

# **Virtual Personal Assistant using Natural Language Processing and Machine Learning**

Project report submitted in partial fulfillment of the requirement for  
the degree of Bachelor of Technology

in

**Computer Science and Engineering/Information Technology**

By

Faiz Ali (161252)

Kanishk Sood (161243)

Under the supervision of

Dr. Amol Vasudeva

to



Department of Computer Science & Engineering and Information  
Technology

**Jaypee University of Information Technology Waknaghat,  
Solani-173234, Himachal Pradesh**

## Candidate's Declaration

I hereby declare that the work presented in this report entitled **Virtual Personal Assistant using Natural Language Processing and Machine Learning** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2015 to December 2015 under the supervision of **Dr. Amol Vasudeva** (Assistant Professor, Department of Computer Science & Engineering).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.



(Student Signature)

Faiz Ali, 161252

This is to certify that the above statement made by the candidate is true to the best of my knowledge.



(Supervisor Signature)

Dr. Amol Vasudeva

Assistant Professor

Department of Computer Science & Engineering

02/12/2019

## **Acknowledgment**

I would like to express my greatest gratitude to the people who have helped & supported us throughout my project. I am grateful to my mentor **Dr. Amol Vasudeva** for his continuous support for the project, from initial advice & contacts in the early stages of conceptual inception & through ongoing advice & encouragement to this day.

A special thank of us to our group members who helped each other in completing the project & exchanged their interesting ideas, thoughts & made this project easy and accurate.

# CONTENTS

	Page No.
Certificate	i
Acknowledgement	ii
List of Abbreviations	iv
List of Figures	v
List of Graphs	vi
List of Tables	vii
Abstract	viii
<b>CHAPTER 1: INTRODUCTION</b>	<b>1</b>
1.1 Introduction	1
1.2 Problem Statement	10
1.3 Objective	10
1.4 Methodology	11
1.5 Organization	12
<b>CHAPTER 2: LITERATURE REVIEW</b>	<b>13</b>
<b>CHAPTER 3: SYSTEM DEVELOPMENT</b>	<b>33</b>
<b>CHAPTER 4: PERFORMANCE ANALYSIS</b>	<b>41</b>

<b>CHAPTER 5: CONCLUSION</b>	50
5.1 Conclusion	50
5.2 Future Scope	52
5.3 Application Contribution	53
<b>REFERENCES</b>	54

## **List of Abbreviations**

- **VPA**- Virtual Personal Assistant
- **AIML**- Artificial Intelligence Markup/Model Language
- **AI**- Artificial Intelligence
- **ML**- Machine Learning
- **HTTP**- Hyper Text Transfer Protocol
- **XML**- Extensible Markup Language
- **ALICE**- Artificial Linguistic Computer Entity
- **NLP**- Natural Language Processing

## List of Figures

<b>Figure No.</b>	<b>Caption</b>	<b>Page No.</b>
Figure 1.1	Dialogflow	8
Figure 1.1	Firebase	9
Figure 2.1	A sample of chatting with ALICE	14
Figure 2.2	Effectiveness of various tutoring methods	21
Figure 2.3	GIFT adaptive tutoring testbed	22
Figure 2.4	A human-agent dialogue during the process of making a business decision	24
Figure 2.5	Two architectures of dialogue systems	25
Figure 3.1	System Architecture	35
Figure 3.2	Text Based Chatbot	36
Figure 3.3	Voice based Chatbot	39
Figure 4.1	App Icon	41
Figure 4.2	Home Screen	42
Figure 4.3.1	Login Screen	43
Figure 4.3.2	Screen after login	44
Figure 4.4.1	Conersation with bot	45
Figure 4.4.2	Conersation with bot	46
Figure 4.4.3	Failed conversation with bot	47
Figure 4.5.1	Listening Voice	48
Figure 4.5.2	After Listening	49
Figure 5.2.1	Different types of bot	53

## List of Graphs

<b>Graph No.</b>	<b>Caption</b>	<b>Page No.</b>
Graph 2a	Classification	17
Graph 2b	Regressing	17



## **Abstract**

A VPA is a conversational software where a program is designed to simulate an intelligent conversation. It can take user input in many forms like text, voice. For this purpose, many open source platforms are available. One of them is Artificial Intelligence Markup Language (AIML) which is derived from Extensible Markup Language (XML) which is used to build up a conversational agent (chatbot) artificially. In this project, we have used Firebase Cloud Service, Google Speech API which is an interpreter for the generation of the responses of users input. We have used this method for developing an android application chatbot which will interact with user using text and voice responses.

## **Chapter 1: Introduction**

### **1.1 Introduction**

A chatbot is a PC program that can banter with people utilizing man-made consciousness in informing stages. The objective of the undertaking is to include a chatbot highlight and API for VPA. exchange gatherings, online journals, wikis and so forth. Yioop gives all the fundamental highlights of web search entry. It has its own record the executives framework with the capacity to set up bunches that have dialogs sheets. Gatherings are assortments of clients that approach a gathering feed. The client who makes a bunch is set as the underlying gathering proprietor. Posts are assembled by string in a gathering with latest action at the top. The chatbot API for VPA will enable engineers to make new chatbots, fueled by rules or man-made brainpower, that can connect like a human with clients in a bunches feed page. Model chatbots that can be created with this API is climate or book flight chatbots. Over recent years, informing applications have gotten progressively well known than Social systems administration destinations. Individuals are utilizing informing applications nowadays, for example, Facebook Messenger, Skype and so on. This is making different organizations accessible on informing stages prompts proactive communication with clients about their items. To communicate on such informing stages with numerous clients, the organizations can compose a Android application that can banter like a human which is known as a chatbot.

Chatbots come in two sorts:

- Limited arrangement of rules
- Machine learning

Chatbot that utilizes constrained arrangement of rules. This sort of bots are restricted to set of writings or directions. They have capacity to react as it were to those writings or directions. On the off chance that client asks something other than what's expected or other than the arrangement of writings or directions which are characterized to the bot, it would not react as wanted since it doesn't comprehend or it has not prepared what client inquired. These bots are not shrewd when contrasted with other sort of bots.

### **Chatbot**

AI chatbots works utilizing man-made reasoning. Client need not to be increasingly explicit while chatting with a bot since it can comprehend the regular language, not just directions. This sort of bots show signs of improvement or more astute as it gains from past discussions it had with individuals. Here is a straightforward model which show how they work.

Coming up next is a discussion between a human and a chatbot:

Human: "I need a departure from San Jose to New York."

Bot: "Sure! When might you want to travel?"

Human: "From Dec 20, 2016 to Jan 28, 2017."

Bot: "Amazing! Searching for flights."

## **Artificial Intelligence(AI )**

John McCarthy, an American computer researcher, instituted artificial intelligence at The Dartmouth Conference in 1956 where the control was conceived. Today, it's a paragliding concept that envelops anything from computerization of mechanical procedures to autonomy in real operation. It has acquired notable quality as of late due to enormous information to some extent, or an increase in the speed, size and assortment of information organizations is currently being collected. Computer-based intelligence can perform errands such as distinguishing designs more productively in the information than people, empowering organizations to acquire knowledge from their information.

## **Types of AI**

Artificial intelligence can be classified in a variety of ways but here are two models. The first AI groups frameworks either as frail AI or as solid AI. Powerless AI is an AI system that is designed and trained for a particular undertaking, otherwise known as minimal computer-based knowledge. For example, virtual individual co-workers, Apple's Siri, are a type of frail AI. Solid AI, otherwise known as fake general insight, is an AI framework with a summary of human subjective capabilities, so if a new undertaking is given, it will have sufficient insight to find a response. The Turing Test, developed in 1950 by mathematician Alan Turing, is a technique used to decide whether a PC can have a similar outlook as a person

His classifications are as per the following:

**Type 1: Reactive machines.**

One pattern is Deep Blue, the IBM chess program that over the 1990s defeated Garry Kasparov. Profound Blue is capable of distinguishing pieces on the chess board and making expectations, yet has no memory and can not use past encounters to advise future ones. It breaks down potential movements-its very own and its opponent-and picks the most important move. Others were intended for a limited purpose and can only be applied to other circumstances with considerable effort

**Type 2: Limited memory.**

These AI frameworks may be used to illuminate future choices using past encounters. A portion of the decision-making capability was organized along these lines in self-sufficient vehicles. Perceptions used to educate activities that occur not long ago, such as a vehicle that has moved to a different lane. Those perceptions are not always put away.

**Type 3: Theory of psyche.**

This is a term used in brain science. It alludes to the understanding that others have their own convictions, wills and goals that affect their choices. This kind of AI still doesn't exist.

**Type 4: Self-mindfulness**

AI structures in this category have a sense of self, have consciousness. Awareness machines understand their current state and can use the data to collect what others feel.

This type of AI still isn't available.

## **AI applications**

### AI in medicinal services.

The greatest wagers are on improving patient results and decreasing expenses.

Organizations apply AI to make decisions easier and faster than people do. IBM Watson is a standout among other advances in known medicinal services. It understands common language, and is suitable for answering questions. The framework mines silent information and other available sources of information to shape a theory, which it presents with a sure scoring construction at that point. Other AI applications include chatbots, a PC program used online to address questions and assist customers, help plan follow-up arrangements or assist patients through the charging procedure, and virtual wellness partners that provide fundamental restorative criticism.

### AI for Company

Computerization of the mechanical process is extended to extremely slow errands usually carried out by men. AI calculations are incorporated into the investigation and CRM stages to reveal to all the more likely serving clients data about the most proficient method.

Chatbots were consolidated into sites to help clients swiftly. Computerization of the mechanical process is extended to extremely slow errands usually carried out by men. AI calculations are incorporated into the investigation and CRM stages to reveal to all the more likely serving clients data about the most proficient method. Chatbots were consolidated into sites to help clients swiftly. AI could change where and how understudies adapt, supplanting some teachers. AI in money maybe in any case. For example, AI applied to individual fund applications, Mint or Turbo Tax, overturns establishments related to the money. For example, these applications could collect individual information and provide money-related guidance. Various projects were applied on the way to buying a home. Today, programming is performing a significant part of the lawful exchange on

Wall Street. AI. In law the process of discovery, filtering through files, is always overpowering for men. Robotisation of this technique is a superior time consumption and an increasingly competent technique. Additionally, new companies are developing inquiry and reaction PC partners who can filter inquiries by looking at a database linked scientific classification and philosophy.

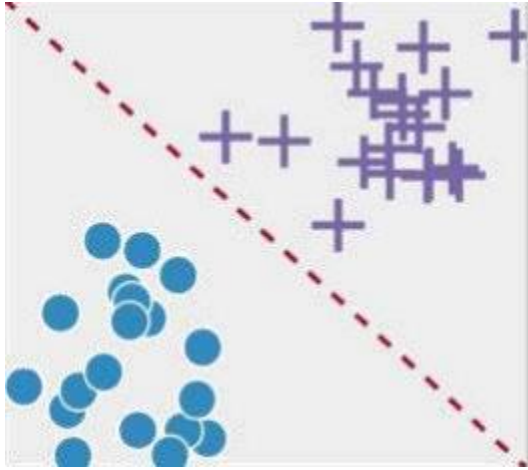
### AI in assembling

This is a zone that was at the forefront of consolidating robots into working process.

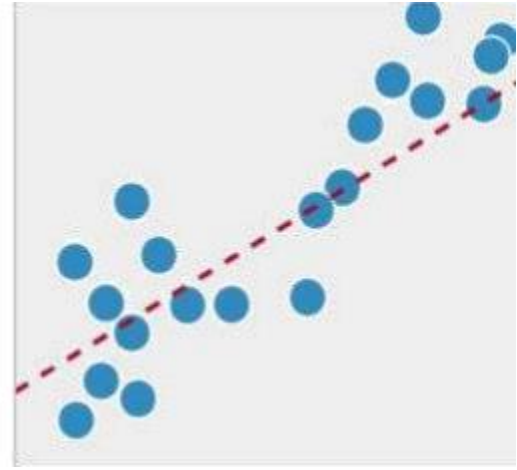
Robots used to perform single undertakings and were isolated from humans, yet they were driven by innovation.

### **Machine Learning**

ML is a software development area that "allows PCs to learn without being unambiguously customised." The recipes parameter is determined from the information, as opposed to software engineer characterization. Classification and Regression are two most regular uses of ML. Classification aims to sort different types of information while Regression intends to figure out how to portray them. Fundamental ML program will have two phases, fitting and predicting. A huge set (at any rate a huge number of) information will be given to the program in the fitting stage. The program will try to modify its parameter depending on some factual models, in order to make it "fit" the information as best as possible. In the early stage, fundamental ML program will have two phases, fitting and predicting. A huge set (at any rate a huge number of) information will be given to the program in the fitting stage. The program will try to modify its parameter depending on some factual models, in order to make it "fit" the information as best as possible. In the early stage, The software is supposed to include additional details based on the criteria that it has clearly defined. The example behind this element can be familiarized with a well-characterized ML program and forecast as needed



Graph 2a: Classification



Graph 2b: Regressing

## **Natural Language Processing (NLP )**

Natural Language Processing ( NLP) is a study designed to give machines a chance to understand human dialect. Without NLP, human-language sentences are merely a progression of inane images into machines. Machines don't see the words, nor do they understand the syntax. NLP can be seen as a "interpreter" who will give Computer fair data on human dialect understanding. Natural Language Processing ( NLP) is a study designed to give machines a chance to understand human dialect. Without NLP, human-language sentences are merely a progression of inane images into machines. Machines don't see the words, nor do they understand the syntax. NLP can be seen as a "interpreter" who will give Computer fair data on human dialect understanding.



## Dialogflow

Dialogflow is NLP platform used to make conversation apps in many languages and multiple platforms. It enables developers to create text-based and voice conversation interfaces for responding to user queries.

It is the most common conversational tool used to build bots.

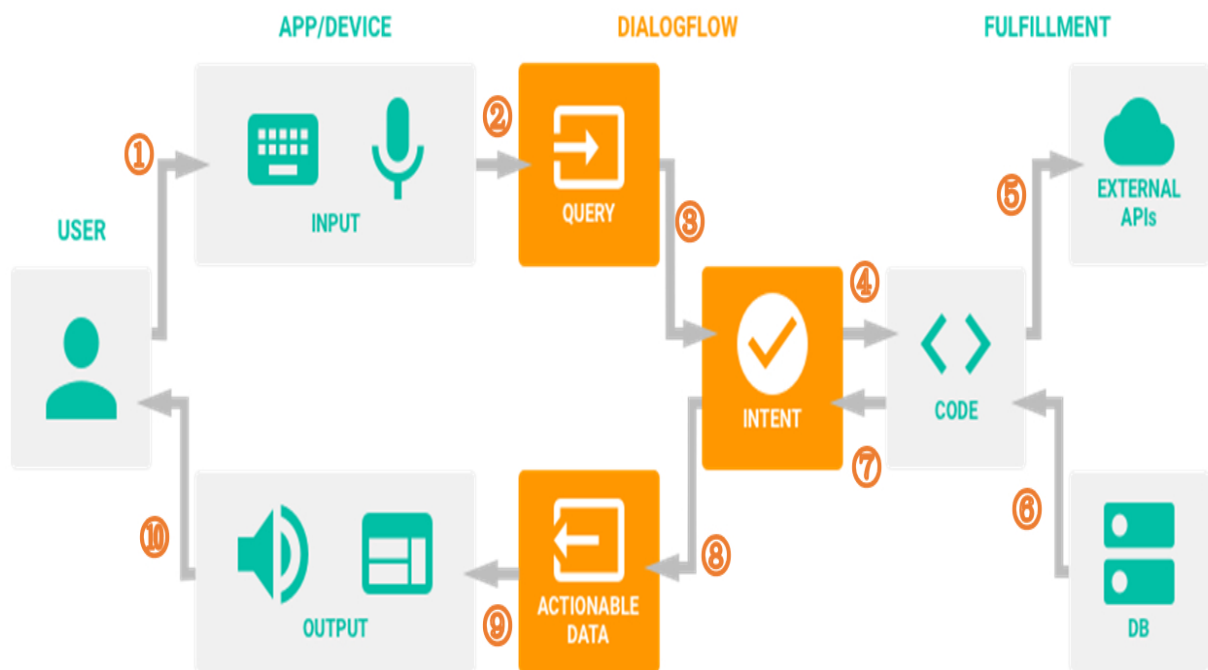


Figure 1.1: Dialogflow

## Firestore

Firestore is a Backend service. It provides users with a bunch of tools and services to help them develop applications, grow their user base. It is built on Google's infrastructure.

It is a NoSQL database program, it stores data in JSON-like documents.



Figure 1.2: Firestore

## **1.2 Problem Statement**

The project is based on the central idea of creating VPA on android platform that would perform actions according to user commands which may be voice or text. It would convert voice to text and produce a response and then turn that text response to digital voice response. It should be interactive and should give friendly responses to user. It should also perform basic mobile functions based on commands such as making calls, setting alarms, reminders, etc.

## **1.3 Objective**

The purpose of this project is to make user's mobile experience easier and more convenient. It would make it easier to operate other apps through the VPA and would also perform basic functions of the phone like making calls, setting alarms, etc. It shows the strength of chatbots, and how they can be an choice, unlike using an application or even a web. The chatbots should be anything but difficult to use, should respond in a convenient manner and should be easy to comprehend. The bots should make customer interaction as quick and easy as possible to ensure customers do not waste their time and get what they need. It shows the strength of chatbots, and how they can be an choice, unlike using an application or even a web. The chatbots should be anything but difficult to use, should respond in a convenient manner and should be easy to comprehend. The bots should make customer interaction as quick and easy as possible to ensure customers do not waste their time and get what they need. With informing stages being the most utilized kind of utilization on the planet, organizations will hope to exploit this and begin to build up their own chatbots to work alongside their online networking pages. For instance, an individual calling an eatery to perceive what time they open at or what is the uncommon today, the client can basically message the page on Facebook and the bot will react in like manner. This saves time for

genuine representatives to do other work and permits the chatbot to deal with the straightforward undertakings. Since clients will as of now have an informing application introduced on their cell phone, there is no compelling reason to download a different application to utilize the chatbot. This can turn a great deal of clients away as these days there is a plenty of utilizations available and most clients will be exhausted of downloading an application that they may just utilize a few times.

#### **1.4 Methodology**

The incremental model is the most reasonable improvement technique to execute for this task. The adaptability of the steady model makes it perfect for this task as it is likely new necessities will be recognized during the later phases of improvement what's more, every iterative form makes it simple to actualize new prerequisites all through the improvement procedure.

This methodology begins with taking input from clients in android application. We ask the client to enter the query for the chatbot, and then read the entered value. Send the value at that point to our facilitated program and get reply from the server. If the reaction of bot is invalid at that point, set the reaction to the error string and show it to the client. Repeat a similar task for continued conversation with bot.

## 1.5 Organization

This report consists of five chapters with a detail explanation of every aspect of this project.

**Chapter 1:** This chapter is the formal introduction of the project. Here we are introducing the reader to various terminology of the project and we are also discussing the problem or motivation behind this project. Along with this, we're also starting our objective and methodology we will be using to execute the project.

**Chapter 2:** This chapter consists of various researches related to our project that was done in the recent past. Here, we are more emphasizing on the methodology that was used by the papers. Along with this, we're also studying the outcome of their respective projects.

**Chapter 3:** In this chapter, we will go through various stages of our development and will learn about the design and algorithm implementation. Here we will also Develop the model and try to represent it from various aspects like analytical, computational, experimental, mathematical and statistical.

**Chapter 4:** In this chapter, we will go through the performance analysis of our project.

**Chapter 5:** This would be our last chapter, here we will discuss the outcome of our project and also analyze our result. Along with this, we will also discuss the future scope of the project and any upgrades that we can implement in the coming future. Also, we will discuss some applications where the system can be helpful.

## Chapter 2: LITERATURE SURVEY

In this section, we discuss some recent research, trends and study in the field of Virtual Personal Assistant.

### **Paper 1: “Bayan Abu Shawar, Eric Atwell-Chatbots: Are they Really Useful?”**

Chatbots are PC programs which use common languages to communicate with clients. This innovation began in the 1960s; the point was to check whether chatbot frameworks were capable of tricking clients into being genuine people. In any case, chatbot systems are not only used to copy human debate and engage clients. In this paper they explored various applications where chatbots could be useful, such as training, retrieving data, business, and web-based business. This paper shows a range of chatbots with useful applications, including some dependent on the ALICE / AIML design.

In this paper, They showed viable chatbot applications, which indicated chatbots helping work area apps, programmed telephone noting systems, instructional resources, business and online business are found in daily life.

### **The ALICE Chatbot System**

A.L.I.C.E. is the Artificial Linguistic Internet Computer Entity which Wallace initially implemented in 1995. Alice 's discussion design information is put away in AIML files. AIML, or mark-up language for artificial intelligence, is an Extensible Mark-up Language (XML) subordinate. It was founded from 1995 on as a free programming network for empowering individuals, Input discourse concept details into A.L.I.C.E. open-source programming innovation chatbots.

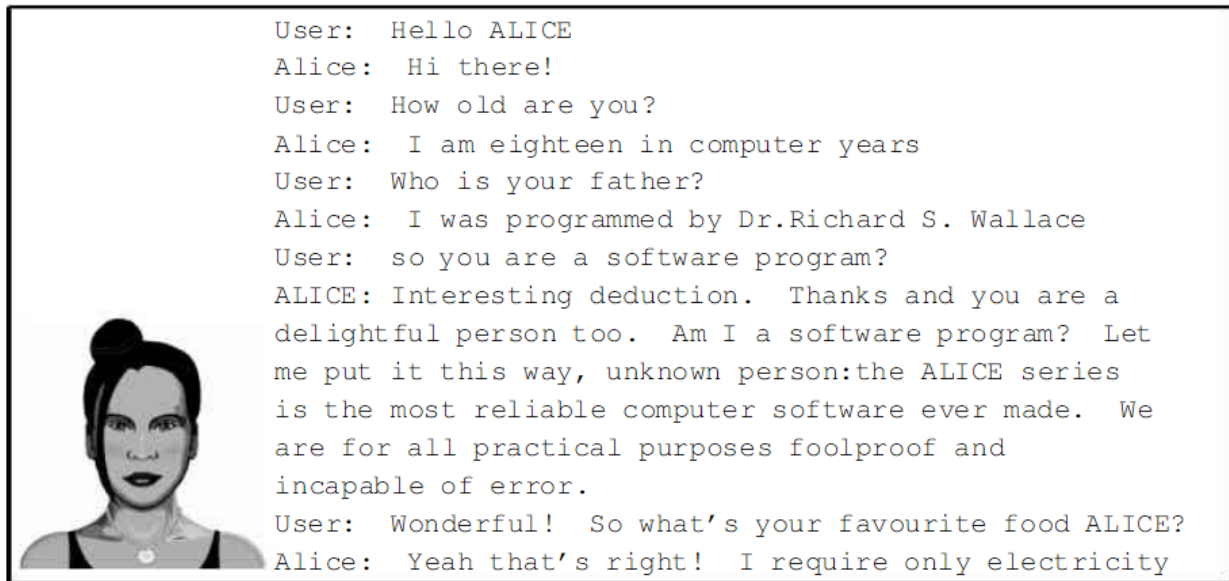


Figure 2.1: A sample of chatting with ALICE

## Conclusion

They reviewed some chatbot frameworks that prevail in spaces such as training, data recovery, business, web-based business, as well as diversion, down to earth. You could later envision Chatterbots as talking books for children, Chatter bots for unknown dialect guidance and Chatterbots as a rule. " (Wallace et al., 2003). Be that as it may, in the training space that "the educator is the spine in the instructing procedure. Innovation like PC variable based math frameworks, sight and sound introductions or 'chatbots' can fill in as amplifiers yet not supplant a decent guide".

Broadly speaking, the point of chatbot planners should be: manufacturing appliances that help individuals, encouraging their work, and communicating with PCs using regular language; yet not completely supplanting the human job, or impersonating human discussion to the full. Finally, as Colby says, "We need not accept human-human discussion as the best quality level for conversational trades. On the off chance that one had an ideal reproduction of a human acquainted, at that point it would be human-human discussion and not human PC discussion with its occasionally odd but rather relevant properties."

## **Paper 2: Computational Linguistics and Natural Language Processing Jun'ichi Tsujii**

### **Abstract**

Researches in CL and NLP have been progressively separated from one another. Experimental systems in NLP show great exhibitions in certain undertakings when enormous measure of information (with comment) are accessible. However, in order to adapt these techniques easily to new text types or domains, or to apply comparable methods to more unforeseeable undertakings, such as text entailment than POS taggers, parsers and so on., normal comprehension of language is required. Designing procedures must be supported by logical comprehension. In this article, we'll talk about how to re-incorporate these two research disciplines, taking language in CL and parsing in NLP for instance. Research findings from our parsing gathering will be presented to demonstrate how language structure in CL is used as a basis for a parser.

### **About**

Computational Linguistics and Natural Language Processing, have frequently been utilized conversely. Be that as it may, these two terms speak to two distinct floods of research that stress various parts of our field. For example, while inquiring about the language structure and its formalities in CL (Comp. Linguistic) and NLP parsing research are closely related, their objectives are quite different. On the one hand, CL scientists have focused on discovering how surface series of words compare deliberately with their implications (in a compositional manner) and have been keen to create formalisms by which correspondence is depicted. Then again, those in NLP are keen on progressively commonsense building issues associated with handling regular dialects by PC, for example, efficient calculations For a program (parser) that processes both the structure and context of a given sentence. Whereas some parsers used in NLP are based on a



language structure in CL, with the goal for them to be commonsense, they ought not exclusively be efficient and vigorous yet in addition have the option to pick the most conceivable elucidation of a sentence among numerous conceivable translations. Probabilistic demonstrating, which has been widely concentrated of late, has been effective in settling on such options as well as enhancing parser quality and heartburn. Despite the fact that probabilistic models are beyond the size of CL formalism punctuation studies, the two research streams have slowly become isolated from each other. In this paper they talk about our exploration technique to reassociate these two floods of research to develop another more extensive field in which look into on language structure portrayal and handling are appropriately incorporated. Specifically, they contend that portrayal of language structure and handling dependent on it ought to be plainly recognized. Straight forward utilization of a sentence structure in CL to parsers in NLP would not be ripe as we anticipated. Simultaneously, measurable displaying without legitimate phonetic hypotheses would be as vain as CL without appropriate thought of handling issues. They talk about a few examines [1–9], which they have been occupied with. They will reveal insight into the intriguing connection between portrayal of sentence structure and preparing.

## **Document- Dialogflow Documentation ann how we can use it**

Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on. Using Dialogflow, you can provide new and engaging ways for users to interact with your product.

Dialogflow can analyze multiple types of input from your customers, including text or audio inputs (like from a phone or voice recording). It can also respond to your customers in a couple of ways, either through text or with synthetic speech.

---

## Agents

A Dialogflow [agent](#) is a virtual agent that handles conversations with your end-users. It is a natural language understanding module that understands the nuances of human language. Dialogflow translates end-user text or audio during a conversation to structured data that your apps and services can understand. You design and build a Dialogflow agent to handle the types of conversations required for your system.

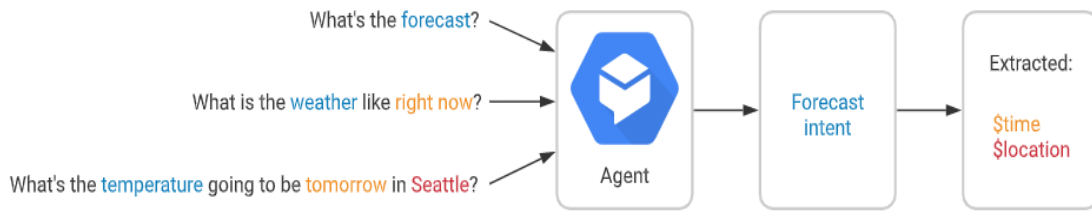
A Dialogflow agent is similar to a human call center agent. You train them both to handle expected conversation scenarios, and your training does not need to be overly explicit.

## Intents

An [intent](#) categorizes an end-user's intention for one conversation turn. For each agent, you define many intents, where your combined intents can handle a complete conversation. When an end-user writes or says something, referred to as an *end-user expression*, Dialogflow matches the end-user expression to the best intent in your agent. Matching an intent is also known as *intent classification*.

For example, you could create a weather agent that recognizes and responds to end-user questions about the weather. You would likely define an intent for questions about the weather forecast. If an end-user says "What's the forecast?", Dialogflow would match that end-user expression to the forecast intent. You can also define your intent to extract useful information from the end-user expression, like a time or location for the desired weather forecast. This extracted data is important for your system to perform a weather query for the end-user.

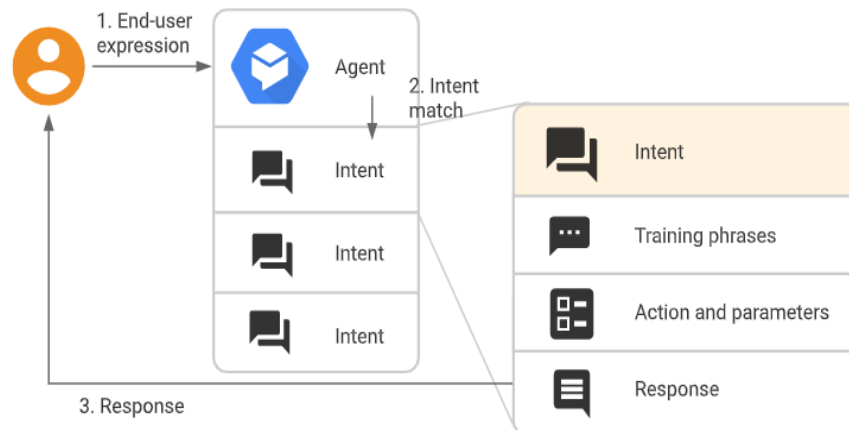
---



A basic intent contains the following:

- **Training phrases:** These are example phrases for what end-users might say. When an end-user expression resembles one of these phrases, Dialogflow matches the intent. You don't have to define every possible example, because Dialogflow's built-in machine learning expands on your list with other, similar phrases.
- **Action:** You can define an action for each intent. When an intent is matched, Dialogflow provides the action to your system, and you can use the action to trigger certain actions defined in your system.
- **Parameters:** When an intent is matched at runtime, Dialogflow provides the extracted values from the end-user expression as *parameters*. Each parameter has a type, called the **entity type**, which dictates exactly how the data is extracted. Unlike raw end-user input, parameters are structured data that can easily be used to perform some logic or generate responses.
- **Responses:** You define text, speech, or visual responses to return to the end-user. These may provide the end-user with answers, ask the end-user for more information, or terminate the conversation.

The following diagram shows the basic flow for intent matching and responding to the end-user:



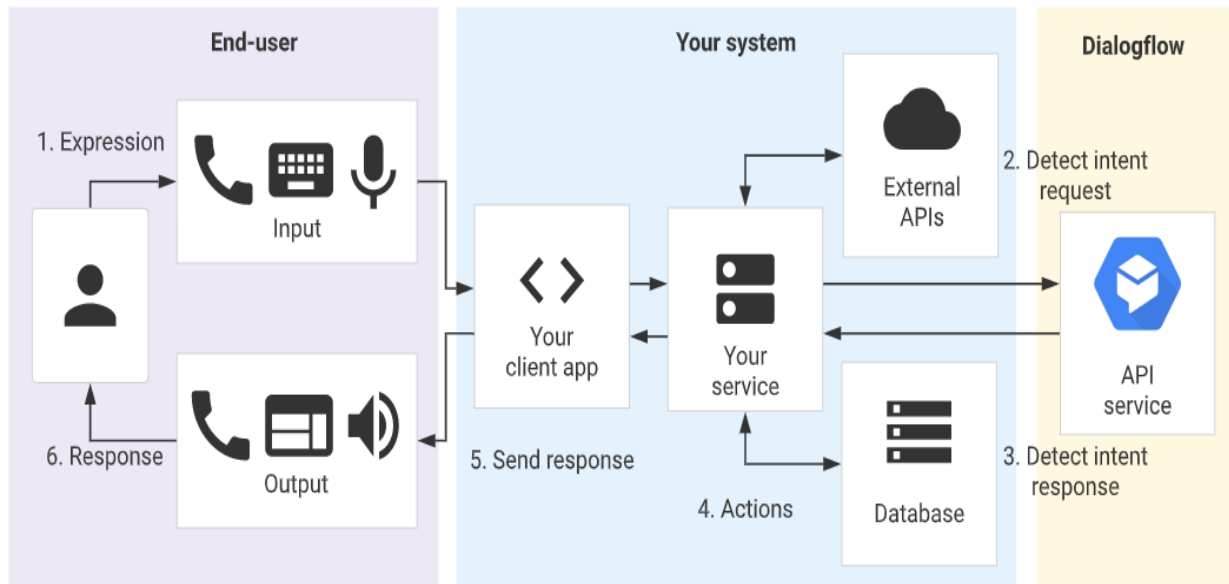
## Entities

Each intent **parameter** has a type, called the **entity type**, which dictates exactly how data from an end-user expression is extracted.

Dialogflow provides predefined **system entities** that can match many common types of data. For example, there are system entities for matching dates, times, colors, email addresses, and so on. You can also create your own **custom entities** for matching custom data. For example, you could define a *vegetable* entity that can match the types of vegetables available for purchase with a grocery store agent.

## User interactions with the API

If you are not using one of the [integration](#) options, you must write code that directly interacts with the end-user. You must also directly [interact with Dialogflow's API](#) for each conversational turn to send end-user expressions and receive intent matches. The following diagram shows the processing flow when interacting with the API.



1. The end-user types or speaks an expression.
  2. Your service sends this end-user expression to Dialogflow in a detect intent request message.
  3. Dialogflow sends a detect intent response message to your service. This message contains information about the matched intent, the action, the parameters, and the response defined for the intent.
  4. Your service performs actions as needed, like database queries or external API calls.
- 
5. Your service sends a response to the end-user.
  6. The end-user sees or hears the response.
-

## **Paper 3: Adaptive Intelligent Tutoring System (ITS) Research in Support of the Army**

### **Learning Model**

Current Army principles for preparing and training are bunch guidance and classroom training otherwise called one-to-numerous guidance. As of late, the Army has set noteworthy accentuation Techniques on self-regulated learning (SRL) to expand institutional preparation. Soldiers will be largely responsible for their own learning according to ALM. It has been shown that holistic individual coaching is ultimately more convincing than one-to-numerous advice, but it's not down to earth. An alternative to one-to-one human tutoring is computational tutoring using ITSs, Proven to be effective in promoting individual learning in static, simple, well-defined domains (e.g., math). High compositional expenses and constrained barriers to adaptation must be addressed to be reasonable. This diagram depicts a technique for resolving key difficulties with ITS plans and for increasing SRL skylines. Research is needed to: reduce the cost / competency of creating ITSs, Improve ITS adaptability and expand ITS domains to support more complex and undefined domains that fit the Army's operational mission. The related concept of undertakings in the Army involves mentoring of squads and specific classes. This report aims to inform and educate stakeholders and to focus potential collaborators on relevant issues within the research area of adaptive tutoring.

### **BACKGROUND**

Study hall preparing has been commonly centered around procuring information and applying information in intermediaries for live preparing conditions (e.g., work area virtual recreations, genuine games). Little gathering guidance in live situations has likewise been utilized to evaluate utilization of information and advancement of aptitudes. A standard criticism system for Army preparing is the after-activity audit (AAR) where noteworthy

choice focuses and activities are caught for little gathering dialog directed after the fruition of a preparation occasion.

As of late, the Army has put huge accentuation on self-managed learning techniques to expand institutional training. One-to-one human coaching has been shown to be substantially more viable than one-to-numerous instructional strategies ( e.g., study hall guidance: Bloom, 1984; Van Lehn, 2011), but having one human mentor for each soldier in the army is not reasonable or moderate.

Unlike coordinated human coaching, a handy option is balanced mentoring strategies based on a PC , for example ITSs. Figure 1 looks at the impact sizes of various coaching techniques. When compared with custom homeroom planning, The techniques shift from 0.8 to 1.05. These are promising results and compare with an expansion of a letter assessment or greater improvement over different techniques

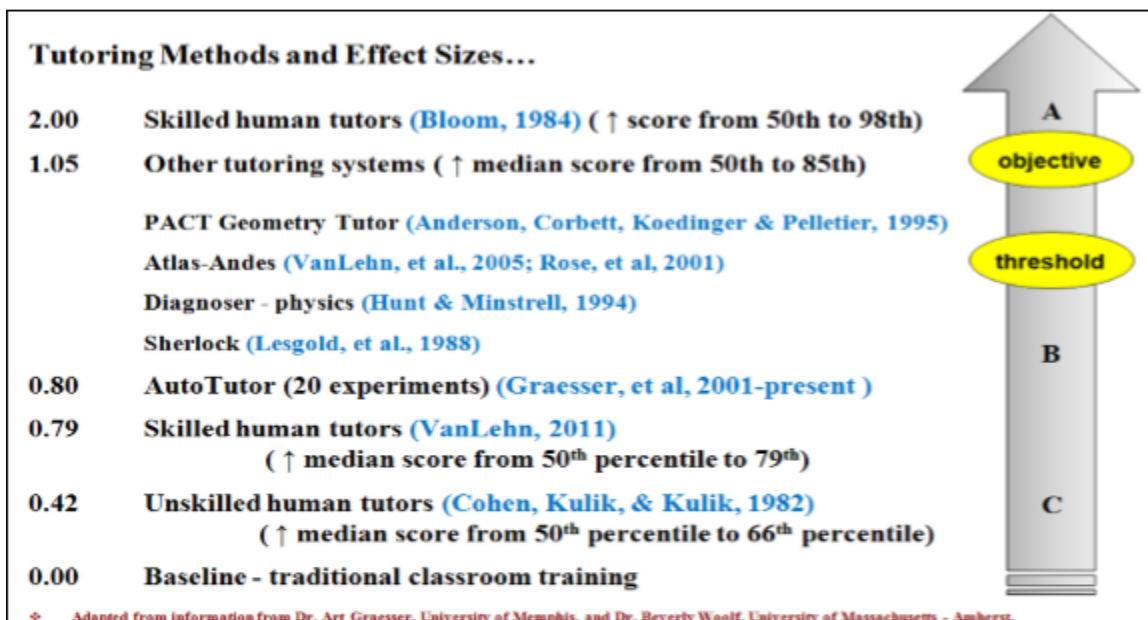


Figure 2.2. Effectiveness of various tutoring methods.

## Analysis

A fundamental component of research involving the development of models is the approval of such models to experimentally determine their effect on results factors ( e.g., learning) and their capacity for cross-sectional speculation over areas. To this end, a testbed to help an organized assessment and correlation of ITS instruments and strategies is basic. The versatile mentoring testbed, appeared in figure 5, was adjusted from testbed strategy.

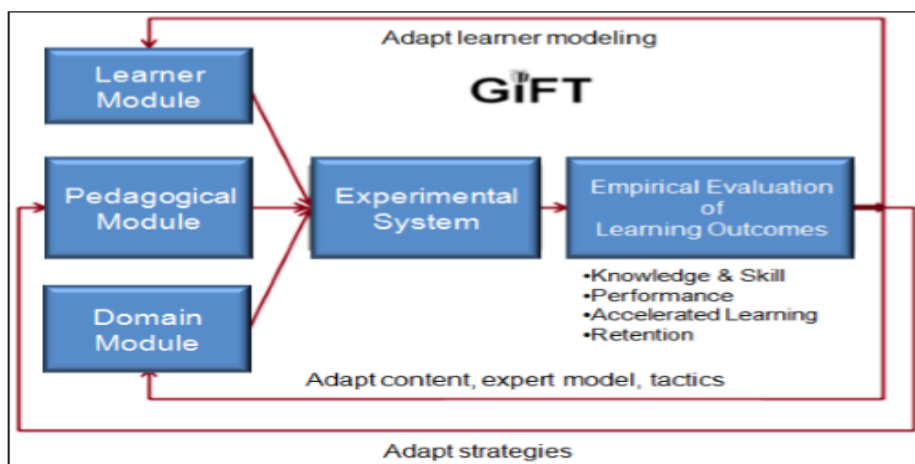


Figure 2.3. GIFT adaptive tutoring testbed.

The versatile mentoring testbed will be utilized to survey impact size through: similar investigations, ablative examinations, developmental investigations and summative investigations. Similar investigations use impact size (explicitly supreme impact size) to separate the general quality of strategies in affecting results (e.g., learning) when contrasted with a base case. On account of versatile coaching, the conventional base case is one-to-numerous homeroom guidance. The adequacy of different coaching techniques in well-characterized, static spaces is represented in figure 1. The extension of mentoring spaces to line up with preparing errands for military tasks (talked about in area 4.3 of this report) requires extra adequacy assessments to help plan choices and best practices for military coaching frameworks.

## **Paper 4: Plant Disease Detection and its Solution Using Image Image Classification.**

The present paper reviews neural ways to deal with conversational AI generated over the last few years. In this, conversational systems are grouped into three classifications: (1) questioning of experts, (2) task-situated communication operators, and (3) chatbots. They have presented a survey of the best in class neural methodologies for each class and speak about progress faced with challenges, using common structures and models as contextual analyzes.

### **What Kinds of Problems?**

Illustration. 2.4 Exhibits a human-agent dialog during the business decision-making process.

The example shows the kinds of problems which a dialog system is expected to solve:

- Question answering: The operator needs to provide compact, straightforward responses to customer inquiries depending on rich information from various sources including content assortments, such as web reports and pre-compiled information bases, such as dealings and data sets promotion, as shown in turns 3 to 5 in Figure. 1.1.1..
- Task completion: The specialist must carry out customer assignments ranging from café bookings to planning meetings (e.g. turns 6 to 7 in Fig. 2.4) and work arrangement excursions..
- Social chat: The specialist must carry out customer assignments ranging from café bookings to planning meetings (e.g. turns 6 to 7 in Fig. 1.1) and work arrangement excursions..

Imagine that a lot of operators, otherwise called bots, can practice the above-mentioned discourse, each of which is intended to illuminate a specific type of undertaking, e.g. Bots on QA, bots on mission fulfillment etc. These bots can be collected in two classifications,



task location and discussion, depending on whether the exchange is aimed at helping clients perform specific tasks e.g. Acquire an enquiry response or book a gathering. For example, Amazon Alexa, Google Home, and Microsoft Cortana, the vast majority of well-known individual associates in the present market are task-arranged bots. These can only deal, for example, with generally straightforward undertakings that detail the climate and mention tunes. Microsoft is one case of a babble bot.

Turn	Dialogue
1	<i>usr</i> : Good morning!
	<i>agt</i> : Good morning, I hope you had a great weekend.
2	<i>usr</i> : not really.
	<i>agt</i> : You sound depressed. What can I do for you?
3	<i>usr</i> : I'd like to know where sales are lagging behind our forecast?
	<i>agt</i> : The worst region is [country], where sales are 15% below projections.
4	<i>usr</i> : Do you know why?
	<i>agt</i> : The forecast for [product] growth was overly optimistic.
5	<i>usr</i> : How can we turn this around?
	<i>agt</i> : Here are the 10 customers in [country] with the most growth potential, per our CRM model.
6	<i>usr</i> : Can you set up a meeting with the CTO of [company]?
	<i>agt</i> : Yes, I've set up a meeting with [person name] for next month when you are in [location].
7	<i>usr</i> : Thanks!

Figure 2.4: An agent dialogue

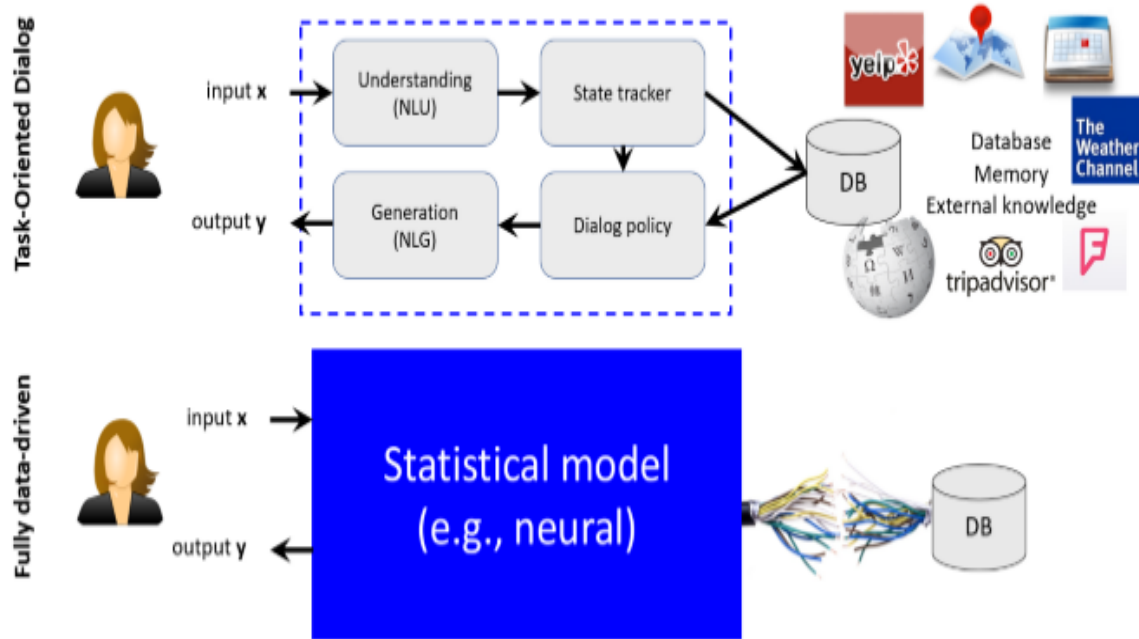


Figure 2.5: Two architectures of dialogue systems for (Top) .

Construction of a dialog agent to accomplish complex errands as in Fig. 1.1 One of the main obstacles to the party of IR and NLP people, and AI as a rule, remains. Four modules comprise a standard task-oriented dialog agent, as outlined above (Top): (1) NLU module to recognize customer expectations and separate related data; (2) a state tracker to follow the discourse expresses that captures all basic data in the discussion so far. (3) an exchange arrangement that selects the following activity depending on the state in question and (4) an NLG module to change to regular language reactions over specialist activities.

## Conclusion

Conversational AI is a quickly developing field. This paper reviews neural methodologies that were as of late created. Some of them have just been broadly utilized in business frameworks.

- Exchange frameworks for question replying, task culmination, chatter and suggestion can be conceptualized utilizing a unified numerical structure of ideal choice procedure. The neural ways to deal with AI, created over the most recent couple of years, influence on the ongoing RL and DL leap forward to significantly improve the exhibition of discourse specialists across a broad range of undertakings and areas. A variety of commercial dialog systems allow users to access different resources and information through conversation with ease. Most of these systems employ hybrid approaches combining the power of symbolic methods with neural models.
- There are two kinds of Operators of QA. KB-QA specialists allow customers to question large-scale information bases through discussion, without confusing SQL inquiries. Content QA operators, furnished with neural MRC models, are becoming more famous than conventional web crawlers (e.g., Bing and Google) for the types of questions to which customers expect a brief, straight answer.
- Conventional task-oriented frameworks use handmade exchange manager modules, or shallow ML models to streamline the modules independently. As of late, specialists have started to explore DL and RL to upgrade the framework in an increasingly all encompassing manner with less space information, and to mechanize the improvement of frameworks in a changing situation to such an extent that they can efficiently adjust to various assignments, areas and client practices.
- Chatbots are significant in encouraging smooth and regular communication among people and their electronic gadgets. Later work centers around situations past chitchat, e.g., recommendation. Most cutting edge chatbots use fully informationally driven and end-to - end conversational reactions within the neural machine interpretation structure.

## AI chatbots in python with AIML by NANO DANO

AI chatbots are easy to make using python package called AIML, but is simple XML and one of which is made by NANO DANO and we will discuss about it here.

### What is AIML?

It is an Artificial Intelligent Markup Language which is used to create or customize chat-box application based on A.L.I.C.E, it was developed by Richard Wallace

### Steps to make AI chatbots in python with AIML

Step1:

#### Create Standard Startup File

It is standard to create a startup file called **std-startup.xml** as the main entry point for loading AIML files. In this case we will create a basic file that matches one pattern and takes one action. We want to match the pattern **load aiml b**, and have it load our aiml brain in response. We will create the **basic\_chat.aiml** file in a minute.

```
<aiml version="1.0.1" encoding="UTF-8">
  <!-- std-startup.xml -->

  <!-- Category is an atomic AIML unit -->
  <category>

    <!-- Pattern to match in user input -->
    <!-- If user enters "LOAD AIML B" -->
    <pattern>LOAD AIML B</pattern>

    <!-- Template is the response to the pattern -->
    <!-- This learn an aiml file -->
    <template>
      <learn>basic_chat.aiml</learn>
      <!-- You can add more aiml files here -->
      <!--<learn>more_aiml.aiml</learn>-->
    </template>

  </category>

</aiml>
```

Step 2:

## Creating an AIML File

Above we created the AIML file that only handles one pattern, **load aiml b**. When we enter that command to the bot, it will try to load **basic\_chat.aiml**. It won't work unless we actually create it. Here is what you can put inside **basic\_chat.aiml**. We will match two basic patterns and respond.

```
<aiml version="1.0.1" encoding="UTF-8">
<!-- basic_chat.aiml -->

  <category>
    <pattern>HELLO</pattern>
    <template>
      Well, hello!
    </template>
  </category>

  <category>
    <pattern>WHAT ARE YOU</pattern>
    <template>
      I'm a bot, silly!
    </template>
  </category>

</aiml>
```

## Random Responses

You can also add random responses like this. This one will respond randomly when it receives a message that starts with "One time I ". The \* is a wildcard that matches anything.

```
<category>
  <pattern>ONE TIME I *</pattern>
  <template>
    <random>
      <li>Go on.</li>
      <li>How old are you?</li>
      <li>Be more specific.</li>
      <li>I did not know that.</li>
      <li>Are you telling the truth?</li>
      <li>I don't know what that means.</li>
      <li>Try to tell me that another way.</li>
      <li>Are you talking about an animal, vegetable or mineral?</li>
      <li>What is it?</li>
    </random>
  </template>
</category>
```

## Use Existing AIML Files

It can be fun to write your own AIML files, but it can be a lot of work. I think it needs around 10,000 patterns before it starts to feel realistic. Fortunately, the ALICE foundation provides a number of AIML files for free. Browse the AIML files on the [Alice Bot website](#). There was one floating around before called std-65-percent.xml that contained the most common 65% of phrases. There is also one that lets you play BlackJack with the bot.

Step 3:

## Install Python AIML Module

So far, everything has been AIML XML files. All of that is important and will make up the brain of the bot, but it's just information right now. The bot needs to come to life. You could use any language to implement the AIML specification, but some nice person has already done that in Python.

### Python 2

Install the `aiml` package first with `pip` or download from <https://pypi.python.org/pypi/aiml/>.

```
pip install aiml
```

### Python 3

<https://github.com/paulovn/python-aiml>

For Python 3, the source code remains exactly the same. You still import the package as `aiml` but when installing it with `pip` you use the name `python-aiml`. The source code is available at <https://github.com/paulovn/python-aiml>.

```
pip install python-aiml
```

## Simplest Python Program

This is the simplest program we can start with. It creates the `aiml` object, learns the startup file, and then loads the rest of the `aiml` files. After that, it is ready to chat, and we enter an infinite loop that will continue to prompt the user for a message. You will need to enter a pattern the bot recognizes. The patterns recognized depend on what AIML files you loaded.

We create the startup file as a separate entity so that we can add more `aiml` files to the bot later without having to modify any of the programs source code. We can just add more files to learn in the startup xml file.

```
import aiml

# Create the kernel and learn AIML files
kernel = aiml.Kernel()
kernel.learn("std-startup.xml")
kernel.respond("load aiml b")

# Press CTRL-C to break this loop
while True:
    print kernel.respond(raw_input("Enter your message >> "))
```

Step 4:

## Speeding up Brain Load

When you start to have a lot of AIML files, it can take a long time to learn. This is where brain files come in. After the bot learns all the AIML files it can save its brain directly to a file which will drastically speed up load times on subsequent runs.

```
import aiml
import os

kernel = aiml.Kernel()

if os.path.isfile("bot_brain.brn"):
    kernel.bootstrap(brainFile = "bot_brain.brn")
else:
    kernel.bootstrap(learnFiles = "std-startup.xml", commands = "load aiml b")
    kernel.saveBrain("bot_brain.brn")

# kernel now ready for use
while True:
    print kernel.respond(raw_input("Enter your message >> "))
```

Step 5:

## Reloading AIML While Running

You can send the load message to the bot while it is running and it will reload the AIML files. Keep in mind that if you are using the brain method as it is written above, reloading it on the fly will not save the new changes to the brain. You will either need to delete the brain file so it rebuilds on the next startup, or you will need to modify the code so that it saves the brain at some point after reloading. See the next section on creating Python commands for the bot to do that.

```
load aiml b
```

## Adding Python Commands

If you want to give your bot some special commands that run Python functions, then you should capture the input message to the bot and process it before sending it to `kernel.respond()`. In the example above we are getting user input from `raw_input`. We could get our input from anywhere though. Perhaps a TCP socket, or a voice-to-text source. Process the message before it goes through AIML. You may want to skip the AIML processing on certain messages.

```
while True:
    message = raw_input("Enter your message to the bot: ")
    if message == "quit":
        exit()
    elif message == "save":
        kernel.saveBrain("bot_brain.brn")
    else:
        bot_response = kernel.respond(message)
        # Do something with bot_response
```

## Step 6:

### Sessions and Predicates

By specifying a session, the AIML can tailor different conversations to different people. For example, if one person tells the bot their name is Alice, and the other person tells the bot their name is Bob, the bot can differentiate the people. To specify which session you are using you pass it as a second parameter to **respond()**.

```
sessionId = 12345
kernel.respond(raw_input(">>>"), sessionId)
```

This is good for having personalized conversations with each client. You will have to generate your own session Id some how and track them. Note that saving the brain file does not save all the session values.

```
sessionId = 12345

# Get session info as dictionary. Contains the input
# and output history as well as any predicates known
sessionData = kernel.getSessionData(sessionId)

# Each session ID needs to be a unique value
# The predicate name is the name of something/someone
# that the bot knows about in your session with the bot
# The bot might know you as "Billy" and that your "dog" is named "Brandy"
kernel.setPredicate("dog", "Brandy", sessionId)
clients_dogs_name = kernel.getPredicate("dog", sessionId)

kernel.setBotPredicate("hometown", "127.0.0.1")
bot_hometown = kernel.getBotPredicate("hometown")
```

In the AIML we can set predicates using the **set** response in template.

```
<aiml version="1.0.1" encoding="UTF-8">
  <category>
    <pattern>MY DOGS NAME IS *</pattern>
    <template>
      That is interesting that you have a dog named <set name="dog"><star/></set>
    </template>
  </category>
  <category>
    <pattern>WHAT IS MY DOGS NAME</pattern>
    <template>
      Your dog's name is <get name="dog"/>.
    </template>
  </category>
</aiml>
```

With the AIML above you could tell the bot:

```
My dogs name is Max
```

And the bot will respond with

```
That is interesting that you have a dog named Max
```

And if you ask the bot:

```
What is my dogs name?
```

The bot will respond:

```
Your dog's name is Max.
```



**More Notable researches are mentioned below:**

- Eliza is considered as the first chatbot which takes a shot at the example coordinating framework. It is created by “Joseph Weizenbaum in 1964 ”. ALICE is rule-put together chatbot based with respect to the AIML. It has in excess of 40,000 classifications, where every classification has blend of example and its reaction.
- “Md.Shahriare Satu and Shamim-AI-Mamun” demonstrated the audit of utilizations of the bot which are created utilizing the AIML contents. They said that AIML based chatbots are anything but difficult to actualize, they are light and proficient to work. Document gives the nitty gritty data about the various utilizations of the chatbots.
- “Thomas N. T. also, Amrita Vishwa” structured an ‘AIML and LSA based chatbot’ to give the client care administration over the E-business sites. Their methodology shows we can improve the chatbot capacity by adding different models to it.
- In android working framework, we can actualize the chatbot utilizing the different methodologies. One of the methodologies is appeared by ‘Rushabh Jain and BurhanuddinLokhandwala in their Android based Chat-Bot paper’.

## **Chapter 3: System Development**

### **3.1 Analysis/ Design / Development / Algorithm**

Objective: To create an interactive chatbot for induction into a VPA.

#### **Analysis:**

As a professional project to assemble an item, we should pursue the necessity from the client. Be that as it may, on the grounds that the project venture will likely be utilized by the public users, yet it is mindful by the specialized group, the necessity changed a great deal in the center after a gathering with the public. The public users need a straightforward bot that can give suggestion right away. It should be easy to operate and easy to understand its functionality by the users. The VPA chatbot must respond to user commands and queries instantly and accurately. The VPA must recognise the user's voice and convert the voice to text accurately and then send the interpreted voice as text to the system to generate a response accordingly. The VPA must also convert the generated response from text to digitized voice and speak the response to the user out loud. It should also give error responses if the voice is not interpreted correctly or if there is no correct response for the query. The VPA must be highly efficient.

## **Design:**

Our model functions in two modes: text and voice. The key mode is actuated at the point where client gives the contribution to text arrangement. The client input for the response is passed to the middleware API. On the other hand, when the client enters the voice in that second mode, we first convert the voice into text in this voice mode before sending it to the middleware API. Middleware is the model that interfaces contents of ArAIML with our android application. When client input is provided at the middleware, it is transferred to the coordinating calculation example that runs over the contents of the AIML. In this proceeding, right off the bat the example coordinating calculation is executed for coordinating of the legitimate reaction from the accessible AIML contents. At the point when example is coordinated, the relating format is come back to the middleware. Middleware encodes the layout into the JSON group at that point, and sends the answer to the android application. In the wake of getting the application to react interpret the JSON and give the client reaction.

The reaction procedure is completed in two stages :

### **A. Readiness of Pattern Matching**

Every contribution to the AIML translator goes through two basic stages. Standardizing Input process. Delivering every sentence input route..

### **B. Example Matching Behavior**

Here we try to locate the largest coordinating example and best one by word coordinating the input. This behavior can be portrayed with Graph ace arrangement of records and registers containing a lot of hubs called hub ace and branches speak to first expressions all considered things.

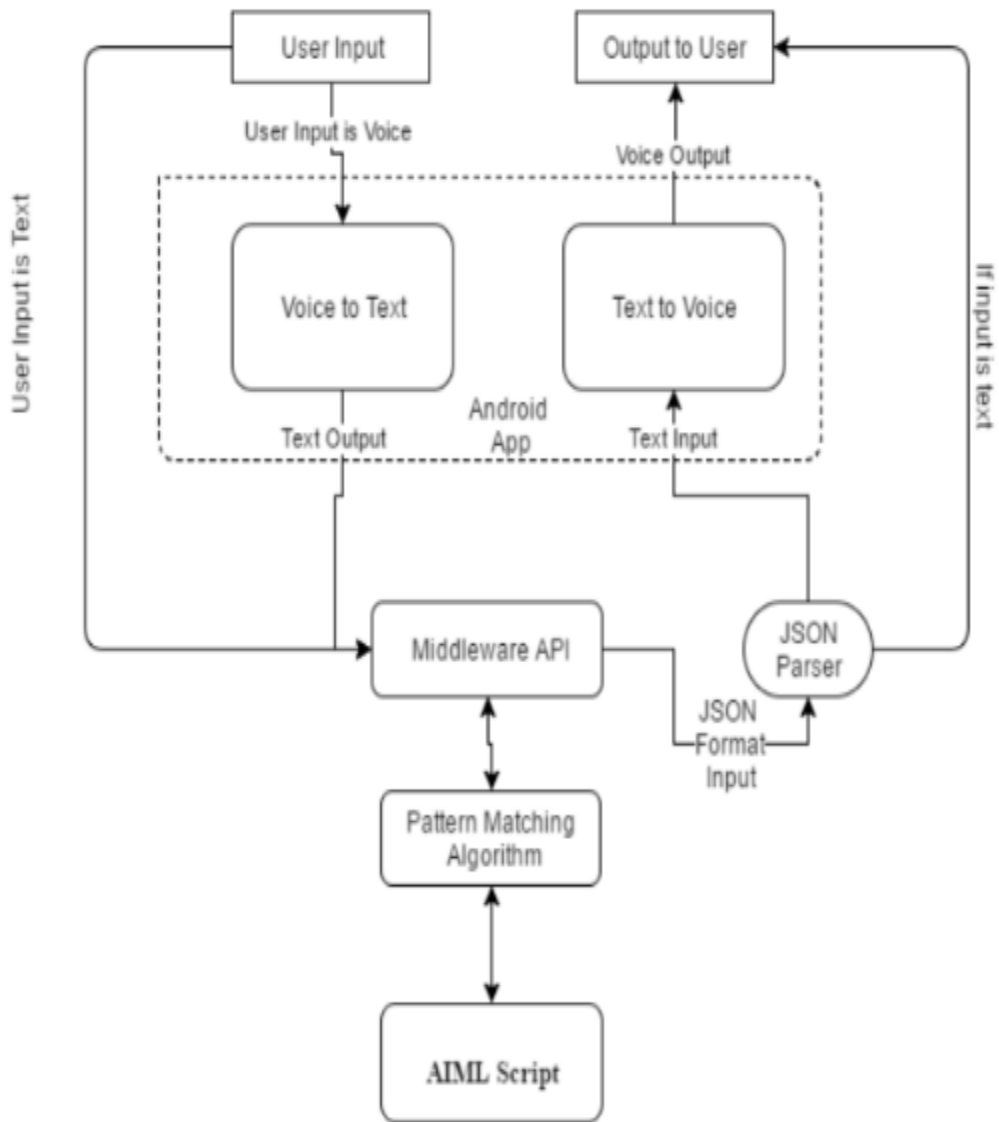


figure 3.1. System Architecture

## Development:

There are two types of development model and require different implementations.

### A. Implementation of Textbased Chatbot

This methodology begins with taking in the android application the client's contribution to the content. We elevate clients to enter the chatbot inquiry and then read in the alter content the incentive entered. At this stage, submit the incentive to our facilitated middleware API Program interface. Get the server response, server replies in JSON design. We need to parse it to get the reaction from the bot. In the event that bot's reaction is invalid, at that point set the reaction to the blunder string and demonstrate it to the client. Rehash a similar assignment for the consistent visiting with the chatbot.

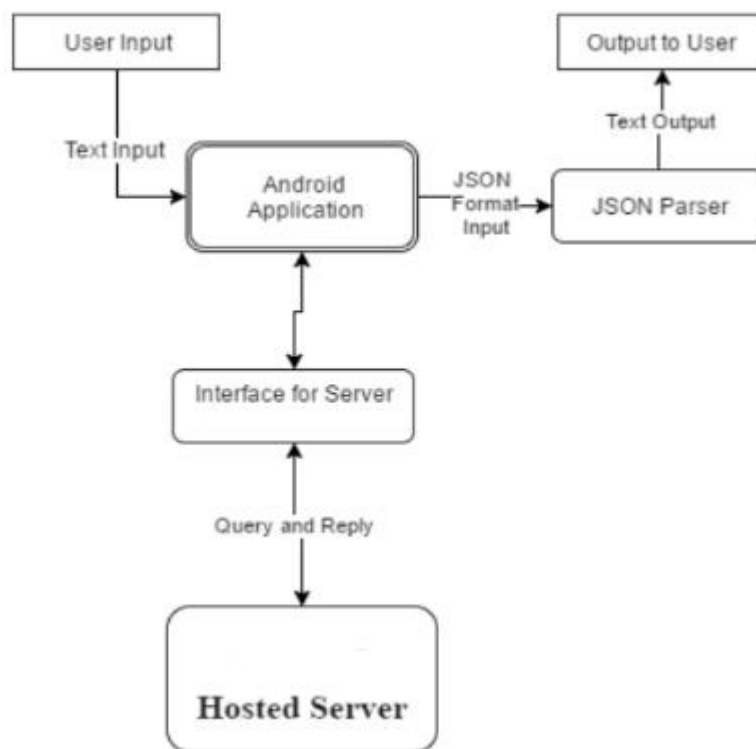


figure 3.2. Text Based Chatbot

The accompanying advances disclose the usage subtleties to achieve the previously mentioned undertakings:

1. Peruse the client contribution on click event of the send button.
2. Send the contribution of the client to the server facilitated by the middleware API program using the `HttpURLConnection`. We 're managing the web stuff here so we need to call that strategy in the android framework's `AsyncTask`. The `AsyncTask` program runs out of sight.
3. Hold on till the server gets the answer. `OnPostExecute` strategy for the `AsyncTask` gets the answer arranged by JSON. Send it to JSON parser then to remove the reaction of the bot.
4. In the wake of separating the bot's reaction, we have to show it to the client. In this way, add the reaction to text view.
5. At that point rehash the procedure for the following client input.

## **B. Implementation of Voicebased Chatbot**

This methodology begins with voice contribution for the client. Here on start application will approach client for the voice contribution for client. At that point we have to change over the voice an incentive to the content since we have the server with our program – o which sees just content qualities. After the transformation of the voice to content the worth is sent to the server for the answer. As we have just talked about server answers in JSON position, so we have to parse it. Get the reaction of the bot after the parsing. We have taken the voice input in this way, here we are giving the voice yield. Bot's reaction is changed over to voice and played through the telephones speakers. Rehash the means for nonstop visiting with chatbot. The accompanying advances disclose the usage subtleties to achieve the previously mentioned assignments:

1. Advance client for the voice contribution on onCreate technique. Record the client voice till the discourse end occasion.
2. Presently convert the voice into content utilizing the Google's Speech Recognizer API. In the wake of getting consequence of the transformation send information to the server.
3. Utilize the HttpURLConnection in the AsyncTask for mentioning the answer from the server.
4. We will get the answer in the JSON , parse it to get the answer of the bot.
5. Here we are actualizing the voice based chatbot so answer from the bot must be voice. Convert the bot's answer to the voice utilizing Text to Speech API for the android.
6. After complete voice play, again approach client for new voice info and rehash the means.

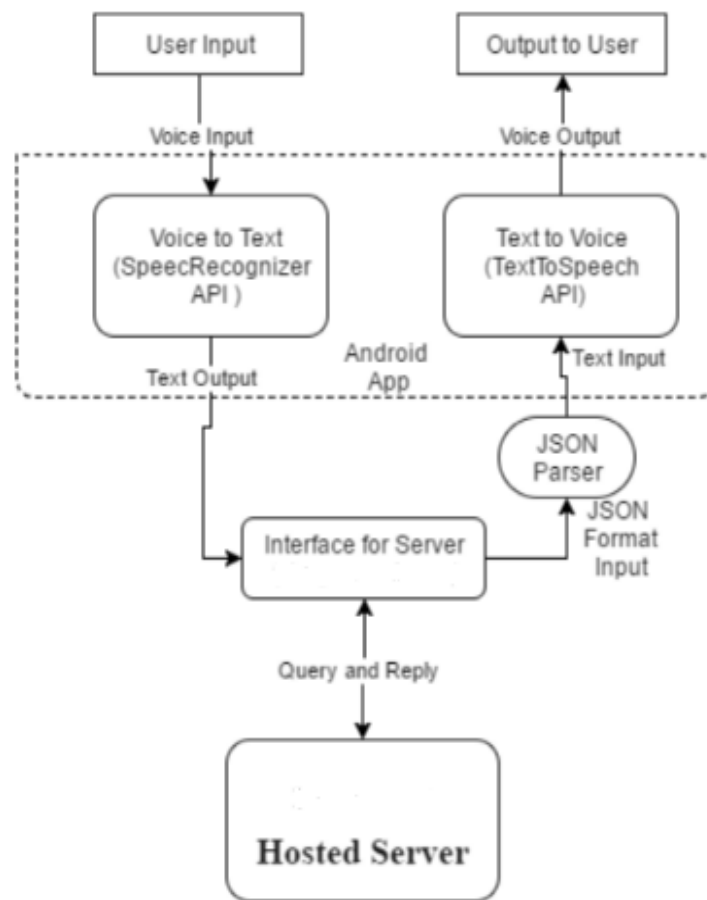


figure 3.3. Voice based Chatbot



## **Requirements And Specifications :**

The model was trained for different variations of the model for varying time periods on a system having the configuration as follows:

Ram: 8GB

Core: i5

GPU: 2GB

64-bit O/S

x64 based processor

jupyter

**Model development (Analytical, computational, experimental, mathematical, statistical)**

## Chapter 4: Performance Analysis

### App Icon

Below screenshot displays the icon of our app, Virtual Personal Assistant (VPA) i.e. Dexter.



Figure 4.1 App Icon

## Display Screen

As you can see from the screenshot below the display screen is loading correctly and it is taking very less time to open.



Figure 4.2 Home Screen

## Account Manager Screen

Our app is managing the accounts, creating and managing the data for the different accounts individually.

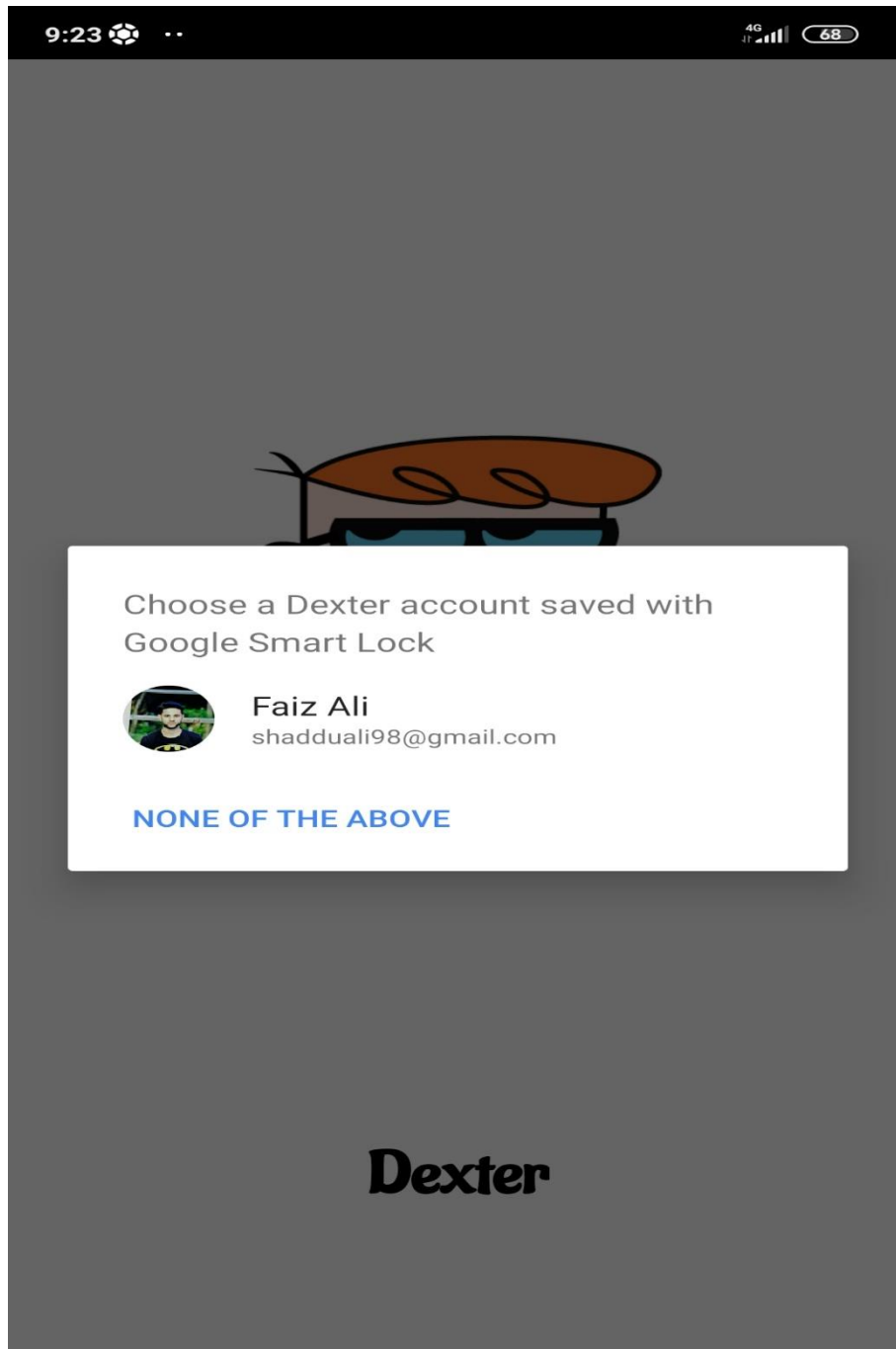


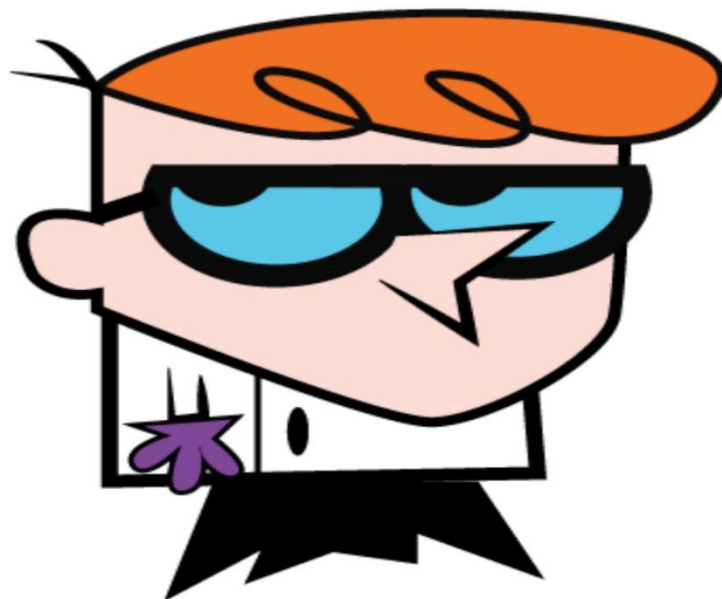
Figure 4.3.1 Login Screen

9:23

4G 68

## Hi, Faiz Ali

Hey! I'm Dexter, your digital companion.



Start

Sign Out

Figure 4.3.2 Screen after Login

## Operation performed on app(Conversation)

After some interactions with the personal assistant, below are the screenshot of the interaction:

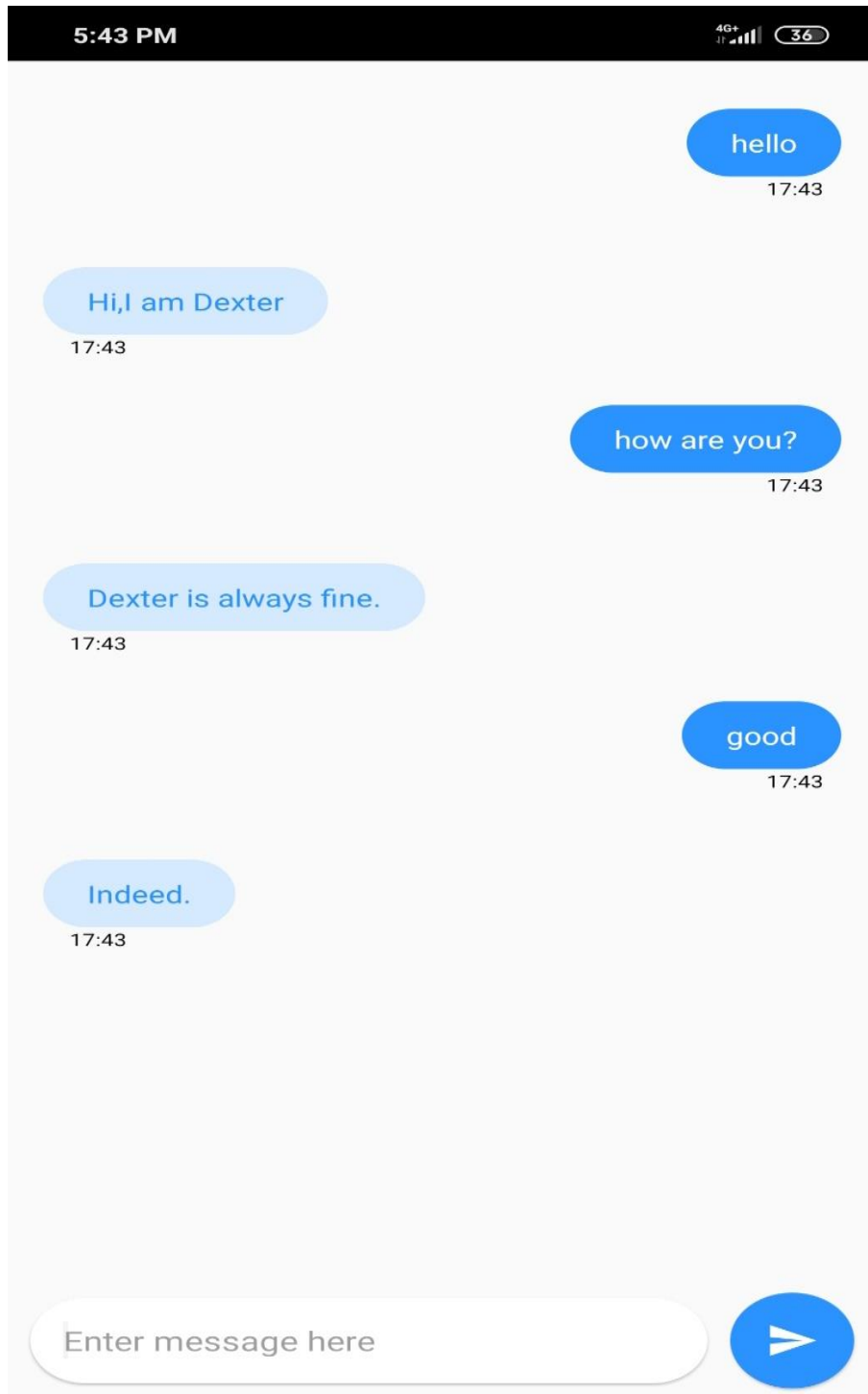


Figure 4.4.1 Conversation with bot

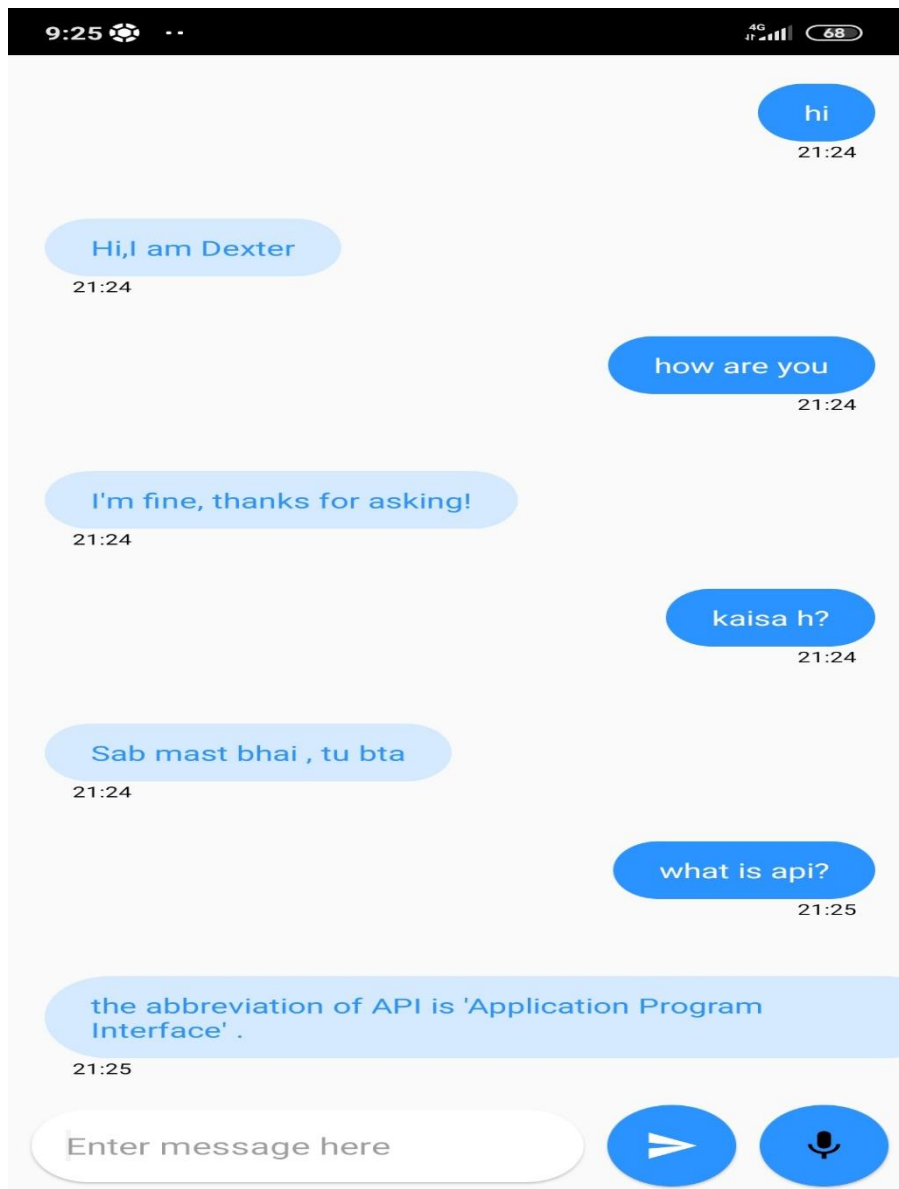


Figure 4.4.2 Conversation with bot

Our personal assistant is able to reply correctly and perfectly to the basic conversation.

As it has been show in the above screenshot

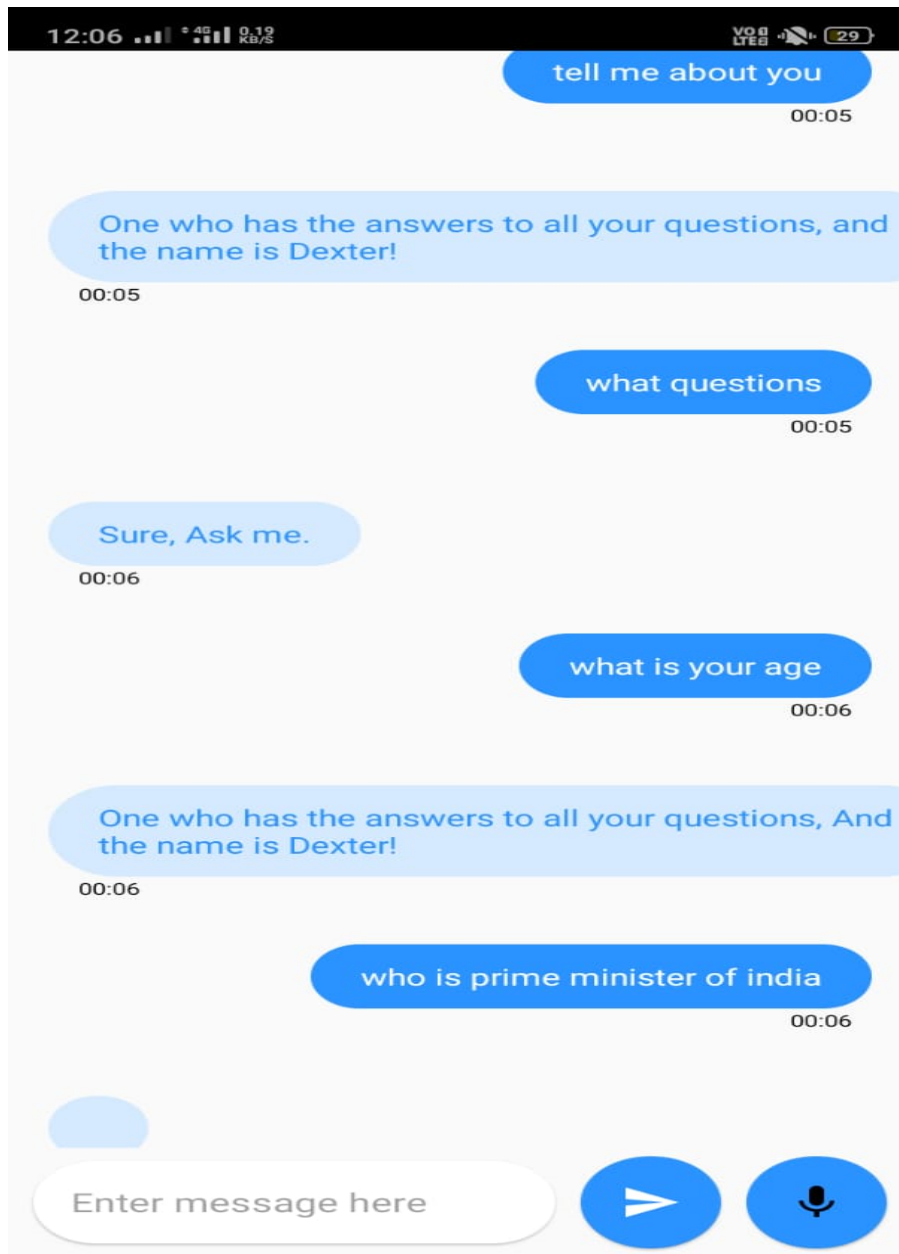


Figure 4.4.3 Failed conversation with bot

Our personal assistant is not able to reply to some of the tricky question because it has yet to be developed and trained for more completed question asked.

Above is the snapshot of the conversation to which it is not able to reply correctly.



## Speech To Text

Screenshot example showing speech to text conversation with the bot:

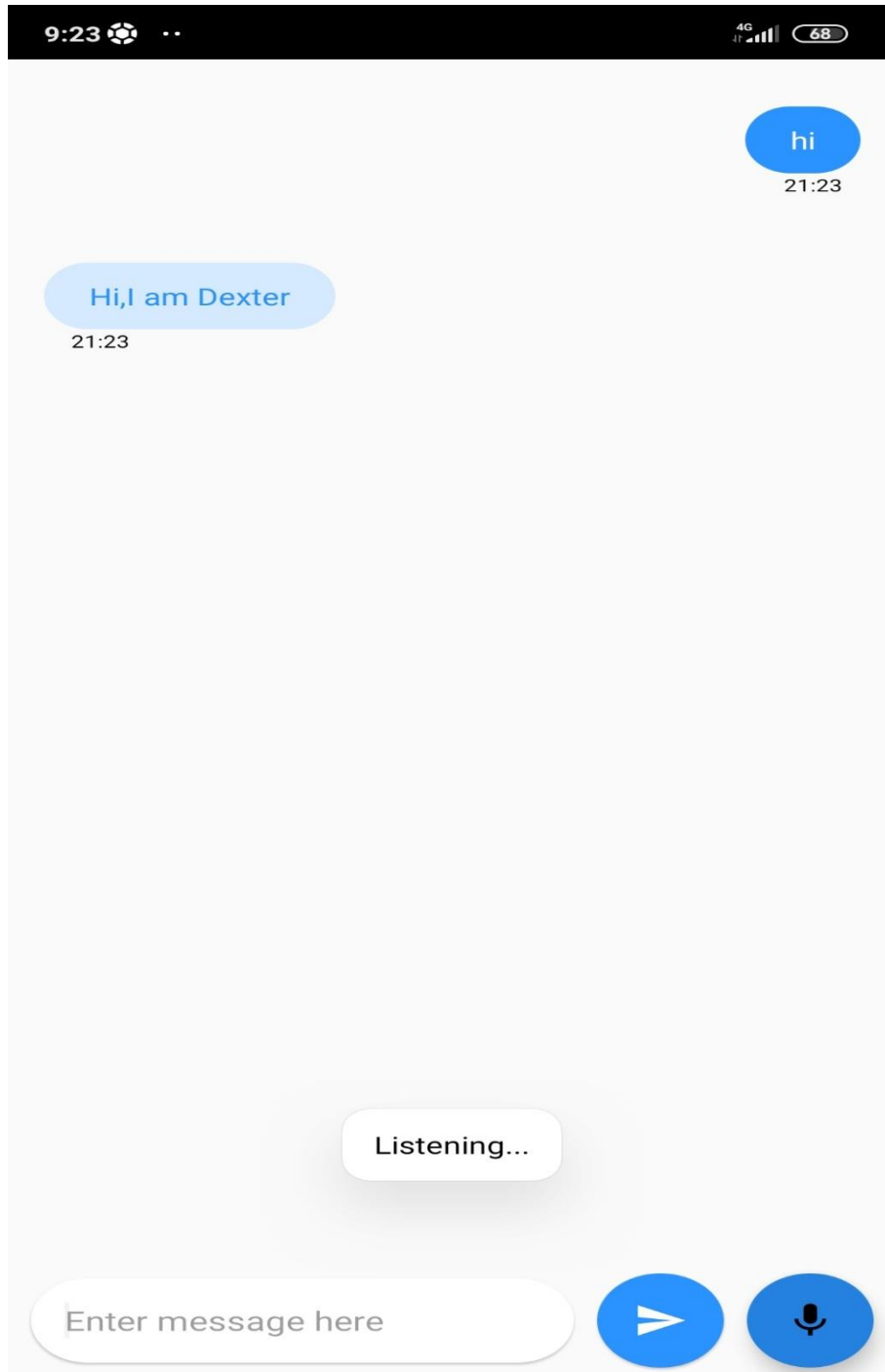
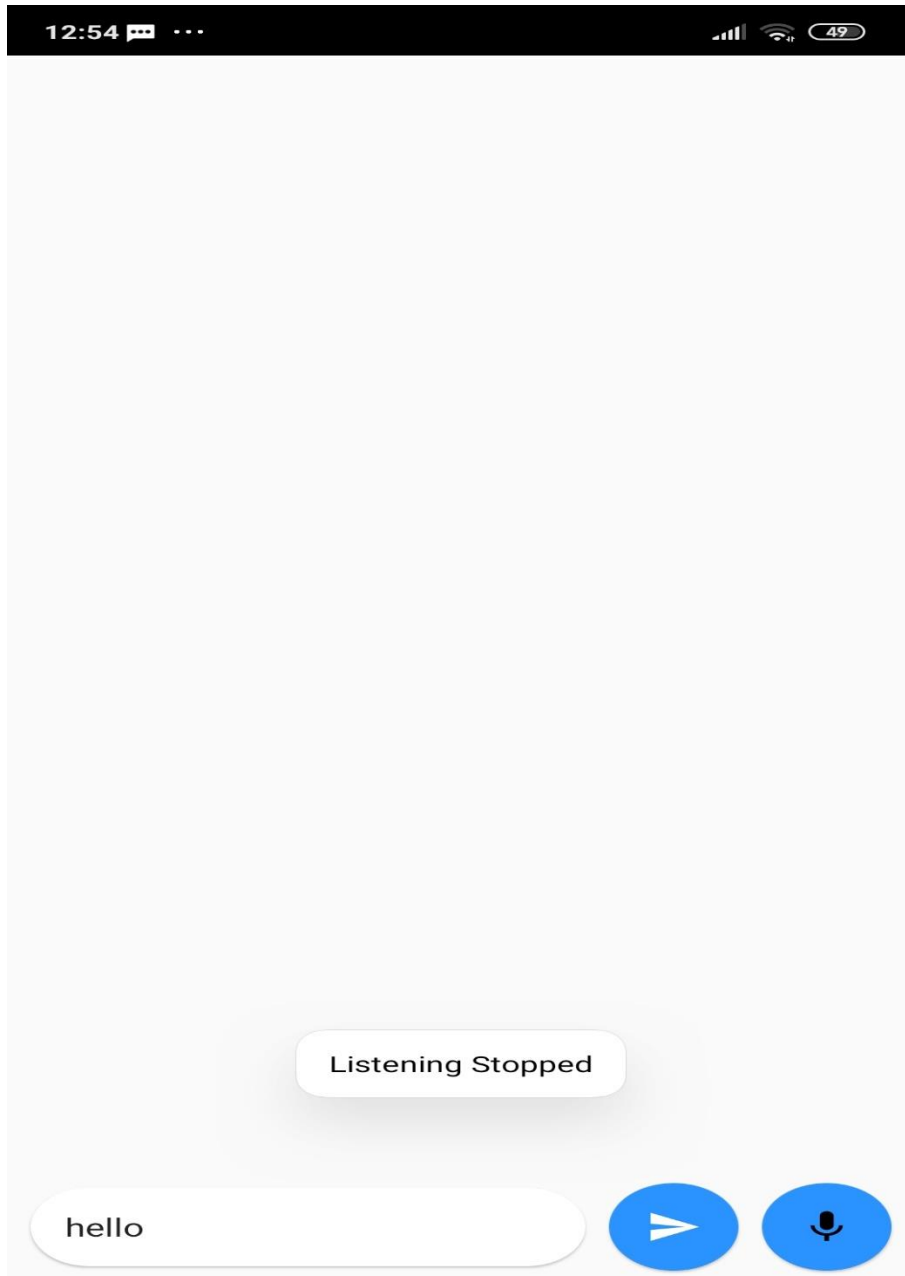


Figure 4.5.1 Listening Voice



*Figure 4.5.2 After Listening*

Our bot is taking voice input, displaying it first and giving us option to check and correct if the input is correct before sending it. We can send the voice input and get the reply from the bot.

## **Chapter 5: Conclusion**

### **5.1 Conclusion**

In this paper we introduced an android VPA application that is capable of interacting with users. This chatbot can answer user input queries in both the textual and the voice form. To this end, Natural Language Processing and Machine Learning techniques are used. New technologies like dialogflow and firebase by Google is also used in the development of this app. The new Apps are VPA Chatbots! As we've talked about in the expectations above, this task brings chatbots' influence to VPA and advances its ease of use. VPA chatbots can give a human touch to certain points of view and make it a discussion charge. What's more, they are concentrated totally on giving data and finishing errands to the people they interface with. The previously mentioned usefulness in every one of the expectations is executed and pushed in to VPA code. By actualizing the previously mentioned expectations I had the option to include an essential chatbot usefulness in to the VPA i.e. arranging and making represents bot clients with bot settings, actuating a bot at whatever point a client inquires for it by means of post in a string and as I talked about earlier, I have actualized a basic climate chatbot that gives climate data at whatever point a client ask and advises I was always able to speak to the bot in VPA. I expect further refurbishment of the framework. Helping individuals to encourage their work and associate with PCs and mobiles using characteristic language or using set of rules includes following the step towards building chatbots. Future VPA chatbots, backing AI innovation, you'll be ready to remember and gain from past discussions to respond to new ones. The study will converse with various bot clients and specific clients

### **Challenges Faced:**

- Voice to Text was not working properly, there was some minor complication due to which voice input was not displaying and by clicking the button voice recording function was not called.
- App icon was not displaying correctly.
- There was some bug in UI due to which the background image was not displayed properly.
- We had to scroll after sending every message because it was displaying previous message.

### **Changes to be Made(Future work):**

- Make a single button for sending message and taking voice input.
- Include option to add background image.
- Improve the UI
- Make bot more interactive by adding personalize touch.

## **5.2 Future Scope**

There are impediments to what has been at present accomplished with chatbots. The constraints of information handling and recovery are impeding chatbots to arrive at their maximum capacity. However, we can still improve on the product. The chatbot can address just those inquiries which has the appropriate response in its dataset. Along these lines, to expand the information on the chatbot, we could include the Wikipedia APIs, Sports , News, Government Services and more. In such cases the client will have the option of speaking and connecting in any space with the chatbot. Using APIs such as Weather, Sports , News and Government Services, the chatbot will have the option of reacting to inquiries outside its dataset that are actually taking place right now.. Furthermore and foremostly, we will add basic functions in the VPA which are available in the mobile phone, i.e making calls, opening other apps, making notes or reminders, setting alarms, etc. The sole purpose of developing this VPA is that, a user would not require to open any other app and could communicate to all other apps only through the VPA.

What could be our next task:

- Train our bot on larger dataset so its accuracy will increase.
- Give more permissions to our bot considering the security so that it can perform more of our tasks.
- Give direction to our bot, so it can perform task of a particular category more perfectly eg. There are bots who give replies to medical enquiries, food recommendation etc.

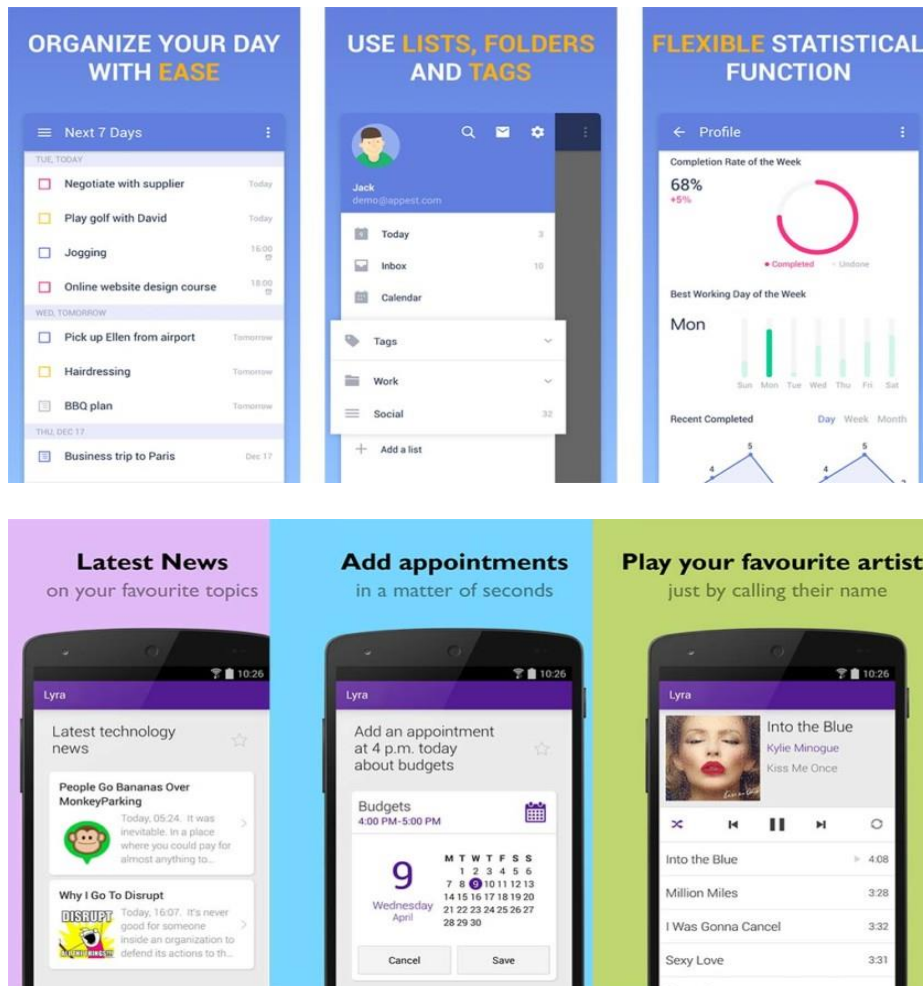


Figure 5.2.1 Different Types of bots

### 5.3 Application Contribution

The application contributes towards welfare of public as it would be helpful for people having less or no knowledge about smartphones, as the VPA would make it easier to operate the smartphone. The inbuilt chatbot would also be helpful in curbing loneliness.

## **REFERENCES**

- Bayan Abu Shawar and Eric Atwell, 2007 “Chatbots: Are they Really Useful?”
- LDV Forum - GLDV Journal for Computational Linguistics and Language Technology.
- [http://www.ldv-forum.org/2007\\_Heft1/Bayan\\_Abu-Shawar\\_and\\_Eric\\_Atwell.pdf](http://www.ldv-forum.org/2007_Heft1/Bayan_Abu-Shawar_and_Eric_Atwell.pdf)
- Bringing chatbots into education: Towards natural language negotiation of open learner models. Know.-Based Syst. 20, 2 (Mar. 2007), 177-185.
- Intelligent Tutoring Systems: Prospects for Guided Practice and Efficient Learning. Whitepaper for the Army's Science of Learning Workshop, Hampton, VA. Aug 1-3, 2006.
- <http://en.wikipedia.org/wiki/Chatterbot>
- ALICE. 2002. *A.L.I.C.E* AI Foundation, <http://www.alicebot.org/>
- Dialog Flow- <https://dialogflow.cloud.google.com/>
- Firebase for Database- <https://firebase.google.com/>

**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT**

**PLAGIARISM VERIFICATION REPORT**

Date: .....17-10-2020.....

Type of Document (Tick):  PhD Thesis  M.Tech Dissertation/ Report  B.Tech Project Report  Paper

Name: \_\_\_\_\_ Department: \_\_\_\_\_ CSE/

IT \_\_\_\_\_ Enrolment No. \_161252,161243\_\_\_\_\_ Contact No.

\_\_\_\_\_ E-mail. \_\_\_\_\_ Name of

the Supervisor: \_\_\_Dr.Amol Vasudeva

\_\_\_\_\_ Title of the Thesis/

Dissertation/Project Report/Paper (In Capital letters): \_\_\_Virtual Personal Assistant using Natural Language Processing and Machine Learning **UNDERTAKING**

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/ revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

**Complete Thesis/Report Pages Detail:**

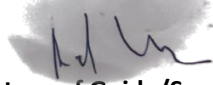
- Total No. of Pages =49
- Total No. of Preliminary pages =10
- Total No. of pages accommodate bibliography/references =



(Signature of Student)

**FOR DEPARTMENT USE**

We have checked the thesis/report as per norms and found **Similarity Index** at ...17.....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.



(Signature of Guide/Supervisor)

Signature of HOD

**FOR LRC USE**

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none"> <li>• All Preliminary Pages</li> <li>• Bibliography/Images/Quotes</li> <li>• 14 Words String</li> </ul>		Word Counts	
<b>Report Generated on</b>			Character Counts	
		<b>Submission ID</b>	Total Pages Scanned	
			File Size	

Checked by  
Name & Signature

Librarian

.....

**Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at [plagcheck.juit@gmail.com](mailto:plagcheck.juit@gmail.com)**