and Beta testing.

Alpha testing is testing of an application when development is about to complete, that is before the delivery of product. Minor design changes can still be made as a result of alpha testing. Beta testing is also called as field testing as it takes place at customer’s site after the delivery of the product.

This project was designed for every end user who intends to automate his daily tasks over desktops and laptops. It was observed that users found the application easy to use according to today’s technical era.

**TEST CASE DESIGN**

1. **Detection of microphone.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC1.1 | Microphone connected. | Connect microphone and then start the program. | Hello Jia | Jia greets | As expected | Pass |
| TC1.2 | Microphone not connected. | Do not connect microphone and start the program. | Hello Jia | I can’t hear you. Make sure microphone is connected. | As expected | Pass |

1. **Application starts on wake word.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC2.1 | Check application start on wake word. | Speak any of the wake words to start program. | Hello Jia | Jia greets | As expected | Pass |
| TC2.2 | Check application does not start without wake word. | Speak anything in microphone other than wake words. | What time is it? | No response. | As expected | Pass |

1. **Hearing input voice.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC3.1 | Input voice heard. | Speak input query clearly. | What is the height of mount Everest? | 8,848 m. | As expected | Pass |
| TC3.2 | Input voice not heard. | Speak input query in noisy environment. | What is the height of mount Everest?  (Speak noisily) | Sorry, I didn’t hear that. | As expected | Pass |

1. **Approximate answers on mathematical questions.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC4.1 | Approximate answers. | Ask any mathematical question. | What is the value of pi (π)? | 3.14 | As expected | Pass |
| TC4.2 | Accurate answers. | Ask any mathematical question. | What is the answer of 11/7? | 1.57142857 | As expected | Pass |

1. **Differentiate between query and chat.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC5.1 | Input is a query. | Ask any query. | What time is it? | 12:27 PM | As expected | Pass |
| TC5.2 | Input is a dialogue. | Ask about Jia. | You are great. | Thanks, I work hard. | As expected | Pass |

1. **Set password on user data.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC6.1 | Not asking password. | Tell her about best friend. | Rita is my best friend. | Added user data. | As expected | Pass |
| TC6.2 | Ask for password. | Tell her a secret. | I kept my car keys at second shelf.  Keep it protected. | Set password on this data. | As expected | Pass |

1. **Testing user data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC7.1 | User data is known. | Ask something she knows. | Who is my best friend? | Rita is your best friend. | As expected | Pass |
| TC7.2 | User data is unknown. | Ask something new. | When is my birthday? | You never shared that. | As expected | Pass |

1. **Create new password.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC8.1 | Both repetitions match. | Repeat same password twice. | Password = SecureJia  Password = SecureJia | Password set successfully. | As expected | Pass |
| TC8.2 | Repetitions do not match. | Repeat different password. | Password = SecureJia  Password = SafeJia | Passwords do not match. | As expected | Pass |

1. **Change password.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC9.1 | Correct previous password. | Speak true previous password. | oldPassword = SecureJia | You can set new password. | As expected | Pass |
| TC9.2 | Incorrect previous password. | Speak wrong previous password. | oldPassword = SafeJia | Password does not match. | As expected | Pass |

1. **Fetch online query.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC10.1 | Results found. | Ask something obvious. | What is the height of mount Everest? | 8,848 m. | As expected | Pass |
| TC10.2 | Results not found. | Ask something unspecific. | What is the height of mount Tablow? | I didn’t get you. Please be more specific. | As expected | Pass |

1. **Internet connection for online queries**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC11.1 | Internet available. | Ask a query. | What is the height of mount Everest? | 8,848 m. | As expected | Pass |
| TC11.2 | Internet not available. | Disconnect the internet. | What is the height of mount Everest? | No internet connection. | As expected | Pass |

1. **Follow through queries**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC12.1 | Asking follow-up questions. | Ask a query.  Ask another related query. | * What is the height of Mount Everest? * Where is it located? | * 8,848 m. * Border of Tibet and Nepal. | As expected | Pass |
| TC12.2 | Asking different questions | Ask two differentiable queries. | * What is the height of Mount Everest? * Who built Taj Mahal? | * 8,848 m. * Shah Jahan | As expected | Pass |

1. **Create a note**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC13.1 | Incomplete details. | Create a note. (unspecified) | Create a note | What goes in note? | As expected | Pass |
| TC13.2 | Complete details. | Ask to create a note. | Create a note that I lost to Rita yesterday. | Note created. | As expected | Pass |

1. **Create Reminder**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC14.1 | Complete details. | Create a reminder. | Remind me to mail Rita this evening. | Reminder set for evening. | As expected | Pass |
| TC14.2 | Incomplete details. | Ask to create a reminder without time. | Remind me to mail Rita | When to remind? | As expected | Pass |
| Create reminder without mentioning what. | Remind me this evening. | What to be reminded of? | As expected | Pass |

1. **Raising reminder**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC15.1 | Reminder raised on time. | Create a reminder for a few minutes. | Remind me that I have to do testing after 3 minutes. | Reminder rose. | As expected | Pass |

1. **Protecting reminder or note**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC16.1 | Protected note. | Create a note. Protect with password. | * Create a note that I lost to Rita. * Keep it secure. | * Note created. * Secured. | As expected | Pass |
| TC16.2 | Protected reminder. | Create a reminder. Protect with password. | * Remind me to go to bank tomorrow. * Keep it secure. | * Reminder created. * Secured. | As expected | Pass |

1. **Viewing protected note or reminder**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC17.1 | Protected note/reminder | View a protected note. | Show me note about last match. | It is protected. Password please. | As expected | Pass |
| TC17.2 | Not protected note/reminder. | View unprotected note. | Show me note about cookie store. | Opens the note. | As expected | Pass |

1. **Playing Songs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC18.1 | .mp3 available | Ask to sing a song. | Sing Old Mc Donald had a fall. | Plays the lullaby. | As expected | Pass |
| TC18.2 | .mp3 not available | Ask to sing song, you don’t have. | Sing Hush, Little Baby. | Let’s play on YouTube. | As expected | Pass |

1. **Reading documents and e-books.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC19.1 | .epub / .pdf available | Ask to read a story you have. | Read Snow White for me. | Starts reading. | As expected | Pass |
| TC19.2 | .epub / .pdf not available | Ask to read unavailable story. | Read The little Prince. | It will take a while to download. | As expected | Pass |

1. **Open a file**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC20.1 | Opens a file. | Ask to open a file. | Open myfile. | File opened. | As expected | Pass |
| TC20.2 | File not found. | Ask to open a file. | Open myfile2 | File does not exist | As expected | Pass |

1. **Find a file**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC21.1 | File found. | Ask to find a file. | Find myfile. | File Found in Local Disk E. | As expected | Pass |
| TC21.2 | File not found. | Ask to find a file. | Find myfile2 | File does not exist. | As expected | Pass |

1. **Play music or movies**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC22.1 | Plays any music/movie. | Ask to play songs. | Play music. | Plays music on shuffle. | As expected | Pass |
| TC22.2 | Play specific song/movie. | Ask to play specific song. | Play Coca-cola. | Plays that song. | As expected | Pass |
| TC22.3 | Song/movie not found | Ask to play specific song | Play Apna time aayega | We don’t have that. Lets YouTube. | As expected | Pass |

1. **Testing Sleep keywords**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case** | **Test Procedure** | **Test data** | **Expected Result** | **Actual Result** | **Remark** |
| TC23.1 | Check application stops on sleep word. | Speak any of the sleep words to stop program. | Goodbye Jia | Happy to have helped you. | As expected | Pass |
| TC23.2 | Check application does not stop without sleep word. | Speak anything in microphone other than sleep words. | What time is it? | 5:15 PM | As expected | Pass |

* 1. **MODIFICATIONS AND IMPROVEMENTS**

The main concept of testing is that you can never ensure that all swans are white. There always exist some defects. Also there is always scope for improvement. After executing the test cases, it was found that some aspects of the project needed rework and improvement. Besides, several issues were found which were fixed immediately. A few major improvements in the project are listed below:

* **Playing on YouTube**

There could be a scenario where the user asks to play a song and it is not found. In such case, the program is directed to visit YouTube and play the music on browser. This modification improved the project capability and user satisfaction.

* **Performance of follow-up queries**

When the user asks a similar question, the program would go again and search the new query. This is modified by first searching through the related questions and the side block of search page. This increased the performance.

* **Balanced humor and wisdom**

The assistance must not be completely robotic. And it must have a little humor, especially when having a conversation. But that should be balanced in a manner that it does not sound rude. If you say, “I am smart.” The assistant replies, “Yeah, little less than me.”

The end user tests showed that the majority of the responses given by Jia were appropriate. Over multiple sessions the program displayed the capability to learn from a users input and simulate a conversation on the relevant subject. This gives supporting evidence to the idea that a chatbot based virtual assistant is a feasible system when focusing on supporting individuals.

1. **RESULTS AND DISCUSSION**

After executing all the test cases and fixing the defects, regression and re-testing was also done to ensure reliability of the system. The modifications were also made and tested again. The test report is created after the test cases were successfully executed and met the exit criteria for acceptance of user.

* 1. **TEST REPORTS**

The results gathered from the tests with Jia have given evidence to support hypothesis. This was achieved by Jia showing the capability to meet the requirements discussed to support our initial hypothesis. Throughout the conversations with Jia during tests outputs given were appropriate and contextual to the input. This contextual awareness allowed the program to expand on the appropriate answer with data that is related to the subject.

The end result of this is an expanded response which in turn goes beyond appropriate and becomes useful in a meaningful way to the user. Throughout the analysis and testing of the program, a number of potential improvements for the ideal virtual assistant have been identified and improved. The assistant holds a small preset library containing basic conversations (small talk, greetings) and provocations to continue the conversation (questions, curiosity).

The patterns formed between data helps each program to react differently to different inputs. When this is combined with the local storage of data libraries it becomes a personalization of the program. The programs function will be uniquely different as it understands the data in relation to the user.

This virtualassistant is capable of emulating both *logical* and *emotional interest* in the interaction with its user. Bringing these ideas and technologies together would lead the way for a truly useful virtual personal assistant. Logical aspects make the system usable, however the combination of logical with emotional aspects make such a system worthwhile, and so ultimately useful for the user.

* 1. **USER DOCUMENTATION**

1. **CONCLUSIONS**

* 1. **CONCLUSION**

Virtual assistant software improves user productivity by managing routine tasks of the user and by providing information from online sources to the user. Difficulties associated with capturing human intelligence in models that can be used to drive the virtual assistant have been one of the primary bottlenecks in building such assistants. There was an expectation that as the virtual assistant gained more knowledge about the user, ad funded recommendations could be introduced that were genuinely useful to the user, in a similar manner to paid for search results.

The Virtual Assistant proves useful for the people who don’t have much knowledge to use computers. The people who are not familiar getting around on their own on a computer can use this application. Want to open and close an application, file? Just say it as you would to a person, it will do the work for you. It is the solution for the people who are willing to make a computer as their real assistant!

* 1. **LIMITATIONS OF THE SYSTEM**

There are some limitations for the current system to which solutions can be provided as a future development:

* Stronger NLP to capture user input unambiguously
* Leaning from user inputs and actions
* Integrating with social networks
* Making online payments
  1. **FUTURE SCOPE OF THE PROJECT**

As for other future developments, the following can be included in the project:

* Adding reasoning capabilities
* Optimizing algorithms
* Logging the system activities
* Connecting with IoT

**REFERENCE AND BIBLIOGRAPHY**

* Websites referred
  + - www.stackoverflow.com
    - www.pythonprogramming.net
    - www.codecademy.com
    - www.tutorialspoint.com
    - www.google.co.in
* Books referred
  + - Python Programming - Kiran Gurbani
    - Learning Python - Mark Lutz
* YouTube Channels referred
  + - CS Dojo
    - edureka!
* Documents referred
  + - Designing Personal Assistant Software for Task Management using Semantic Web Technologies and Knowledge Databases

- Purushotham Botla

* + - Python code for Artificial Intelligence: Foundations of Computational Agents

- David L. Poole and Alan K. Mackworth

THANK YOU