CHATBOT USING PYTHON

A project report

Submitted in the partial fulfillment of the requirements for the Award of the

Degree of

B.Sc.(DATA SCIENCE)

TO

Osmania University, Hyderabad
BY

KAJAL SINGH (125021539015)

PRIYANKA MONDAL (125021539037)

VAISHALI NAGARAHALLI (125021539026)

Under The Esteemed Guidance Of
Mrs. R. Gayathri (MSc Computer Science)

DEPARTMENT OF COMPUTER SCIENCE
VIVEKANANDA DEGREE AND PG COLLEGE
KUKATPALLY,HYDERABAD.

MAY - 2024

CANDIDATE DECLARATION

We here by declare that the work which is being presented in the project report entitled "**CHATBOT USING PYTHON"** in partial fulfillment of the requirement for the award on the Degree of Bachelor of Science and submitted in the Department of Computer Science of Vivekananda Degree & PG College, Kukatpally is an authentic record of our own work carried out during the academic year 2023-2024 under the supervision of Mrs. R. Gayathri, Department of Computer Science of Vivekananda Degree &PG College, Kukatpally. The matter presented in this report has not been submitted by us for the award of any other degree of this or any other institute/University.

PROJECT PRESENTED BY:

KAJAL SINGH	125021539015
PRIYANKA MONDAL	125021539037
VAISHALI NAGARAHALLI	125021539026

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Mrs. R. GAYATHRI**, Lecturer, Department of Computer Science. Vivekananda Degree & PG College, for her guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Mrs. T.BHAGYA,** Head of the Department, Computer Science, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management**, Vivekananda Degree & PG College, for their encouragement and cooperation to carry out this work. We express our thanks to all **Teaching Faculty** of Department of Computer Science, Whose suggestions during reviews helped us in accomplishment of our project. We would like to thank all **Non-Teaching Staff** of the Department of Computer Science, Vivekananda Degree & PG College for providing great assistance in accomplishment of our project. We thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT PRESENTED BY:

KAJAL SINGH	125021539015
PRIYANKA MONDAL	125021539037
VAISHALI NAGARAHALLI	125021539026

CHATBOT
USING PYTHON

ABSTRACT

The use of chatbots evolved rapidly in numerous fields in recent years, including Marketing, Supporting Systems, Education, Health Care, Cultural Heritage, and Entertainment. In this paper, we first present a brief introduction to chatbots and its types, artificial intelligence and its applications .After clarifying necessary technological concepts, we move on to a chatbot classification based on various criteria, such as the area of knowledge they refer to, the need they serve and others. Historical overview of the evolution of the in chatbots. Next, we discuss the existing systems of the. Moreover, we highlight the of chatbots design. Furthermore, we present the general architecture of modern chatbots while also mentioning the main platforms for their creation. Our engagement with the subject so far it helps in studying about the chatbots and encourages us to study them in greater extent and depth.. Example chatbots that can be developed with this API is weather chatbots orbook flight chatbots. Over past few years, messaging applications have become more popularthan Social networking sites. People are using messaging applications these days such asFacebook Messenger, Skype, Viber, Telegram, Slack etc. This is making other businesses available on messaging platforms leads to proactive interaction with users about their products. To interact on such messaging platforms with many users, the businesses can write a computer program that can converse like a human which is called a chatbot. Remember that chatbot development involves a mix of creativity, technical skills, and understanding user needs. Whether you're building a simple chatbot or a sophisticated AI assistant, Python provides a versatile platform to bring your ideas to life! With the power of AI, businesses in various industries can revolutionize customer experiences and provide valuable support and engagement. By leveraging Python and TensorFlow, developers can create AI chatbots that deliver cutting-edge solutions and drive business growth.

CONTENTS

List of figures

CHAPTER 1

INTRODUCTION

1.1 Introduction to chatbots

Artificial Intelligence (AI) increasingly integrates our daily lives with the creation and analysis of intelligent software and hardware, called intelligent agents. Intelligent agents can do a variety of tasks ranging from labor work to sophisticated operations. A chatbot is a typical example of an AI system and one of the most elementary and widespread examples of intelligent Human-Computer Interaction (HCI) . It is a computer program, which responds like a smart entity when conversed with through text or voice and understands one or more human languages by Natural Language Processing (NLP) . In the lexicon, a chatbot is defined as “A computer program designed to simulate conversation with human users, especially over the Internet” . Chatbots are also known as smart bots, interactive agents, digital assistants, or artificial conversation entities. Chatbots can mimic human conversation and entertain users but they are not built only for this. They are useful in applications such as education, information retrieval, business, and e-commerce. They became so popular because there are many advantages of chatbots for users and developers too. Most implementations are platform-independent and instantly available to users without needed installations. Contact to the chatbot is spread through a user’s social graph without leaving the messaging app the chatbot lives in, which provides and guarantees the user’s identity. Moreover, payment services are integrated into the messaging system and can be used safely and reliably and a notification system re-engages inactive users. Chatbots are integrated with group conversations or shared just like any other contact, while multiple conversations can be carried forward in parallel. Knowledge in the use of one chatbot is easily transferred to the usage of other chatbots, and there are limited data requirements. Communication reliability, fast and uncomplicated development iterations, lack of version fragmentation, and limited design efforts for the interface are some of the advantages for developers too .

Chatbots kinds:

a. **Rule-Based** - Rule-based chatbots, also known as decision-tree chatbots, are designed to follow a predefined set of rules and instructions. These chatbots operate based on a decision tree or flowchart structure, where user input triggers specific responses.Rule-based chatbots find application in various scenarios where interactions adhere to predefined patterns, regulations, and structured procedures. Some prevalent use cases of Rule-Based Chatbot are mentioned below:

- Customer Support and FAQs: Rule-based chatbots are frequently deployed to address customer queries and frequently asked questions. They swiftly deliver responses to common inquiries, encompassing product details, troubleshooting guidance, and contact particulars.
- Appointment Scheduling: Entities like healthcare providers, salons, and service providers employ rule-based chatbots to facilitate appointment scheduling. The chatbot can assess availability and confirm appointments based on preset rules.
- Order Tracking and Updates: E-commerce enterprises utilise rule-based chatbots to furnish customers with real-time updates on their orders. Users can seek information regarding the status of their deliveries, estimated delivery times, and tracking data.
- Restaurant Reservations: Restaurants leverage rule-based chatbots to streamline table reservations. Users specify their preferred date, time, and party size, and the chatbot confirms reservations contingent on availability.
- Travel Assistance: Travel agencies and booking platforms harness rule-based chatbots to aid users in trip planning, booking flights, hotels, and transportation. Additionally, they provide travel-related insights, encompassing visa prerequisites and packing recommendations.

b. **AI-Powered** - AI-powered chatbots are sophisticated virtual assistants that leverage artificial intelligence (AI) algorithms to simulate human-like interactions. These chatbots are designed to understand and interpret user input, providing personalized and context-aware responses.

- AI chatbots are commonly used as contact center solutions, real-time assistance to human agents, generative chatbots, voice capabilities, and sentiment analysis.
- Google Cloud's Dialogflow CX can help you create virtual agents that use generative AI to seamlessly switch between topics and operate across multiple channels 24/7.
- Vertex AI Agents enables developers to build AI-powered chat apps.
- Contact Center AI improves call center and customer service experiences.

1.1 Chatbot types:

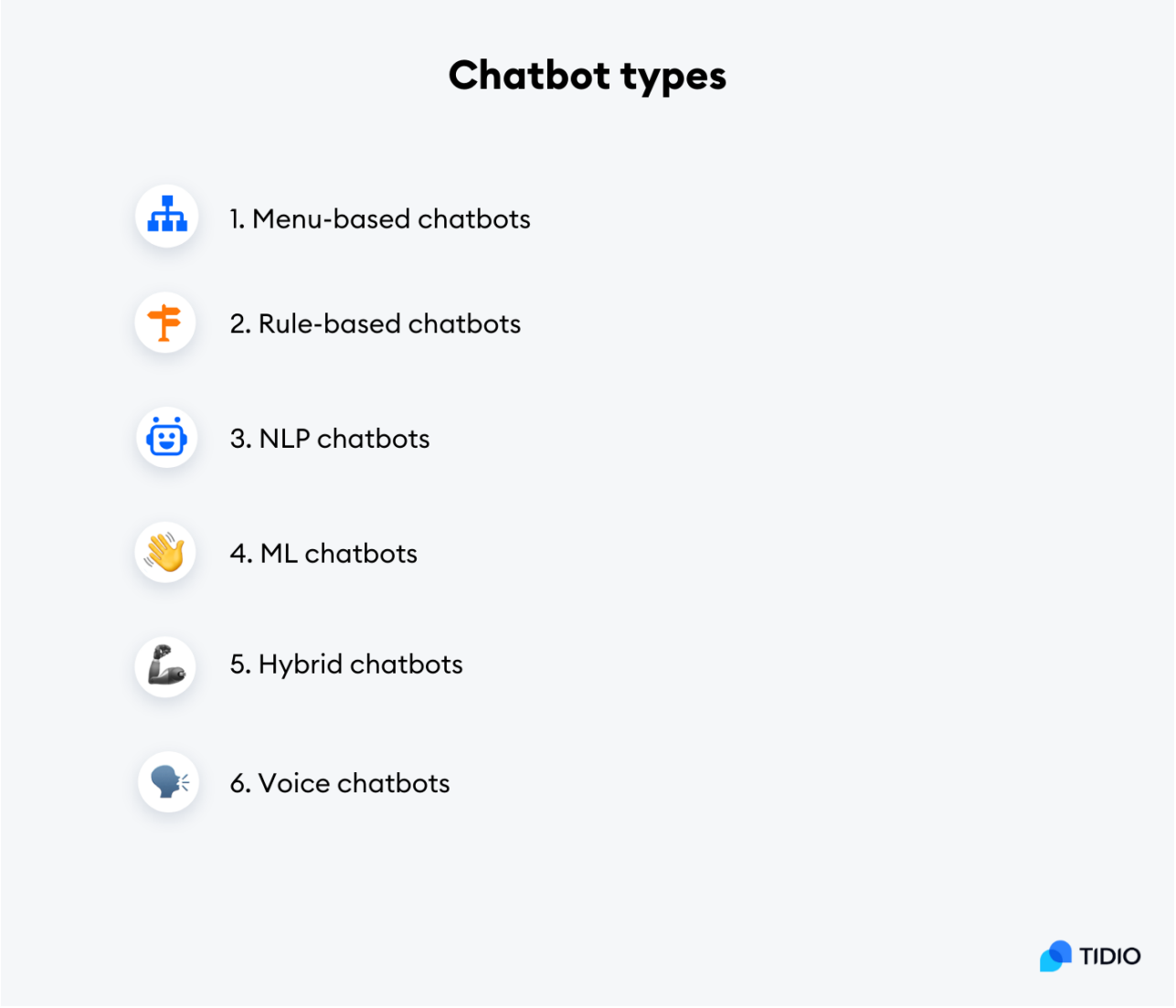


Figure 1. 1 Chatbot types

1. Menu-based chatbots

Menu or button-based chatbots are some of the most commonly used bot types out there. Generally speaking, they use the so-called decision tree type of logic, which is usually displayed as a list of menu buttons to users. Then, a user can pick an option that best corresponds with their inquiry to find an answer they are seeking.

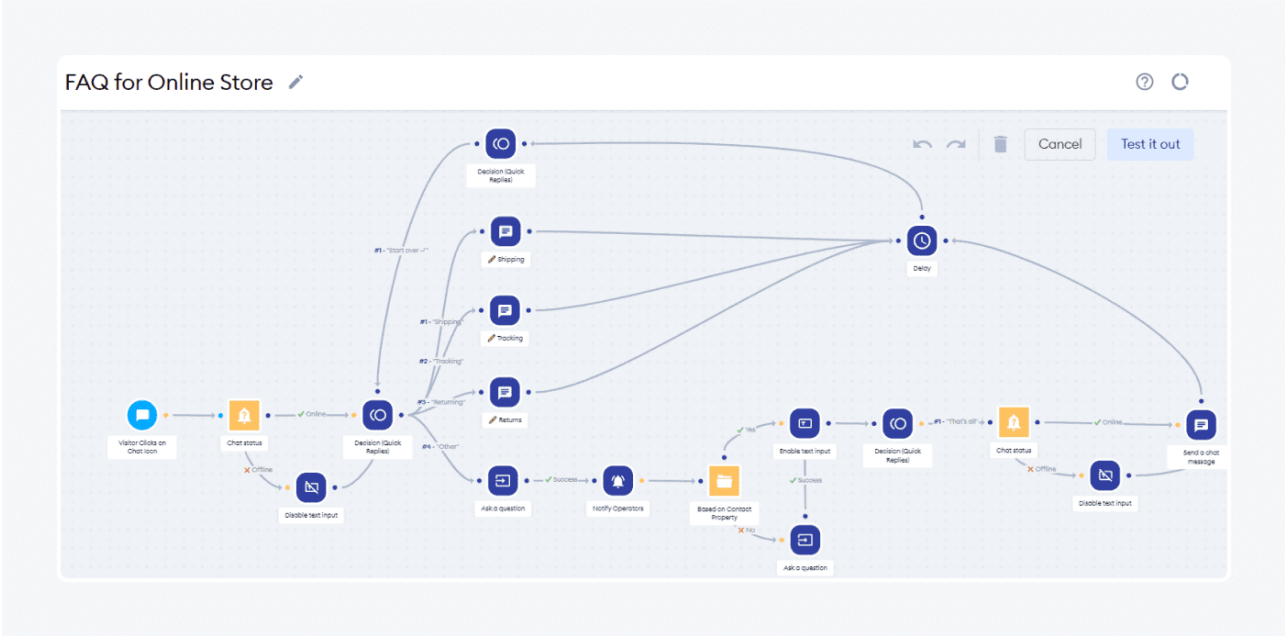


Figure 1. 2 FAQ for online store

These types of chatbots work great for answering simple customers' FAQs. That being said, they may not be as sufficient if users require more detailed answers.

If you own an online store of any kind, some of the frequently asked questions by your visitors may include things like wanting to track their order, learning more about your prices, checking available discounts, etc. Button-based bots may come particularly handy in these cases.

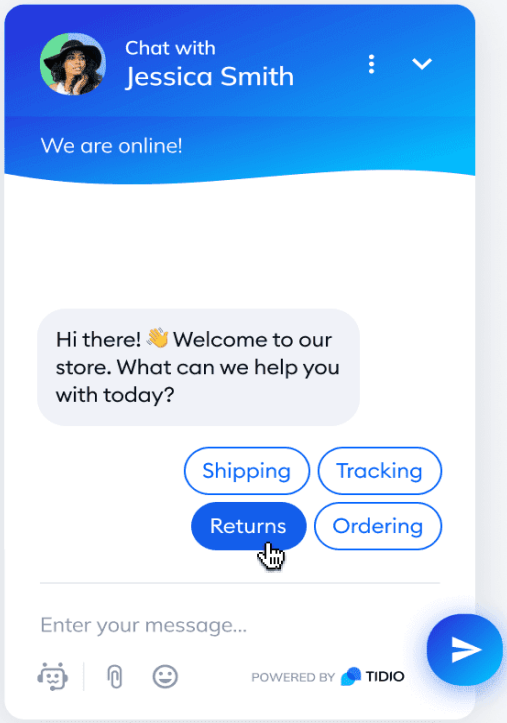


Figure 1. 3 Response

2. Rule-based chatbots

Rule-based chatbotsare perfect fit for businesses that know well which queries they can receive from their customers. Otherwise known as linguistic bots, rule-based chatbots work on the principle of *if-then* logic that helps them create conversational automation flows.

The *if-then* logic works by adding specific words that will trigger appropriate answers by this kind of bot. To make it as precise as possible, there are certain conditions that can be created. These include adding synonyms, specific order sequences of the words, etc.

If a customer uses a word or phrase that can be found within the list of established language parameters, then a bot will provide the appropriate answer.

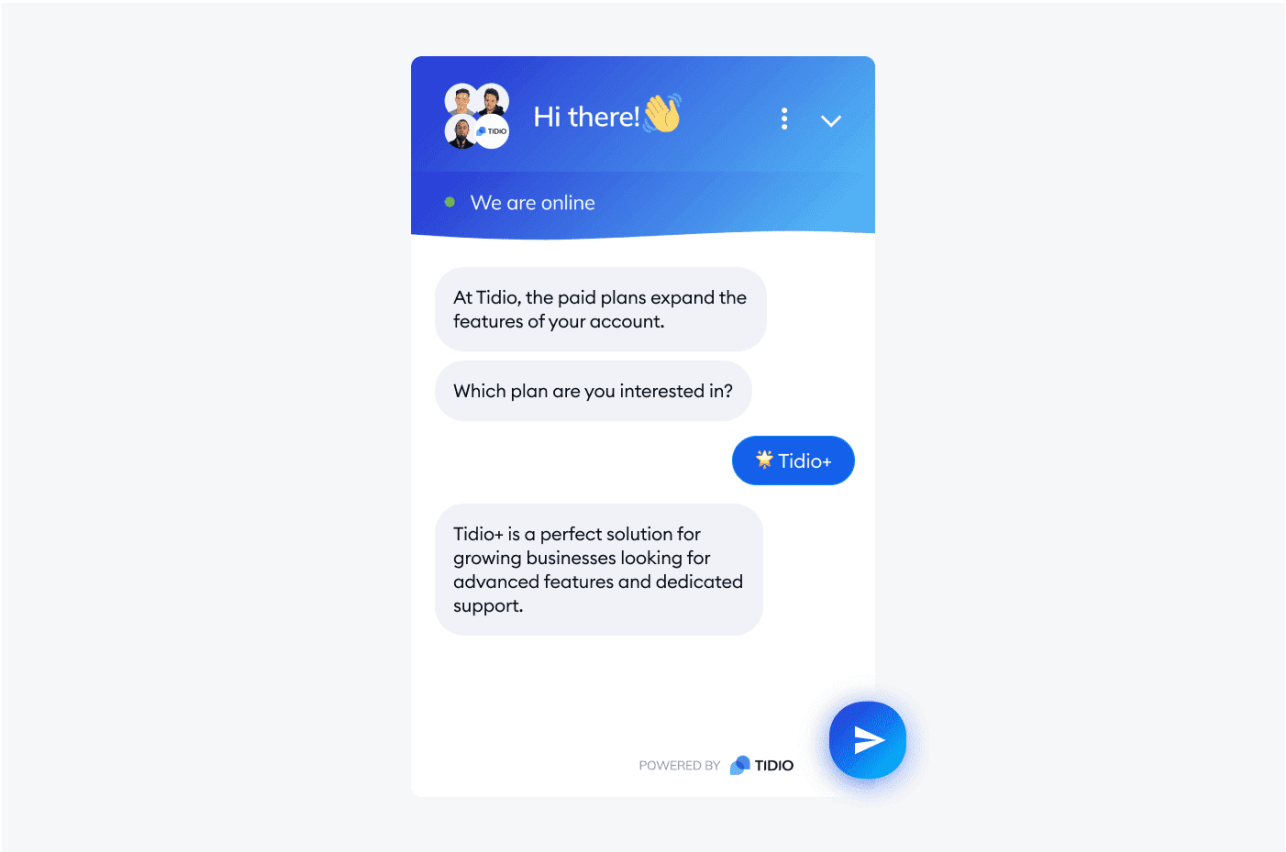


Figure 1. 4 Rule-based chatbot response

For example, you can train a rule-based chatbot to give an answer about the product pricing if a client uses any of the following words in their question:

How much does your product cost?

What is the cost of your product?

What is your pricing?

What are the prices available?

3. NLP bots

NLP chatbots use natural language processing (NLP) technology that helps them understand human language. In turn, it allows a bot to fully interpret the user's intent and respond to their queries in the best way possible.

To help you understand the capabilities this type of bot possesses, let's do a quick comparison with previously mentioned rule-based chatbots.

A more traditional, rule-based bot only understands the query provided by the context of a specific keyword. On the other hand, an NLP chatbot can understand the entire context of user intent.

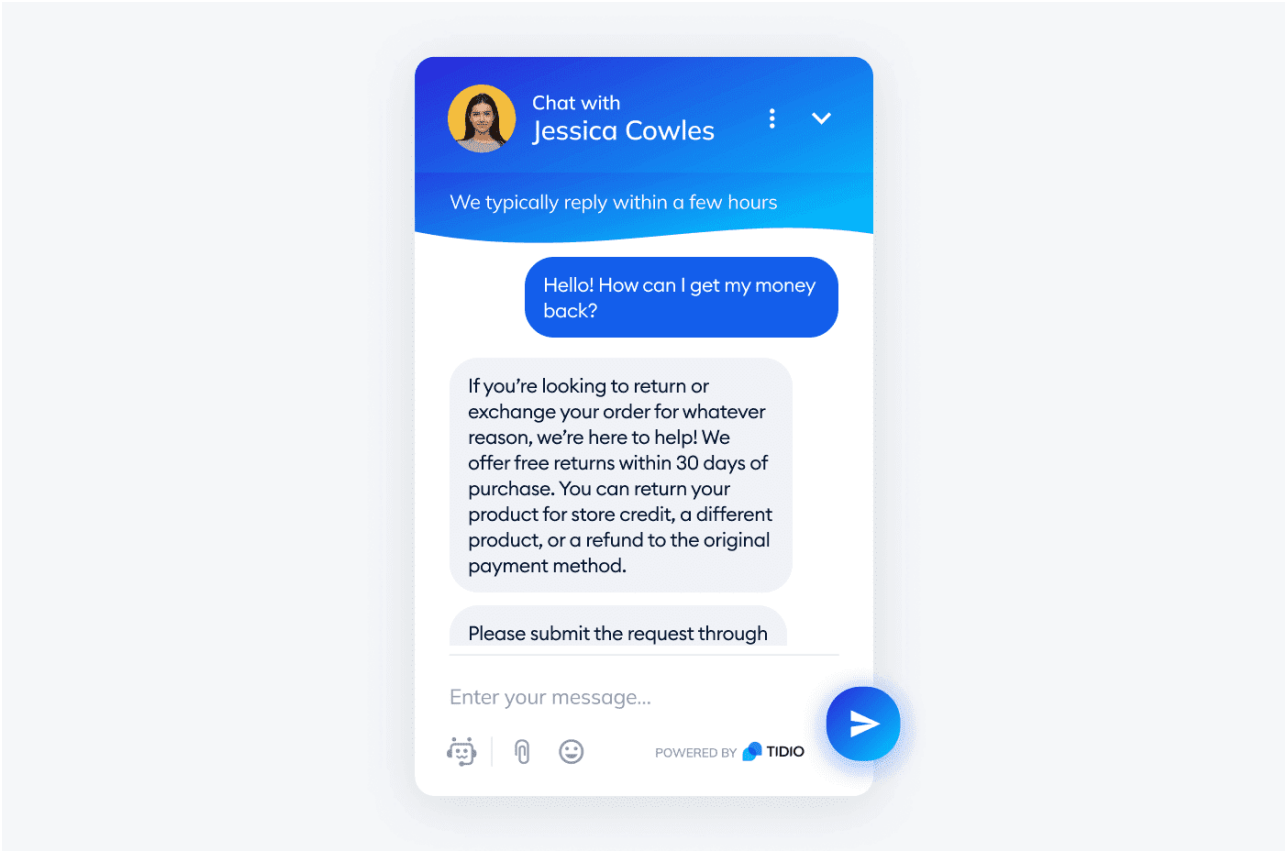


Figure 1. 5 NLP Chatbot of user intent

Say that a customer asks a rule-based chatbot to share shipping info related to your business. Since it's trained to provide an answer according to the "shipping" keyword, it's supposed to give you accurate info on this particular subject.

But what would happen if a person were to ask for something more specific, like "What is the shipping rate for FedEx", for example?

You guessed it—a rule-based bot would only be able to share the same answer as above, only recognizing the keyword “shipping”.

So, unlike rule-based bots, NLP chatbots will understand the more specific user input and will be able to come up with the appropriate reply, answering the user’s question in its entirety.

4. Machine learning bots

Contextual or machine learning (ML) chatbots are among the most advanced ones of the bunch. These bots use machine learning (ML) and AI technologies to memorize conversations made with specific users. By doing so, they have the potential to learn and develop even further with time.

In other words, the more user interactions the ML bot has, the more it’s capable of improving on its own.

A contextual chatbot can save any data from past conversations, such as the customer’s shipping address, billing info, and any other preferences made during the chat.

Say that a visitor has already interacted with an ML bot while making their order. If the same user wants to make the order again, the contextual chatbot can simply ask whether they want to use the same preferences already saved in its database.

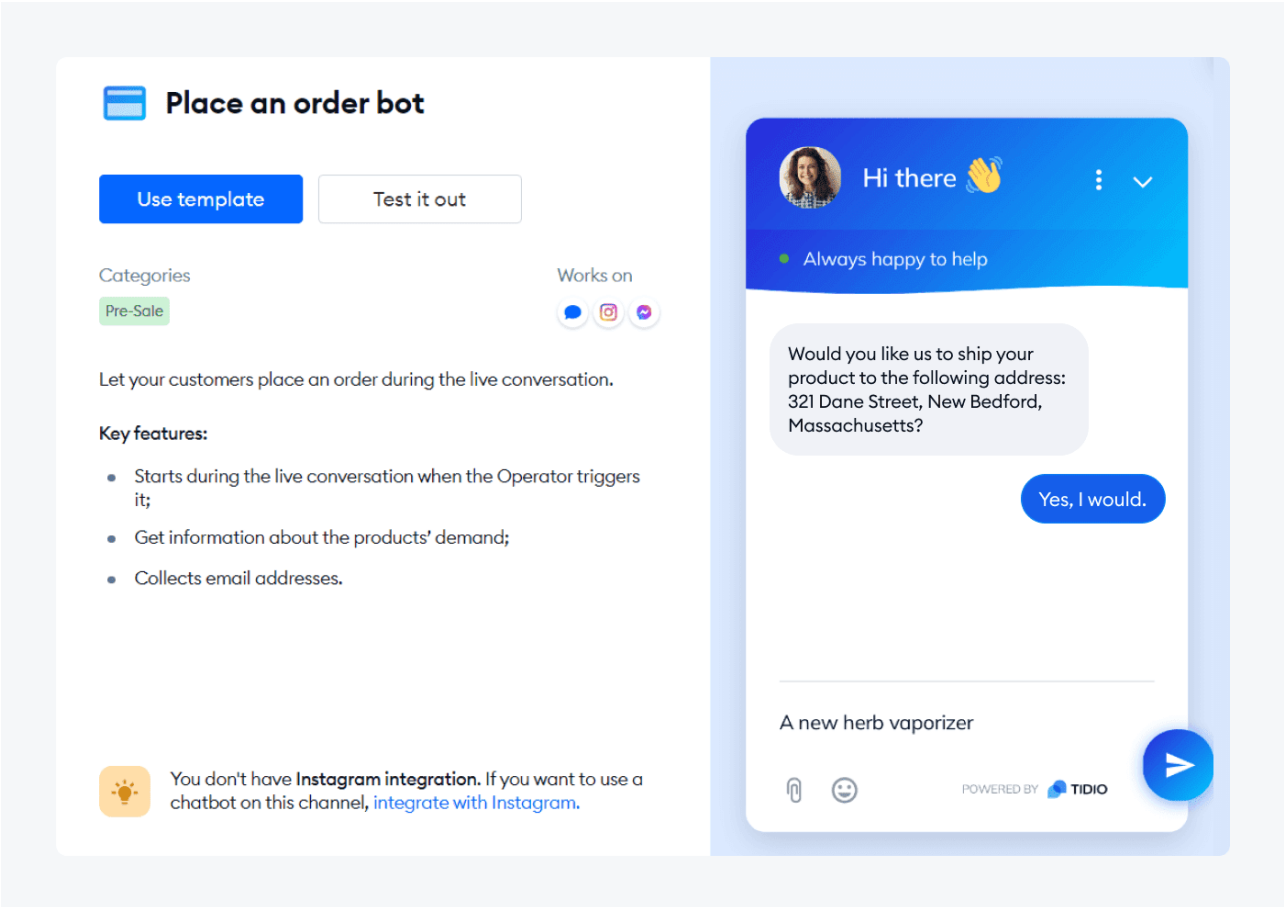


Figure 1. 6 ML chatbot interaction

In short, ML chatbots hold plenty of potential for personalization. As such, they can help your brand to make customers feel more valued, which is a huge plus for any business that wants to maintain customer loyalty.

5. Hybrid chatbots

Hybrid bots use a mix of AI and rule-based chatbot technologies to provide the best possible assistance to users and answer questions. As such, these are the most practical kinds of AI on our list.

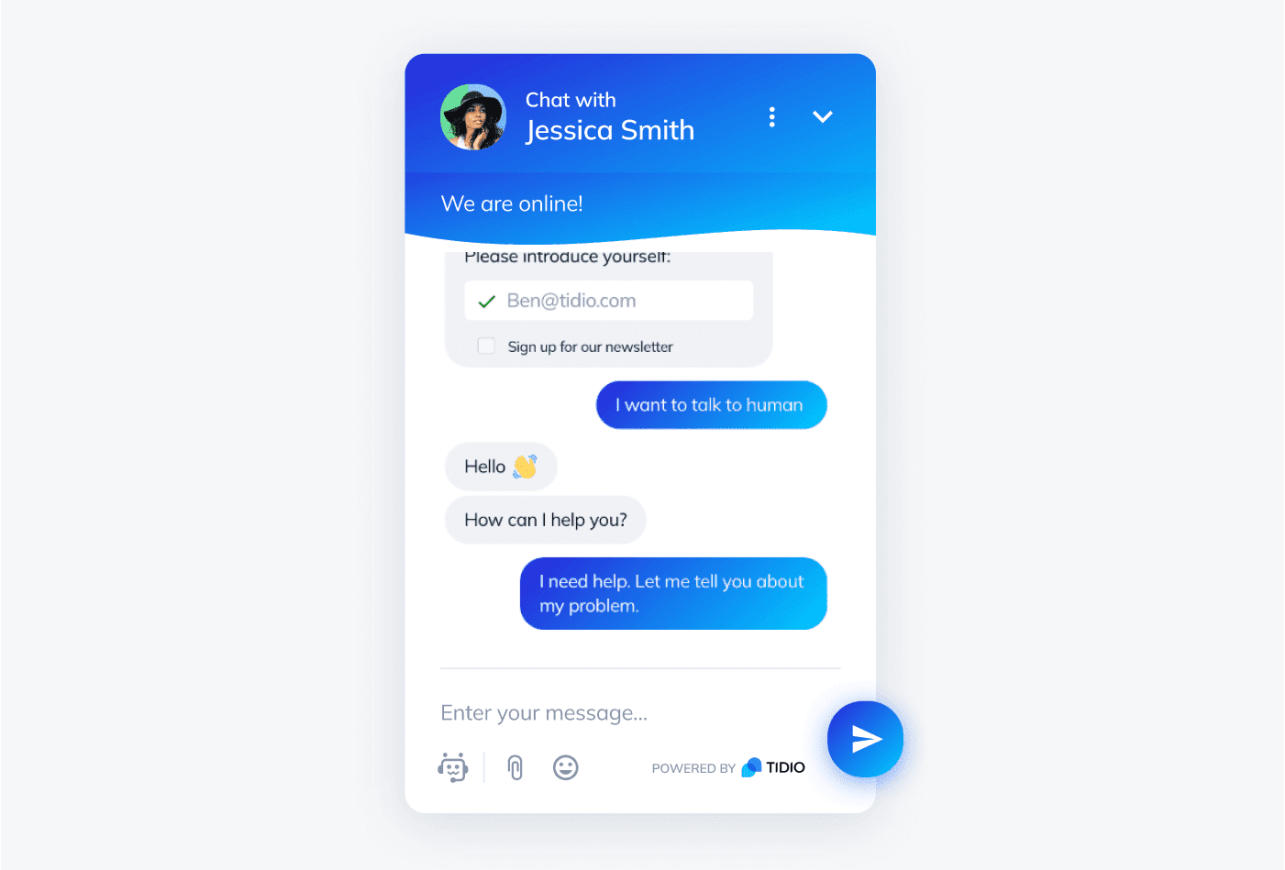


Figure 1. 7 Hybrid chatbot live interaction

What's also great about hybrid chatbots is that they allow you to use the chatbot and live chat technology simultaneously. This means that a bot can simulate real-life conversations based on queries up until the moment a more complex request is made. In this case, one of the service agents from your support team can replace the chatbot and quickly help a client out without disrupting the flow of conversation.

6. Voice bots

Simply put, voice chatbots represent a type of conversational AI that acts as a virtual assistant. They use speech-to-text and text-to-speech functions, as well as AI and natural language understanding (NLU) technologies to recognize user queries.

They are useful as they can offer answers to all sorts of questions and queries without a user having to type anything at all. It's quite convenient—all a person has to do is ask their inquiry or question out loud using their mobile device or computer.

The audio message made by the user is converted to text so that the bot can decipher the user intent. After searching the engines for the most accurate answer, a voice bot will convert said answer using text-to-speech. Finally, the person will receive their answer by voice within seconds.

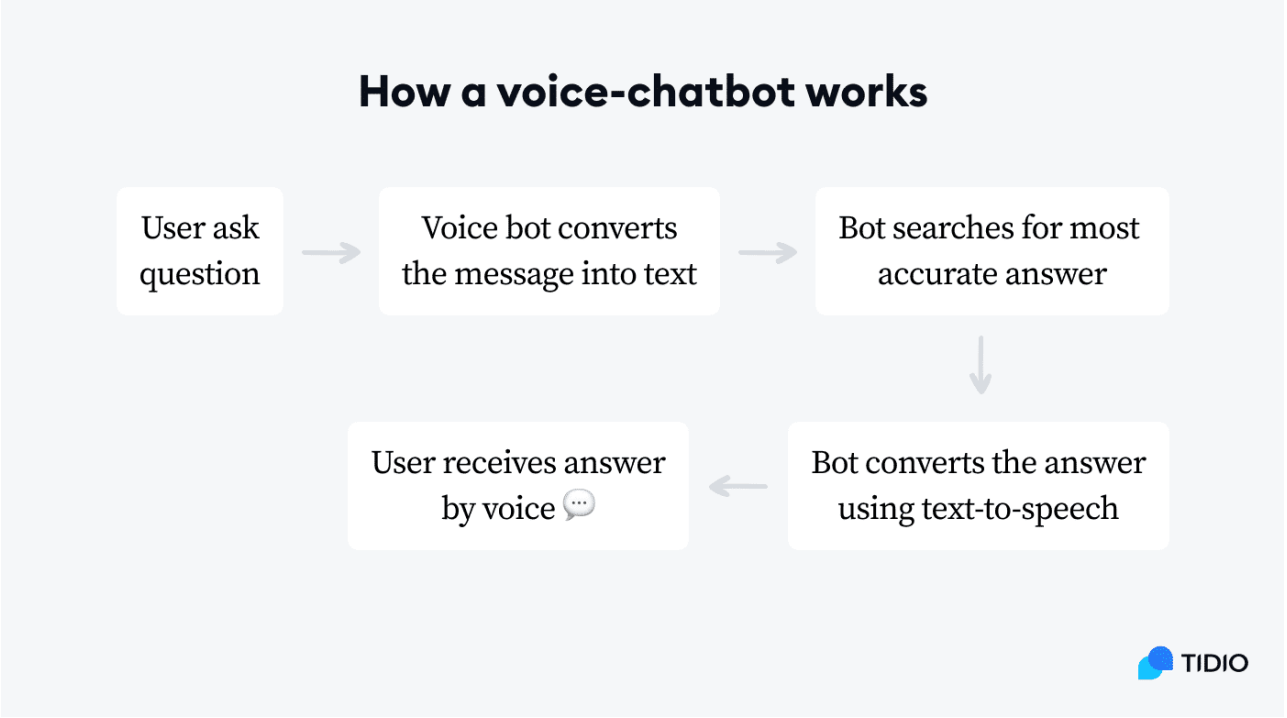


Figure 1. 8 Working of chatbot

It's also worth knowing that voice bot types can be further classified into:

- **Hybrid bots,** which are controlled by both text and voice
- **Voice-only bots,** which are only controlled by a voice

1.2 Artificial Intelligence

AI was coined by John McCarthy, an American computer scientist, in 1956 at The Dartmouth Conference where the discipline was born. Today, it is an umbrella term that encompasses everything from robotic process automation to actual robotics. It has gained prominence recently due, in part, to big data, or the increase in speed, size and variety of data businesses are now collecting. AI can perform tasks such as identifying patterns in the data more efficiently than humans, enabling businesses to gain more insight out of their data.

Types of artificial intelligence

AI can be categorized in any number of ways, but here are two examples.

- The first classifies AI systems as either weak AI or strong AI. Weak AI, also known as narrow AI, is an AI system that is designed and trained for a particular task. Virtual personal assistants, such as Apple's Siri, area form of weak AI.
- Strong AI, also known as artificial general intelligence, is an AI system with generalized human cognitive abilities so that when presented with an unfamiliar task, it has enough intelligence to find a solution. The Turing Test, developed by mathematician Alan Turing in 1950, is a method used to determine if a computer can actually think like a human, although the method is controversial.
- The second example is from Arend Hintze, an assistant professor of integrative biology and computer science and engineering at Michigan State University. He categorizes AI into four types, from the kind of AI systems that exist today to sentient systems, which do not yet exist. His categories are as follows

Type1: Reactive machines.

An example is Deep Blue, the IBM chess program that beat Garry Kasparov in the 1990s. Deep Blue can identify pieces on the chess board and make predictions, but it has no memory and cannot use past experiences to inform future ones. It analyzes possible moves -- its own and its opponent -- and chooses the most strategic move. Deep Blue and Google's Alpha GO were designed for narrow purposes and cannot easily be applied to another situation.

Type2: Limited memory.

These AI systems can use past experiences to inform future decisions. Some of the decision-making functions in autonomous vehicles have been designed this way. Observations used to inform actions happening in the not-so-distant future, such as a car that has changed lanes. These observations are not stored permanently.

Type3: Theory of mind.

This is a psychology term. It refers to the understanding that others have their own beliefs, desires and intentions that impact the decisions they make. This kind of AI does not yet exist.

Type4: Self-awareness

AI systems have a sense of self, have consciousness. Machines with self-awareness understand their current state and can use the information to infer what others are feeling. This type of AI does not yet exist.

Examples of AI technology

Automation is the process of making a system or process function automatically. Robotic process automation, for example, can be programmed to perform high-volume, repeatable tasks normally performed by humans. RPA is different from IT automation in that it can adapt to changing circumstances.

Machine learning is the science of getting a computer to act without programming. Deep learning is a subset of machine learning that, in very simple terms, can be thought of as the automation of predictive analytics. There are three types of machine learning algorithms: supervised learning, in which data sets are labeled so that patterns can be detected and used to label new data sets; unsupervised learning, in which datasets aren't labeled and a resorted according to similarities or differences; and reinforcement learning, in which data sets aren't labeled but, after performing an action or several actions, the AI system is given feedback. Machine vision is the science of making computers see. Machine vision captures and analyzes visual information using a camera, analog-to-digital conversion and digital signal processing. It is often compared to human eyesight, but machine vision isn't bound by biology and can be programed to see through walls, for example. It issued in range of applications from signature identification to medical image analysis. Computer vision, which is focused on machine-based image processing, is often conflated with machine vision.

Natural language processing (NLP) is the processing of human -- and not computer – language by a computer program. One of the older and best known examples of NLP is spam detection, which looks at the subject line and the text of an email and decides if it's junk. Current approaches to NLP are based on machine learning. NLP tasks include text translation, sentiment analysis and speech recognition.

Pattern recognition is a branch of machine learning that focuses on identifying patterns in data. The term, today, is dated.

Robotics is a field of engineering focused on the design and manufacturing of robots. Robots are often used to perform tasks that are difficult for humans to perform or perform consistently. They are used in assembly lines for car production or by NASA to move large objects in space. More recently, researchers are using machine learning to build robots that can interact in social settings.

1.3 AI applications

AI in health care. The biggest bets are on Improving patient outcomes and reducing costs. Companies are applying machine learning to make better and faster diagnoses than humans. On of the best known healthcare technologies is IBM Watson. It understands natural language and is capable of responding to questions asked of it. The system mines patient data and other available data sources to form a hypothesis, which it then presents with a confidence scoring schema. Other AI applications include chatbots, a computer program used online to answer questions and assist customers, to help schedule follow-up appointments or aiding patients through the billing process, and virtual health assistants that provide basic medical feedback.

AI in business. Robotic process automation is being applied to highly repetitive tasks normally performed by humans. Machine learning algorithms are being integrated into analytics and CRM platforms to uncover information on how to better serve customers. Chatbots have been incorporated into websites to provide immediate service to customers. Automation of

job positions has also become a talking point among academics and IT consultancies such as Gartner and Forrester.

AI in education. AI can automate grading, giving educators more time. AI can assess students and adapt to their needs, helping them work at their own pace. AI tutors can provide additional support to students, ensuring they stay on track. AI could change where and how

students learn, perhaps even replacing some teachers.

AI infinance.AI applied to personal finance applications, such as Mintor Turbo Tax, is upending financial institutions. Applications such as these could collect personal data and provide financial advice. Other programs, IBM Watson being one, have been applied to the process of buying a home. Today, software performs much of the trading on Wall Street.

AI in law. The discovery process, sifting through of documents, in law is often overwhelming for humans. Automating this process is a better use of time and a more efficient process. Startups are also building question-and-answer computer assistants that can sift programmed-to-answer questions by examining the tax ono my and onto logy associated with a database.

AI in manufacturing. This is an area that has been at the forefront of incorporating robots into the work flow. Industrial robots used to perform single tasks and were separated from human workers, but as the technology advanced that changed.

CHAPTER 2

TIMELINE AND LITERATURE REVIEW

2.1 Timeline

For a long time, researchers have been fascinated with the idea of being able to talk to computers and receive prompt responses. Even before the term “chatbot” was coined, they started working on machines in the 1950s that interact with humans through natural language. Many algorithmic breakthroughs occurred later in the 1980s and 1990s.

But the customers have become more aware of chatbots’ possibilities only in 2011 with the release of Apple’s Siri. Although its capabilities were limited at that time, functionality improved quickly in the following years, which can be attributed to the new competition in the current global chatbot market, led by major players like Google Assistant, Apple’s Siri, Microsoft’s Cortana and Amazon’s Alexa.

Reports suggest that there are more than 300,000 different chatbots on Facebook alone. Nearly 45% of end-users today prefer chatbots as the main means of communication for customer service inquiries. About 1.4 billion people in the world use messaging apps and are willing to talk to chatbots.

Estimates suggest that the use of chatbots will result in cost savings of more than $8 billion annually by 2022, up from $20 million in 2017. According to Google, chatbots will have a near human-level lingual ability by 2029. This post will present a brief history of chatbots from the 1950s until today.

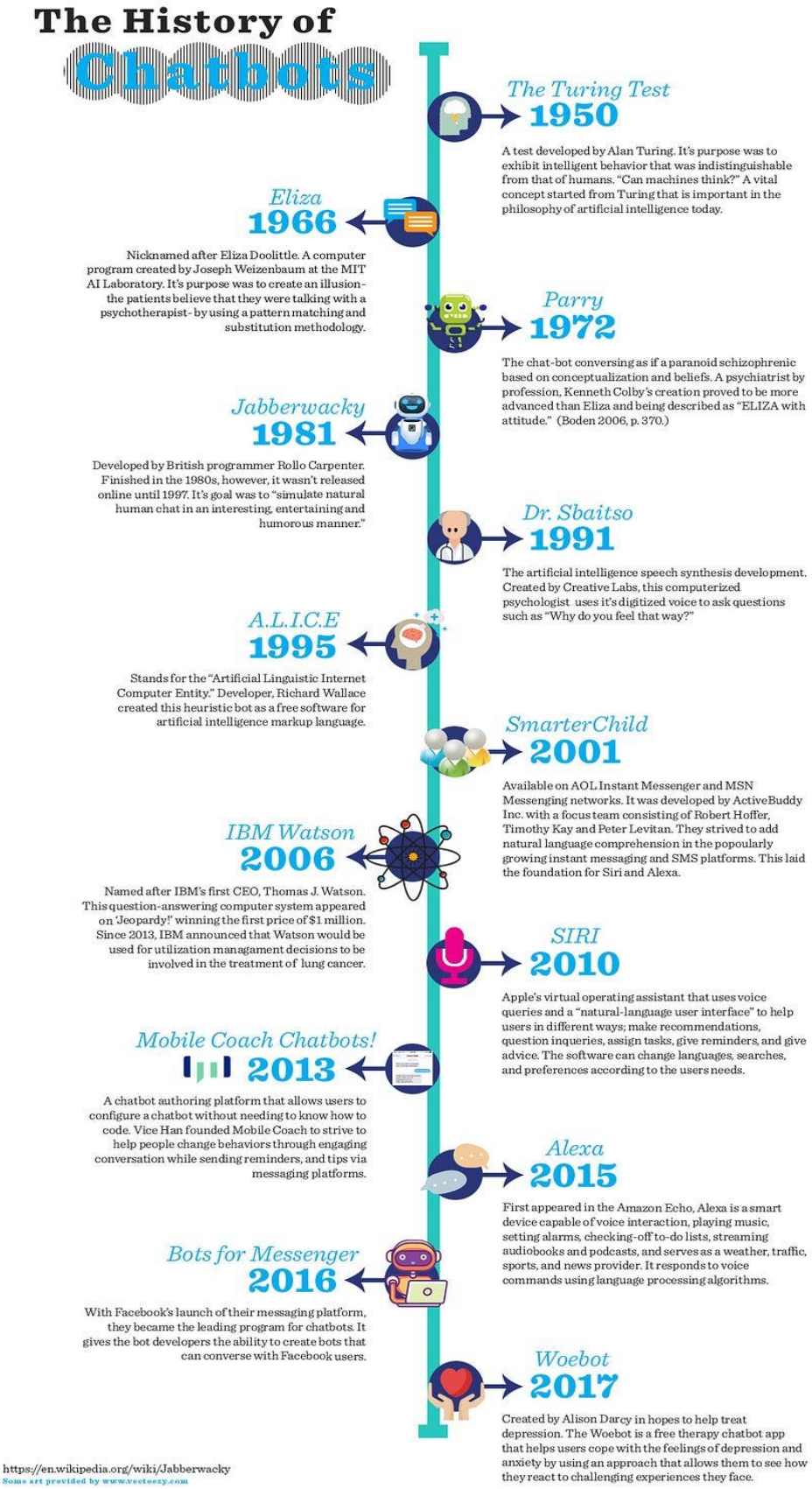


Figure 2. 1 History of chatbots

The history of chatbots will certainly amaze you regarding how far we have come since we started.

- **1950**

  The Turing test was developed by Alan Turing. It was a test of a machine's ability to exhibit intelligent behaviour equivalent to, or indistinguishable from, that of a human.

- **1966**

  Eliza, the first chatbot, was created by Joseph Weizenbaum, designed to be a therapist. It used to simulate a conversation by using a "pattern matching" and substitution methodology that gave users an illusion of understanding on the part of the bot.

- **1972**

  Parry, a computer program by psychiatrist and Stanford scientist Kenneth Colby, modelled the behaviour of a paranoid schizophrenic.

- **1981**

  The Jabberwocky chatbot was created by the British programmer Rollo Carpenter. It started in 1981 and launched on the internet in 1997. The aim of this chatbot was to "simulate natural human chat in an interesting, entertaining and humorous manner."

- **1985**

  The wireless robot toy, Tomy Chatbot, repeats any message recorded on its tape.

- **1992**

  Dr.Sbaitso, a chatbot created by Creative Labs for MS-DOS, “conversed” with the user as if it were a psychologist in a digitized voice. Repeated swearing and malformed input from the users caused Dr.Sbaitso to “break down” in a “PARITY ERROR” before it could reset itself.

- **1995**

  A.L.I.C.E (Artificial Linguistic Internet Computer Entity) was developed by Nobel Prize winner Richard Wallace

- **1996**

  Hex, developed by Jason Hutchens, was based on Eliza and won the Loebner Prize in 1996.

- **2001**

  Smarter child, an intelligent bot developed by Active Buddy, was widely distributed across global instance messaging and SMS networks. The original implementation quickly grew to provide instant access to news, weather, stock information, movie times, yellow pages listings, and detailed sports data, as well as a variety of tools (personal assistant, calculators, translator, etc.).

- **2006**

  The idea of Watson was coined from a dinner table; it was being designed to compete on the TV show “Jeopardy.” In its first pass, it could only get about 15 percent of answers correct, but later Watson was able to beat human contestants on a regular basis.

- **2010**

  Siri, an intelligent personal assistant, was launched as an iPhone app and then integrated as a part of the iOS. Siri is a spin-out from the SRI International Artificial Intelligence Center. Its speech recognition engine was provided by Nuance Communications, and Siri uses advanced machine learning technologies to function.

- **2012**

  Google launched the Google Now chatbot. It was originally codenamed "Majel" after Majel Barrett, the wife of Gene Roddenberry and the voice of computer systems in the Star Trek franchise; it was also codenamed as "assistant."

- **2014**

  Amazon released Alexa. The word "Alexa" has a hard consonant with the X, and therefore it can be recognized with higher precision. This was the primary reason Amazon chose this name.

- **2015**

  Cortana, a virtual assistant created by Microsoft. Cortana can set reminders, recognize a natural voice, and answer questions using information from the Bing search engine. It was named after a fictional artificial intelligence character in the Halo video game series.

- **2016**

  In April 2016, Facebook announced a bot platform for Messenger, including APIs to build chatbots to interact with users. Later enhancements done included bots being able to participate in groups, preview screens, and QR scan capability through Messenger's camera functionality to take users directly to the bot. In May 2016, Google unveiled its Amazon Echo competitor voice-enabled bot called Google Home at the company's developer conference. It enables users to speak voice commands to interact with various services.

- **2017**

  Woe bot is an automated conversational agent that helps you monitor mood, learn about yourself, and makes you feel better. Woe bot uses a combination of NLP techniques, psychological expertise (Cognitive-behavioural therapy [CBT]), excellent writing, and a sense of humour to treat depression.

- **2018**

  Endurance, My mother was diagnosed with aggressive Alzheimer's disease two years ago and having observed her sudden decline firsthand, I can tell you how difficult it is to watch someone with dementia struggle with even the most basic of conversational interactions.

- **2019**

  If you suffer from insomnia, as I do, you'll know that the feeling of almost suffocating loneliness — the idea that everyone else in the world is resting peacefully while your own mind betrays you with worries and doubts — is among the worst parts of not being able to sleep.

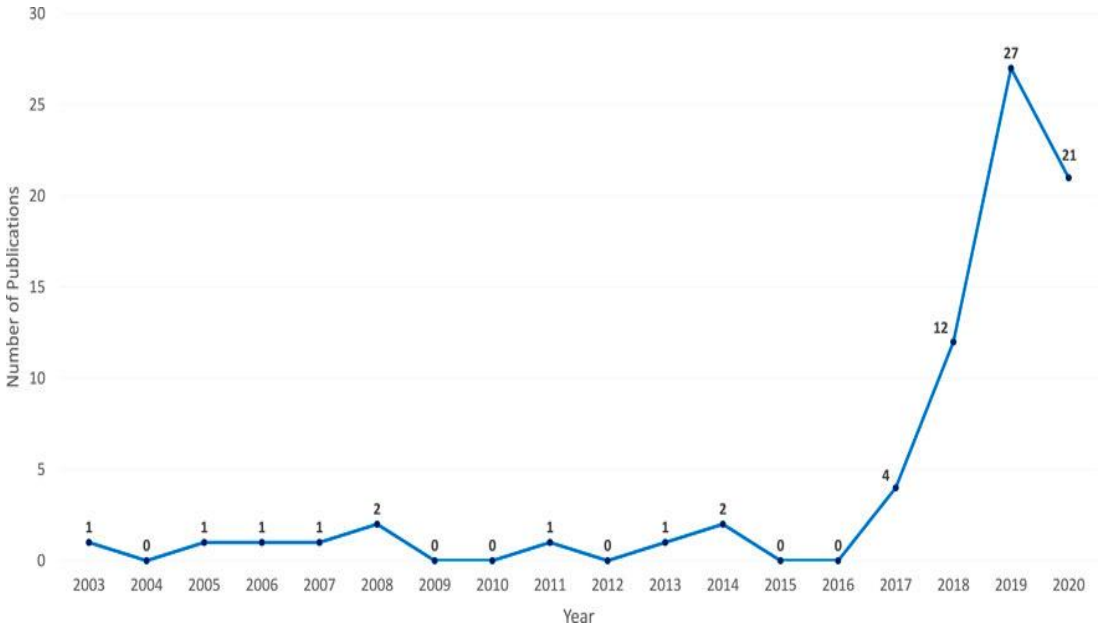


Figure 2. 2 search results in Scopus by year for "chatbot" or "conversation agent" or "conversational interface" as keywords from 2000 to 2019

As shown in Fig.2.2 according to Scopus, there was a rapid growth of interest in chatbots especially after the year 2016. Many chatbots were developed for industrial solutions while there is a wide range of less famous chatbots relevant to research and their applications.

2.2 Literature Survey

Purpose	Papers	Year of publications	Pros	Cons
Usefullness of chatbots	Bayan Abu Shawar and Eric Atwell, 2007 "Chatbots: Are they Really Useful?"	2007	• Significance of Chatbot	Machine like Response
Natural language processing	Journal for Computational Linguistics and Language	2007	• Benefit of natural language processing in any AIML chatbot • Detailed overview of Natural language processing mechanism	• Technical and obsolete problems in Nlp mechanism
Real time solution for Open learner model	Towards natural language negotiation of open learner models	2009	• Use of technology to enhance response.	• Limited information • Problem in integration
Intelligent tutorial system	Whitepaper for the Army's Science of Learning Workshop, Hampton	2006	• Detailed description about Python and AIML	• Only for chatbot based on Aiml
Aiml and Brain loading	AI Chat Bot with AIML Submitted by NanoDano	2015	• Effective way of installation.	• Slow brain loading

Figure 2. 3 Literature Survey

2.3 Natural Language Processing

Natural Language Processing (NLP) is the study of letting computers understand human languages. Without NLP, human language sentences are just a series of meaning less symbol strong computers. Computers don't recognize the words and don't understand the grammars. NLP can be regard as a "translator", who will translate human languages to computer understandable information.

Traditionally, users need to follow well-defined procedures accurately, in order to interact with computers. For example, in Linux systems, all commands must be precise. A single replace

a one character or even a space can have significant difference. However, the emergence of NLP is changing the way of interacting. Apple Siriand Microsoft Cortanahave made it possible to give command in every day languages and is changing the way of interacting

2.4 Machine Learning.

Machine Learning (ML) is an area of computer science that "gives computers the ability to learn without being explicitly programmed". The parameter of the formulas is calculated from the data, rather than defined by the programmer. Two most common usage of ML is Classification and Regression. As shown in figure1 , Classification means to categorize different types of data, while Regression means to find away to describe the data. Basic ML program will have two stages, ***fitting*** and ***predicting.*** In the fitting stage, the program will be given a large set (at least thousands) of data. The program will try to adjust its parameter based on some statistical models, in order to make it "fit" the input data best. In the predicting stage, the program will give a prediction for a new input based on the parameters it just calculated out. For example, the famous Iris flower dataset contains the measurement of several features of three different species of flowers, such as the length of sepals and petals. A well-defined ML program can learn the pattern behind this feature and give prediction accordingly

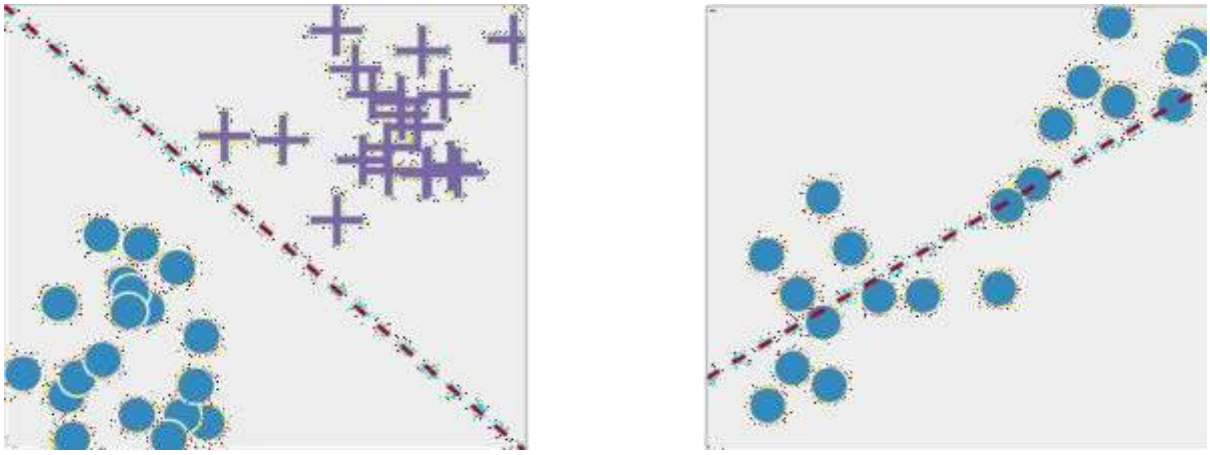


Figure 2.4 Classification and Regression

CHAPTER 3

EXISTING SYSTEM AND PROBLEM IDENTIFICATION

3.1 Existing System:

i. **Task-Oriented Dialogue Systems**:

These chatbots are designed for specific tasks and aim to complete them efficiently.They engage in short conversations with users to gather information and achieve the desired outcome.Examples include customer support bots, appointment schedulers, and order tracking assistants.

ii. **Chatbots (Social Dialogue Systems):**

These chatbots focus on human engagement and social interactions.They aim to create natural conversations and enhance user experiences.Chatbots can be used in various domains such as e-commerce, hospitality, healthcare, and more1.

iii. Architectures of Dialogue Systems:

a. **Rule-Based Systems**:

These systems decompose input text based on specific criteria and keywords. They follow predefined rules to transform and assemble responses.Early examples include ELIZA and PARRY. More recent rule-based chatbots include the DB pedia Chatbot1.

b. **Corpus-Based Systems:**

These systems use data from corpora (collections of text).They can be further classified into:

- **Information Retrieval Systems:**

  Retrieve relevant information from a dataset.

- **Generation-Based Systems:**

  Generate responses based on patterns learned from data.These systems often employ natural language processing (NLP) techniques.

iv. Existing Chatbot Frameworks:

Several frameworks facilitate chatbot development like Microsoft Bot Framework, Wit.AI, Dialog flow, IBM Watson, BotKit.ai, Rasa, Chatter Bot

- **Wit.ai**:

  Another product by Facebook Engineering and is also coming as an SDK for NodeJS, Python, Ruby and Go.

- **Dialog flow**:

  In my opinion, Dialog flow is the most famous **chatbot** framework and this helps the developers to create a simple chatbot without any programming knowledge. But this is a Google Service runs on the Google cloud platform. This means the users do not have control over their chatbots.

- **IBM Watson**:

  IBM Watson is also a chatbot service as dialog flow provided by IBM. These chatbots can be built easily but they have only limited features and also need to pay to expand.

- **Rasa:**

  Rasa can be considered as an open source machine learning framework for automating conversations. This framework supports integrating the bot to various chat applications easily. Developers just need a piece of knowledge in programming into some extent to develop bots with Rasa. But on the other hands this framework gives full control over the chatbot. This chatbot can setup in our own server and use as we like. And also as this is an open source framework we can extend their given functionalities as we like.Chatbots have evolved significantly over time, from simple rule-based systems to sophisticated AI-powered models. Their applications continue to expand, enhancing communication between humans and machines Problem Identification-.

3.2 Problem Identification:

Changing of requirements as an industrial project to build a product, we must follow the requirement from the user. However, because the project's goal is to be used by the business team, but it is responsible by the technical team, the requirement changed a lot in the middle after a meeting with the business team. The business team want a simple bot that can give recommendation immediately. We had to archive what we had done before and build a new one.

- **Lacking of training data.** The quantity and quality of training data is critical to the performance to a machine learning model. However, because some confidential and privacy reasons, the business team cannot provide enough data for us, and we had to make up data by our own. For the machine learning model, we generate some fake data based on our daily life experience, which is really biased, although with a good accuracy on the faked at.

- **Unstable API version.** Because API service we are using are still under development, and we cannot fix to a version for the API, the API may changes overtime. Moreover, there are inconsistencies between the APIs and their documents or sample codes.

- **Not familiar with the PHP language and .NET framework.** None of the three of such as previous knowledge with PHP language. Programming in a new language in such a huge framework is quite challenging for us at the beginning of the project. However, when we comes to the later phases, we are more used to that.

CHAPTER 4

REQUIREMENTS AND SPECIFICATION

4.1 Hardware Requirements:

Processor	: Pentium IV (minimum)
Hard Disk	: 40GB
RAM	: 256MB(minimum)

4.2 Software Requirements:

Operating System	: Windows or Linux
Technology	: PYTHON, AI, ML

CHAPTER 5

GENERAL ARCHITECTURE

- An appropriate chatbot architectural design is useful to the study of chatbots and the aspiring chatbots creator. Therefore, several architectural designs have been proposed .the authors propose an architecture, but it is specific for Rule-based chatbots. Similarly, present an architecture, which is specific for Retrieval-based chatbots. Moreover, they do not include a design the chatbot's Knowledge Base is connected to other databases and information systems that give answers to the user's queries. However, there is an absence of vital services like sentiment analysis and ambiguity handling. An interesting design presents an architecture structured in layers. However, it is quite abstract as it does not provide essential details for each layer. Also, it does not discuss how the user's input is analysed and which are the components of the Dialog Management Component. Besides , present a simplistic design that lacks many essential details; for example, there is no information about the backend. A chatbot is proposed that generates emotionally consistent responses, but the architecture design focuses only on the Response Generation Module. Hahm works on architecture too, but it gives no architectural design. The architecture in serves the integration of big data with the Knowledge Base of a chatbot. However, it concerns only Rule-based models implemented with AIML.A detailed architectural design is presented for a chatbot that supports students. This design lacks a sentiment analysis component the detailed architecture of Microsoft XiaoIce is presented. It includes many interesting characteristics, but it is specific to XiaoIce the research focuses on the layers of campus and how they are connected to a chatbot, but it gives no architectural design of the chatbot.
- In this section, we compose an architectural design that is general and, at the same time, includes all the details we consider crucial. We describe the main components and the individual parts they contain and focus on what we consider most important. The developer can decide which parts he/she is going to implement depending on the type the chatbot.
- there is a summary of the architectural features of the chatbots of the before mentioned studies and the proposed architecture.

5.1. User Interface Component

- The operation of the chatbot begins when it receives the user's request through an application using text or speech input, such as a messenger application like Facebook, Slack, WhatsApp, WeChat, Viber, or Skype.

5.2. User Message Analysis Component

- The User Interface Controller drives the user's request to the User Message Analysis Component to find the user's intention and extracts entities following pattern matching or machine learning approaches. The user's message can be retained as plain text, which keeps all the grammatical and syntactical structures of the input unchanged or processed by Natural Language Processing (NLP) .
- More precisely, through their input to the chatbot, users express their purpose, which is the intent. The chatbot must understand the user's intent and perform the required actions. Different user inputs trigger different intents and may include parameters, called entities, to determine precise details about them .
- Certain cognitive services can be linked to the User Message Analysis Component to improve accuracy.

- The proposed architecture consists of a User Interface Component, a User Message Analysis Component, a Dialog Management Component, a Backend, and a Response Generation Component.

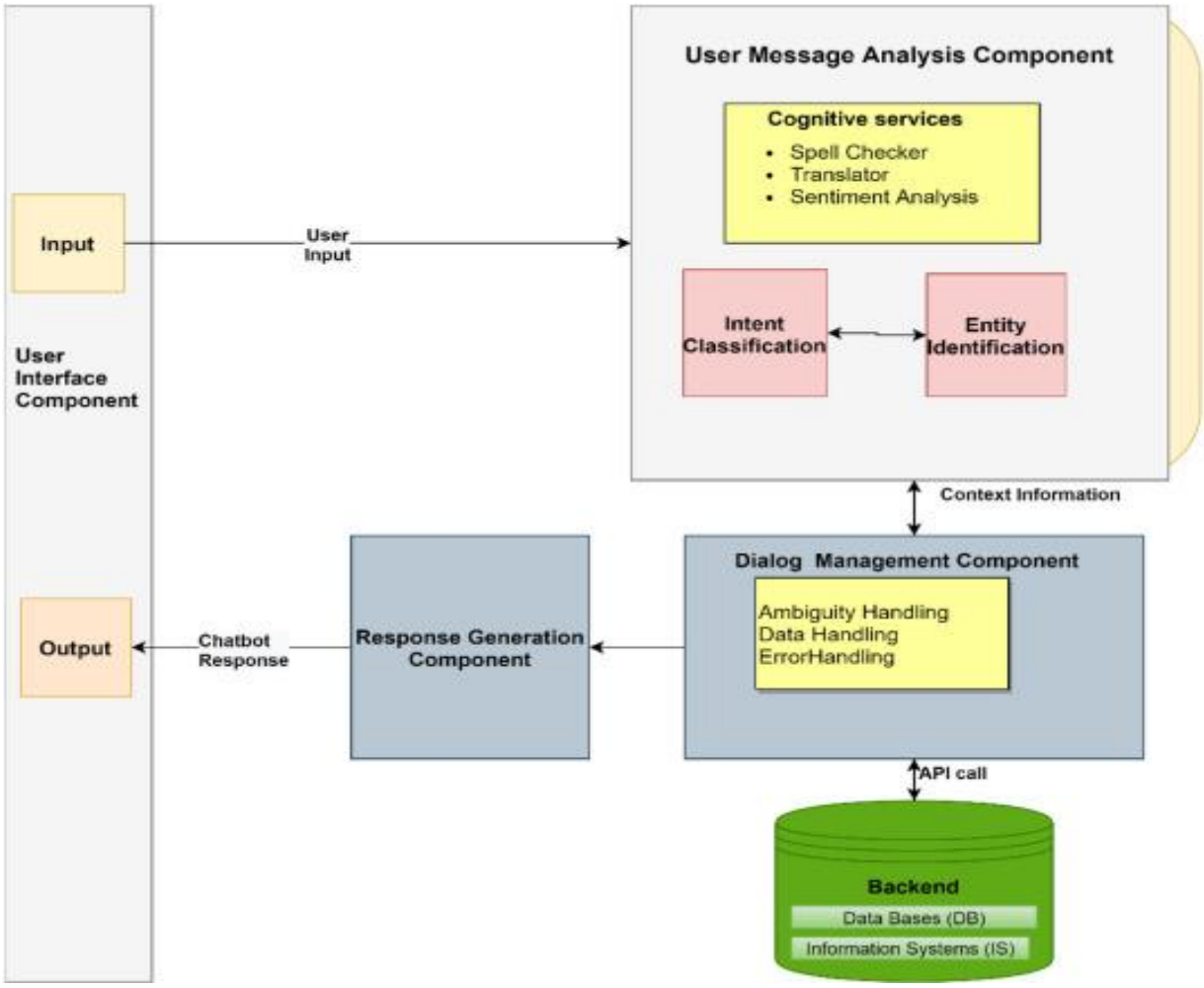


Figure 5. 1 General Chatbot architecture

- A spell checker corrects the user spelling mistakes as the "purified" input usually results in better intent identification.
- A machine translator is used in the case of multilingual chatbot users. The user's language is identified and translated into the language of the chatbot's NLU.
- Sentiment analysis is applied to the user input to see how satisfied or irritated the user seems to be. In some cases, a real person may need to connect to the discussion if the user looks very frustrated.

- Sentiment analysis detects a positive or negative opinion within a text. User reactions were recorded while discussing with chatbots that used different sentiments. Positive chatbot responses, regardless of the user's input sentiment, were preferred. Therefore, chatbots should analyse the sense of the input and learn only from the positive ignoring inputs with a negative or harmful view (Bird et al., 2019). A complete expression–emotion mapping database that maps words and expressions into emotions support an emotionally capable chatbot . Cloud-based sentiment analysis services are already top-rated and can also be used in chatbots to obtain sentiment analysis from the user input .

5.3. Dialog Management Component

- The Dialog Management Component controls and updates the conversation context. It keeps the current intent and the identified entities until that point of the conversation. If the chatbot is unable to collect the necessary context information, it asks for additional context information from the user to fulfil missing entities. It also asks follow-up questions after the intent is recognized.
- The Dialog Management Component typically includes the following modules.
    - **Ambiguity Handling.** This module gives answers when the chatbot cannot find the intent from the user's request or if no input is recognized. The chatbot may indicate that it did not have an answer, ask for clarification, start a new discussion or give a general answer that covers a variety of issues so that the user is satisfied even if he has asked the most unforeseeable question.
    - **Data Handling.** User information is stored in a file. In this way, the chatbot can modify its answers depending on the user giving the impression of being more intelligent.
    - **Error Handling.** The Error Handling Module copes with unexpected errors to ensure the proper chatbot operation.
- After the intent identification, the chatbot proceeds to the next actions, which may be information retrieval from the Backend or responding to the user. In the first case, the control flow handle remains inside the Dialog Management Component, which uses it to determine the next action. In contrast, in the latter case, the control passes to the Backend.

5.4. Backend

- The chatbot retrieves the information needed to fulfil the user's intent from the Backend through external APIs calls or Database requests. Once the appropriate information is extracted, it is forwarded to the Dialog Management Module and then to the Response Generation Module.
- When Rule-based chatbots are used, there is a Knowledge Base (KB). It includes a list of hand-written responses that correspond to the user's inputs. For a chatbot to stand firm, the Knowledge Base must cover a wide variety of user's queries and contain a variety of responses to the same user input to avoid redundancy of replies.
- A Relational Data Base (RDB) may be used so that the chatbot can recall past conversations, making in this way the communication more consistent and relevant. This

approach brings consistency and precision to the dialog as it allows the chatbot to access previous information history.

- Creating the Knowledge Base of a chatbot is a necessary but often demanding and time-consuming task because it is manually developed. A method was proposed that automatically builds the Knowledge Base of a new chatbot from an existing one. Also, a program transforms a corpus to the AIML Knowledge Base of a chatbot. Often, Rule-based chatbots complete their Knowledge Base by asking the user questions and encouraging him for long conversations.
- The Knowledge Base may also support Ontologies (Semantic Nets) like Wordnet or Open Cyc. The chatbot updates the conversation state and searches the nodes of the Knowledge Graph to make connections for the concepts used in the conversation. Also, in S. and Balakrishnan (2018), the AIML Knowledge Base was enriched from a big data Knowledge Base with the chatbot's connection to the big data environment.
- The use of language tricks in a chatbot's Knowledge Base makes it more human-like. These tricks simulate people's behavior in a conversation, such as stereotyped responses, typing errors, the manner of typing, the existence of a personality, and even irrational responses.

5.5. Response Generation Component

- The Response Generation Component produces responses using one or more of three available models: Rule-based, Retrieval based, and Generative-based models.

- The Rule-based model selects the response from a set of rules without generating new text responses. The Dialog Management component passes the placeholder values that may be needed to fill the template for the response to the Response Generation Module. Rule-based models use a Knowledge Base (KB) organized with conversational patterns (Ramesh et al., 2017).
- The Retrieval-based model is more flexible as it selects the most suitable response with the check and analysis of available resources using APIs.
- The Generative model uses Natural Language Generation (NLG) to respond in a human-like natural language based on the last and previous inputs. However, developing and training such a model is challenging because it needs an extensive set of data for training to establish a fruitful conversation (Hien et al., 2018). When the training corpus is small, grammatical errors are made, particularly in long sentences.

- There have also been hybrid approaches that compare the retrieved to the generated response and choose the better.
- When the chatbot produces a response, it presents it to the user and waits until it has feedback.

CHAPTER 6

DEVELOPMENT

6.1. The process of creating a chatbot

- The creation of a chatbot starts with planning the aims, procedures, and user requirements of it. Developing the chatbot using a programming language or a chatbot development platform follows, and afterward, the chatbot functionalities are tested locally. The chatbot is then published on an online website or a data center, and it is connected to one or more channels to send and receive messages. At this point, proper integration of a chatbot in an application is crucial, and there are three methodologies to do this: integration using API, manual integration, or third-party integration. Finally, evaluation is conducted by gathering data during a fixed time after deployment so that developers can detect errors and use the input to enhance the chatbot's performance and capabilities.
- The selection of algorithms or platforms to build chatbots depends on what the chatbot should provide to the users and what category it falls into. The proper selection may result in the benefits of connectivity, efficiency, quick and uncomplicated production revisions, and minimal effort for the designer. Note that a chatbot is considered more efficient when the user can connect directly without downloading and installing it.
- A chatbot can be developed using programming languages like Java and Python or a chatbot development platform that may be commercial or open-source (Nayyar, 2019). Open-source platforms include RASA, Bot kit, Chatterbot, Pandorabots, Botlytics and Microsoft Bot Framework while commercial platforms include Botsify, Chat fuel, Manychat and Flow XO. Some NLU cloud platforms (Braun et al., 2017) powered by machine learning are Google Dialog Flow , Facebook wit.ai , Microsoft LUIS , IBM Watson Conversation (*IBM Watson*, 2019), Amazon Lex , and SAP Conversation AI .
- Open-source platforms make their code available, and the developer can have full control of the implementation. Although commercial platforms do not give full control to developers, developers are usually benefiting from data efficient to train their chatbots. Moreover, commercial platforms facilitate easy integration to other products of the platform. For example, a chatbot built with Dialog flow may be easily interfaced with Google Assistant and other Google devices. Other facilities that many development platforms may offer include:

    - Easy addition of intents and easy testing of intent triggering (Greyling, 2020)

- Context management. Most of the development platforms, although they use technically different approaches, support context management. Proper context management is critical for intent classification. For example, if the user utterance is "In Kavala, today?" it is most likely that the response will come as the result of an unclassified intent. However, if a context like "weather forecasting" is activated, a successful answer will report the weather forecast in Kavala on the day of the conversation.
- Support of both predefined and user-defined entities and automatic detection of entity values.
- Support searching in collections of knowledge documents, like FAQs or articles, to find responses.

- Individual platforms may also offer additional facilities. For example, Google Dialog flow supports events, both platform-defined and custom-defined. Instead of user input, events may also trigger intents. For example, an event occurred at a specific time may trigger an intent to alert the user that his/her break time has end.
- Finally, development platforms may be combined with programming languages if the project requirements demand so.

- Therefore, development platforms may be an integral part of implementing a chatbot. Of course, before choosing a commercial platform, one should consider the costs involved in using it.

6.2. Training a chatbot

- There are many available corpora for training chatbots. Chitchat neural network approaches are usually trained on movie scripts corpora or dialogs from web platforms. A chatbot trained on such datasets, formed by discussions between different speakers, often lacks a specific personality (Zhang et al., 2018). Moreover, various problems may arise when using a dialog corpora with human dialog examples to retrain a chatbot (Atwell &Shawar, 2003). When the dialog corpus is not large enough for training, the chatbot's responses may have syntactic or semantic errors. Therefore, methods have been proposed to pre-train a chatbot using a large non-dialog corpus, and to retrain it using a small dialog corpus.
- Sometimes, dialog corpora and machine learning approaches are used to generate the AIML rules and therefore train Rule-based chatbots. Moreover, various versions of a chatbot in different languages have been created automatically.

- Chatbot developers usually keep files from conversations after the chatbot is deployed. It helps them to understand the user requests better and improve the chatbot, which is continuously learning through the analysis of these conversations. New training examples are extracted from the conversations a chatbot takes part in when the conversation goes on well, or feedback is requested case.

6.3. Connect the chatbot to a channel

- Over the last few years, chatbots have gained popularity and are used in various messenger applications or connected to websites.

- Facebook is a popular social network for chatbots, mainly for performing transactions or services rather than discussing with users. Facebook Messenger chatbots are usually part of group chats and perform functions like providing statistics around a sports match, creating a music playlist, or giving Smart Replies like informing about business hours or making a reservation without leaving the chat window. Therefore, Facebook chatbots primarily support users in a secretarial way rather than communicate with them.

- Skype chatbots, similarly to Facebook ones, are commonly used in group chats for functional purposes. Skype chatbots become more interactive when they give users the alternative of voice chat instead of typing.

- Twitter takes a new approach for its chatbots, offering a platform for businesses to connect with their customers and provides an enjoyable rather than a transactional interaction.

- Companies use Slack chatbots internally to maximize efficiency, boost connectivity, or perform tasks. There are two categories of Slack chatbots, those that send notifications and those that perform specific transactions initiated by the user.

- The benefits of chatbot encapsulation in messaging applications are included: The interaction with the chatbot can quickly be distributed across the user's social network without leaving the messaging app, which ensures the user's identity. Chatbots can be inserted into group chats or exchanged like any contact, whereas a notification system re-engages inactive users. Moreover, payment systems are built into the messaging application and can be used safely and efficiently (Klopfenstein et al., 2017).

- In contrast to the channels mentioned above, website based chatbots provide developers with full control. On the website, one can say precisely how the chatbot works, including its purpose, user interface, and experience. Moreover, the users can engage in conversation without leaving the current page, having an easy way to ask questions.

6.4. Conversational mode of chatbots

- When a chatbot takes the persona of a famous person, the interaction with its user seems to be improved. Freudbot, a chatbot developed with AIML, was used in distance and online education, and it proved to be a useful teaching and learning tool. It was programmed according to the conversational rules related to turn-taking, providing answers with implications that invited the user to request more information making the conversation longer.
- Chatbots should also offer users a more accessible and realistic experience to ask about their privacy settings in applications or websites.
- Moreover, chatbots may draw interest and involve users in activities such as completing questionnaires. Responding to chatbot questions would be a fun and desirable solution to somebody because it does not take as much time as traditional questionnaires.
- Users and chatbots interact with each other, and it is interesting to examine "Symbiotic agency", a term that was initially used for the proxy agency where users and software enact in human–technology interaction. Currently, the term expands this concept of the proxy system to consider both how technology mediates somebody's thoughts, beliefs, and attitudes, and how human agency impacts the use of technological products .

6.5. Other implementation considerations

- When designing a chatbot, we must first determine the goals that it will serve. Based on the objectives, we will evaluate the primary character of the chatbot, for example, if we need a generic, cross-domain, or closed-domain chatbot. In the first two cases, we will probably need to use NLP techniques. Before deciding, we should take into account whether the necessary data for chatbot training is available. Conversely, for closed-domain chatbots, the use of a scripting language may be preferable. In general, it is not easy to find suitable training data for specific purposes.

- On the other hand, a closed-domain chatbot based on scripting language can if properly designed, effectively guide the user towards achieving specific goals. For example, a vocabulary learning assistant can easily redirect the conversation when it fails to classify the user's intention by saying, 'I did not understand exactly. Do you want me to help you study some vocabulary?'. As long as the discussion remains in the context of vocabulary learning, the dialogs are limited and can be precisely designed so that the result is satisfactory. Also, such a chatbot can be an initial approach that will help in collecting data from users' phrases and will enrich the experience of its manufacturer so that if it is deemed useful to proceed to the second phase using NLP technology.
- Whichever approach we follow, we must anticipate that our software will be measuring the user experience so that, based on the measurements, we will be able to make appropriate improvements. However, if we choose NLP technology, we must consider that the choice of an implementation platform that supports cognitive abstraction will provide us, to some extent, automatic updates on the ever-evolving aspects of Artificial Intelligence (Shaw, 2020).
- Also, it is an excellent practice to provide the ability to deal with deadlocks with our staff members to avoid unpleasant user experiences (*What Are the Key Considerations When Implementing a Chatbot*, 2020).
- A decision that is also important is related to the languages we need to support. Special care is required in voice communication as not all Text-To-Speech (TTS) and Automatic Speech Recognition (ASR) systems support all languages (*5 Core Considerations For Choosing Your Chatbot |by Ruth Zive |Chatbots Magazine*, 2020).

CHAPTER 7

ADVANTAGES & DISADVANTAGES

7.1 Advantages

1. Accessible anytime:

- I'm sure most of you are always kept on hold while operators connect you to a customer care executive. On an average people spend around 7 minutes until they are assigned to a person. Gone are the frustrating days of waiting in a queue for the next available operative. They are replacing live chat and other forms of slower contact methods such as emails and phone calls. Since chatbots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break. This improves your customer UX and helps you rank highly in your sector. Another advantage of this instant response is that you can also skill fully craft your chatbot to maintain your image and brand.

2. Handling Capacity:

- Unlike humans who can only communicate with one human at a time, chat bots can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered immediately.
- Imagine you own a restaurant, and you have a good reputation for your food of which most of your revenues come from delivery. As the demand keeps rising, you will have more customers to take orders from but very few staff to attend them all. Having a chatbot would eliminate such problem and cater to each and every person and ensure that no order is missed. Companies like Taco Bell and Dominos are already using chatbots to arrange delivery of parcels.

3. Flexible attribute:

Chatbots have the benefit that it can quite easily be used in any industry. Unlike other products where you have to do a lot of development and testing to change platforms, chatbots are relatively easy to switch. One has to just train the bot by giving the right conversation structure and flow to switch its current field or industry.

- Or if there is a lot of back and forth between two sections of the industry say

customers support and sales, then you could have custom built presets which would already have the conversation flow and structure to carry out the interactions with the user.

4. Customer Satisfaction:

- Humans are bound to change of emotions. Chatbots, on the other hand, are bound by some rules and obey them as long as they're programmed to. They will always treat a customer in the perfect way no matter how rough the person is or how foul language the person uses.
- Not everyone orders the same food every day, people's choices may change every day. In this case, it can use your order history to make suggestions for the next order, learn your address details and much more. Customers love this smooth interaction and want all their transactions to be as simple as possible.

5. Cost Effective:

- Hiring a human for a job is never a cheap affair, and it will be expensive if your revenue are not high or sales targets are not met and would create havoc in the business. Due to the boundaries of human beings, a single human can only handle one or two people at the same time. More than that would be extremely tough for the employee.
- Chatbots could help solve this age-old problem. As one chatbot is equal to loads of employees, it can easily communicate with thousands of customers at the same time. We would only need a handful of people to jump into conversations sometimes when necessary. Hence, it would drastically bring down the expenses and bring about a steep rise in revenue and customer satisfaction.

6. Faster On boarding:

- Before you want to accomplish a task, you first must learn how to work on the task and complete it. Only then will they be considered fit for the job. There is a continuous teaching involved in every level of hierarchy the employee will go through. Also, there will be a lot of change in the employees, some stay, some get fired, some more join in etc.
- What we want to say is, employees will change; it's a fact. And this would require you to allot a lot time of your employees into grooming the new joinees. Chatbots could eliminate that time to almost zero, but provide a very clean and easy to understand conversation flow and structure that needs to be maintained by the chatbot. No doubt there will be change
- this too, but it will rather take a fraction of your time to resolve as compared to human employees.

7.2 Disadvantages-

1. They Provide Potentially Lower Customer Service Quality

- Think of the last time that a company's customer service impressed you. More than likely, you were not speaking to a chatbot during that experience.Calling or messaging a chatbot can be one of the most frustrating ordeals for a customer who needs help. To receive the same generic, perhaps non-relevant answer repeatedly is infuriating and demoralizing.

2. A Chatbot Does Not Provide a Personalized Experience

- Chatbots can't sense when a customer is in distress. They lack empathy and personalization. The lack of emotional reassurance or human connection can cause a customer to seek service elsewhere, which is exactly what you don't want to happen.

3. They Can't Handle Or Resolve Complex Customer Problems

- Chatbots can't handle non-basic problems that don't fit into their preset capabilities. They also can't calm down angry or other highly emotional customers. Finally, they can't resolve an intense situation or make important tail-end decisions, like whether or not to offer a refund.

4. They Can Be Hard To Install Or Deploy

- Chatbots can take a long time to set up and deploy into the workforce. If they are based on flow builders, you will have to build the chatbot flows from the ground up yourself or hire someone to do it. This process could take months to finish.

5. They Lose Customer Insights

- When you leave chatbots to deal with your customers, you may miss out on insights and feedback that those customers express. A chatbot will not report back to you if the customer says the company is not meeting a particular need, for example. It's just not part of the program's job.

6. They Can Be Difficult to Set Up

- A chatbot can be time-consuming and tricky to set up. It's a complex program that requires constant maintenance and attention, during both initial deployment and long-term use.

7. A Chatbot May Not Transmit Data Securely

- Some chatbots do not transmit data securely and may be vulnerable to hacking. This is unacceptable for transferring private information and user data, especially in the medical field with HIPAA violations being a major concern, or financial services that require PCI compliance.

CHAPTER 8

TECHNOLOGY

ABOUT PYTHON:

Dating from 1991, Python is a relatively new programming language. From the start, Python was considered a gap-filler, a way to write scripts that “automate the boring stuff” (as one popular book on learning Python put it) or to rapidly prototype applications that will be implemented in one or more other languages.

However, over the past few years, Python has emerged as a first-class citizen in modern software development, infrastructure management, and data analysis. Itis no longer a back-room utility language, but a major force in web application development and systems management and a key driver behind the explosion in bigdata analytics and machine

Perfect for IT, Python simplifies many kinds of work, from system automation to working in cutting-edge fields like machine learning.

Python is easy to learn. The number of features in the language itself is modest, requiring relatively little investment of time or effort to produce one’s first programs. Python syntax is designed to be readable and straight forward. This simplicity makes Python an ideal teaching language, and allows newcomers to pick it up quickly. Developers spend more time thinking about the problem they’re trying to solve, and less time thinking about language complexities or deciphering code left by others.

Python is broadly used and supported. Python is both popular and widely used, as the high rankings in surveys like the Tiobe Index and the large number of GitHub projects using Python attest. Python runs on every major operating system and platform, and most minor ones too. Many major libraries and API-powered services have Python bindings or wrappers, allowing Python to interface freely with those services or make direct use of those libraries. Python may not be the fastest language, but what it lacks in speed, it makes up for inversatility.

Python is not a “toy” language. Even though scripting and automation cover a large chunk of Python’s use cases (more on that below), Python is also used to build robust, professional-quality software, both as stand alone applications and as webservices.

What is Python used for?

The most basic use case for Python is as a scripting and automation language. Python isn't just are placement for shell scripts or batch files, but is also used to automate interactions with web browsers or application GUIs or system rivisioning and configuration in tools such as Ansible and Salt. But scripting and automation represent only the tip of the icebergwithPython.

- Python is used for general application programming. Both CLI and cross-platformGUIapplicationscanbecreatedwithPythonanddeployedasself-containedexecutables. Python doesn't have the native ability to generate a standalone binary from a script, but third-party packages like cx_Freeze or PyInstaller can be used to accomplish that.

- Python is used for data science and machine learning. Sophisticated data analysis has become one of fastest moving areas of IT and one of Python's star use cases. The vast majority ofthelibrariesusedfordatascienceormachinelearninghavePythoninterfaces, making the language the most popular high-level command interface to for machine learning libraries and other numerical algorithms.

- Python is used for web services and RESTful APIs. Python's native libraries and third-party web frameworks provide fast and convenient ways to create everything from simple REST APIs in a few lines of code, to full-blown, data-driven sites. Python's latest versions have powerful support for asynchronous operations, allowing sites to handle up to ten soft thousands of requests per second with the right libraries.

- Python is used for meta programming. In Python, everything in the language is , including Python modules and libraries themselves. This allows Python to work as a highly efficient code generator, making it possible to write applications that manipulate their own function and have the kind of extensibility that would be difficult or impossible to pull off in other languages.

- Python is used for glue code. Python is often described as a “glue language,” meaning it can allow disparate code (typically libraries with C language interfaces) to inter operate. Its use in data science and machine learning is in this vein, but that’s just one incarnation of the general idea.

The Python language’s pros and cons

Python syntax is meant to be readable and clean, with little pretense. A standard “helloworld” in Python 3.xis nothing more than:

- Print (“Hello world!”)
- Python provides many syntactical elements that make it possible to concisely express many common program flows. Consider a sample program for reading lines from a text file into a list object, stripping each line of its terminating newline character along the way:

- With open(‘myfile.txt’)as my_file:
- file_lines = [x.strip (‘\n’) for x in my_file]
- The with/as construction is a “context manager,” which provides an efficient way to instantiate a given object for a block of code and then dispose of it outside of that block. In this case, the object in question is my_file, instantiated with the open()function. This takes the place of several lines of boilerplate to open the file, read individual lines from it, then close it up.

- The [x … for x in my_file] construction is another Python idiosyncrasy, the “list comprehension. ”It allows a given item that contains other items (here, my_ file and the lines it contains) to be iterated through, and to allow each iterated element (that is, each x) to be processed and automatically appended into a list.

- You could write such a thing as a formal for… loop in Python, much as you would inanother language. The point is that Python has a way to economically express thingslike loops that iterate over multiple objects and perform some simple operation oneach element in the loop, or work with things that require explicit instantiation anddisposal. Constructions like this allow Python developers to balance terseness andreadability.

- Python’s other language features are meant to complement common use cases. Most modern object types—Uni code strings, for instance—are built directly into the language. Data structures—like lists, dictionaries (i.e., hash maps),tuples (for storing immutable collections of objects), and sets (for storing collections of unique objects)—are available as standard-issue items.

- Like C#, Java, and Go, Python has garbage-collected memory management, meaning he programmer doesn’t have to implement code to track and release objects. Normally garbage collection happens automatically in the background, but if that poses a performance problem, it can be triggered manually or disabled entirely.

- An important aspect of Python is its dynamism. Everything in the language, including functions and modules themselves, are handled as objects. This comes at the expense of speed (more on that below), but makes it far easier to write high-level code. Developers can perform complex object manipulations with only a few instructions, and even treat parts of an application as abstractions that can be altered if needed.

- Python’s use of significant white space has been cited as both one of Python’s best and worst attributes. The indentation on these code lines how above isn’t just for readability; it is part of Python’s syntax. Python interpreters will reject programs thatdon’tuseproper indentation to indicate control flow.

- Syntactical white space might cause noses to wrinkle, and some people do rejectPython out of hand for this reason. But strict indentation rules are far less obtrusive inpractice than they might seem in theory, even with the most minimal of code editors, and the end result is code that is cleaner and more readable.

Python2 versus Python3

- Python is available in two versions, which are different enough to trip up many new users. Python2.x, the older “legacy” branch, will continue to be supported (i.e. receive official updates) through 2020, and it might even persist unofficially after that. Python 3.x, the current and future incarnation of the language, has many useful and important features not found in 2.x, such as better concurrency controls and a more efficient interpreter.

- Python 3 adoption was slowed for the longest time by the relative lack of third-partylibrary support. Many Python libraries supported only Python 2, making it difficult toswitch. But over the last couple of years, the number of libraries supporting onlyPython 2 has dwindled; most are now compatible with both versions. Today, there are few reasons against using Python 3.

CHAPTER 9
CONCLUSION AND FUTURE SCOPE

9.1 Conclusions

Chatbots are the new Apps! It can be used for many things in our day to day life even for very small things . As many students for lacking the information about the particular college and university at the time of admissions , because of that some students may not land in the college which they are expecting . At that time this kinds of chatbots come to rescue by providing information from which the student is suffering. This project is designed for solving the basic queries about the college and student can ask questions related to university like course, fees and can get their answers in no time.

Future Scope

Future potential for this kind of study looks bright. Chatbots are extremely useful in areas like mental health care and individualized customer service because of further improvements in emotion recognition models that can result in even more sympathetic and context-aware answers. We can make our chatbot more inclusive and accessible by extending the number of languages and dialects it can understand through developments in multilingual natural language processing. Real-time data analytics and predictive modeling integration may also improve the chatbot's capacity for financial advising services, enabling it to offer even more precise and useful guidance. A continuing focus will be on finding new methods to employ offline capabilities for a variety of tasks and enhancing security protocols to safeguard user data. Conversational AI has a bright future ahead of it, with room to grow in a number of areas, including personalized customer care, mental health services, and financial advising. Chatbots can provide persons with emotional issues with tailored and compassionate counsel, therefore boosting mental well-being and prompt treatment. Chatbots can now be more inclusive and accessible to a worldwide audience thanks to the limitless potential of multilingual natural language processing. The incorporation of predictive modeling and real-time data analytics into

financial advising services offers accurate and practical financial advice, enabling users to make well-informed decisions. Another important area of research is offline functionality, which expands the use of chatbots beyond internet access to help tourists in isolated areas or with spotty connectivity. There are limitations to what has been currently achieved with chatbots. The limitations ofdata processing and retrieval are hindering chatbots to reach their full potential. It is not thatwe lack the computational processing power to do so. However, there is a limitation on“How” we do it. One of the biggest examples is the retail customer market. Retail customersare primarily interested in interacting with humans because of nature of their needs. They don’t want bots to process their needs and respond accordingly.

10. TRAINING

```
import random
import json
import numpyas np
import pickle

import nltk
from nltk.stemimport WordNetLemmatizer

from tensorflow.keras.modelsimport Sequential
from tensorflow.keras.layersimport Dense, Activation, Dropout
import tensorflow

lemmatizer = WordNetLemmatizer()

intents = json.loads(open("intents.json").read())

words = []
classes = []
documents = []
ignore_letters = ["?","/",".",";","}","{","[","]"]

for intent in intents["intents"]:
for pattern in intent["patterns"]:
word_list = nltk.word_tokenize(pattern)
words.extend(word_list)
documents.append((word_list,intent["tag"]))
if intent["tag"] not in classes:
classes.append(intent["tag"])

words = [lemmatizer.lemmatize(word) for word in words if word not in ignore_letters]
```

```
words = sorted(set(words))

classes = sorted(set(classes))

pickle.dump(words, open("words.pkl", "wb"))
pickle.dump(classes, open("classes.pkl", "wb"))

training = []
output_empty = [0] * len(classes)

for document in documents:
    bag = []
word_patterns = document[0]
word_patterns = [lemmatizer.lemmatize(word.lower()) for word in word_patterns]
for word in words:
bag.append(1) if word in word_patternselse bag.append(0)

output_row = list(output_empty)
output_row[classes.index(document[1])] = 1
training.append([bag, output_row])


random.shuffle(training)
training = np.array(training, dtype=object)

train_x = list(training[:, 0])
train_y = list(training[:, 1])

model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(32, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))


sgd = tensorflow.keras.optimizers.legacy.SGD(learning_rate=0.01, decay=1e-6, momentum=0.9,
nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])

hist = model.fit(np.array(train_x),np.array(train_y), epochs=500, batch_size=9, verbose=1)
model.save('chatbotmodel.h5', hist)
print('done')
```

11.Chatbot Source Code

```
import random
import json
import numpyas np
import pickle
import pyttsx3
import nltk
import re
import pyautoguias p
import AppOpeneras a
import webbrowseras web
import time
from nltk.stemimport WordNetLemmatizer

from tensorflow.keras.modelsimport load_model

lemmetizer = WordNetLemmatizer()

intents = json.loads(open('intents.json').read())
words = pickle.load(open('words.pkl', 'rb'))
classes = pickle.load(open('classes.pkl', 'rb'))
model = load_model('chatbotmodel.h5')

def clean_up_sentence(sentence):
sentence_word = nltk.word_tokenize(sentence)
sentence_word = [lemmetizer.lemmatize(word) for word in sentence_word]
return sentence_word

def bag_of_words(sentence):
sentence_words = clean_up_sentence(sentence)
    bag = [0] * len(words)
for w in sentence_words:
for i, word in enumerate(words):
if word == w:
                bag[i] = 1
return np.array(bag)

def predict_class(sentence):
    bow = bag_of_words(sentence)
    res = model.predict(np.array([bow]))[0]
    ERROR_THRESHOLD = 0.25
results =[[i, r] for i, r in enumerate(res) if r > ERROR_THRESHOLD]

results.sort(key=lambda x:x[1], reverse=True)
return_list = []
for r in results:
return_list.append({'intents': classes[r[0]], 'probability': str(r[1])})
return return_list

def get_response(intents_list, intents_json):
    tag = intents_list[0]['intents']
```

```
list_of_intents = intents_json['intents']
for iin list_of_intents:
if i['tag'] == tag:
            result = random.choice(i['responses'])
break
    return result


engine = pyttsx3.init()
voice=engine.getProperty('voices')
rate = engine.getProperty('rate')
volume = engine.getProperty('volume')
engine.setProperty('voice', voice[1].id)

def talk(out):
engine.setProperty('volume', 100)
engine.setProperty('rate', 150)
engine.say(out)
engine.runAndWait()

#print("GO! BOT IS RUNNING!")

def get_res(message):
#message = input("")
ints = predict_class(message)
    res = get_response(ints, intents)
return str(res)

'''
while True:

    message = input(">")

ints = predict_class(message)
    res = get_response(ints, intents)
    #return str(res)
    print(res)
'''
```

11.1 App

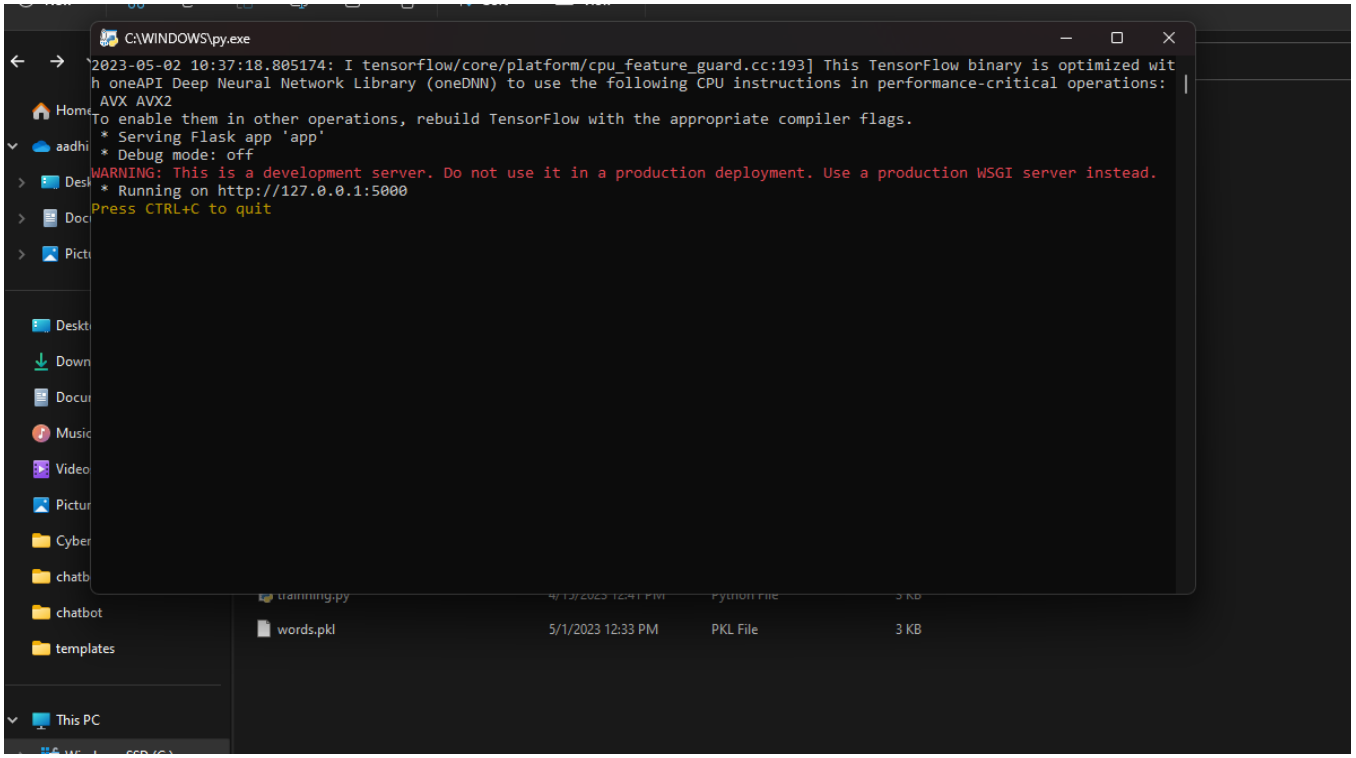
```
import chatbot
from flask import Flask, render_template, request

app = Flask(__name__)
app.static_folder = 'static'

@app.route("/")
def home():
```

```
return render_template("index.html")

@app.route("/get")
def get_bot_response():
userText = request.args.get('msg')

return str(chatbot.get_res(userText))

if __name__ == "__main__":
app.run()
```

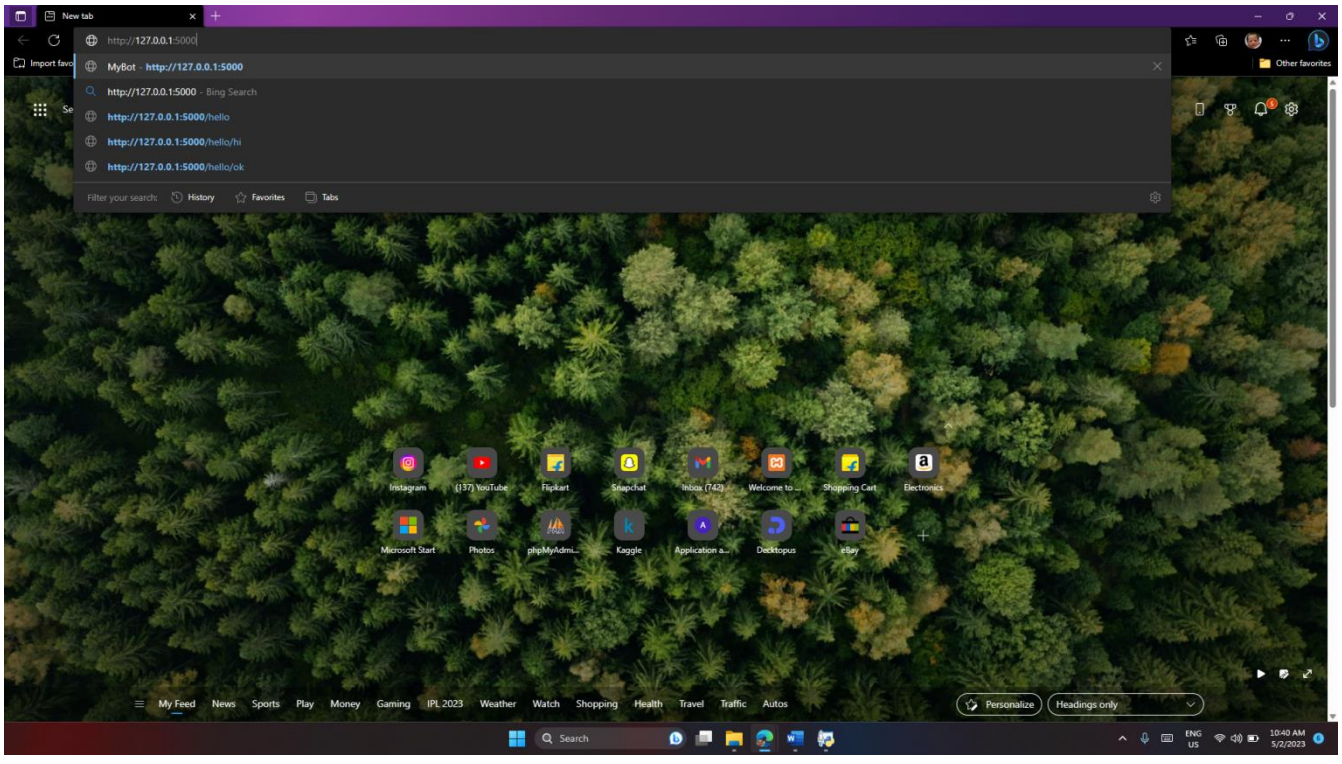
11.2 Intents

```
{   "intents":[
 {
  "tag": "greeting",
  "patterns": [
    "Hi",
    "How are you?",
    "Is anyone there?",
    "Hello",
    "Good day",
    "What's up",
    "how are ya",
    "heyy",
    "whatsup",
    "??? ??? ??"
  ],
  "responses": [
    "Hello!",
    "Good to see you again!",
    "Hi there, how can I help?"
  ],
  "context_set": ""
},
 {
  "tag": "goodbye",
  "patterns": [
    "cya",
    "see you",
    "byebye",
    "See you later",
    "Goodbye",
    "I am Leaving",
    "Bye",
```

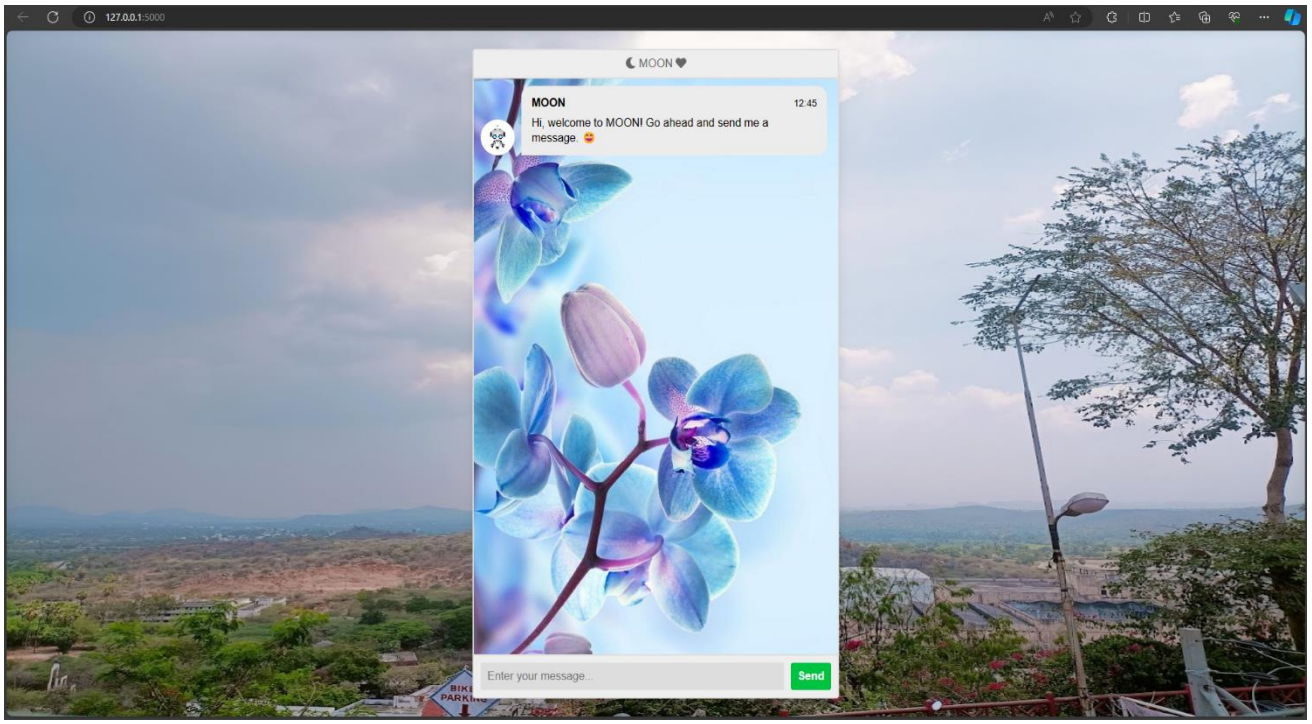
```
  "context_set": ""
},
```



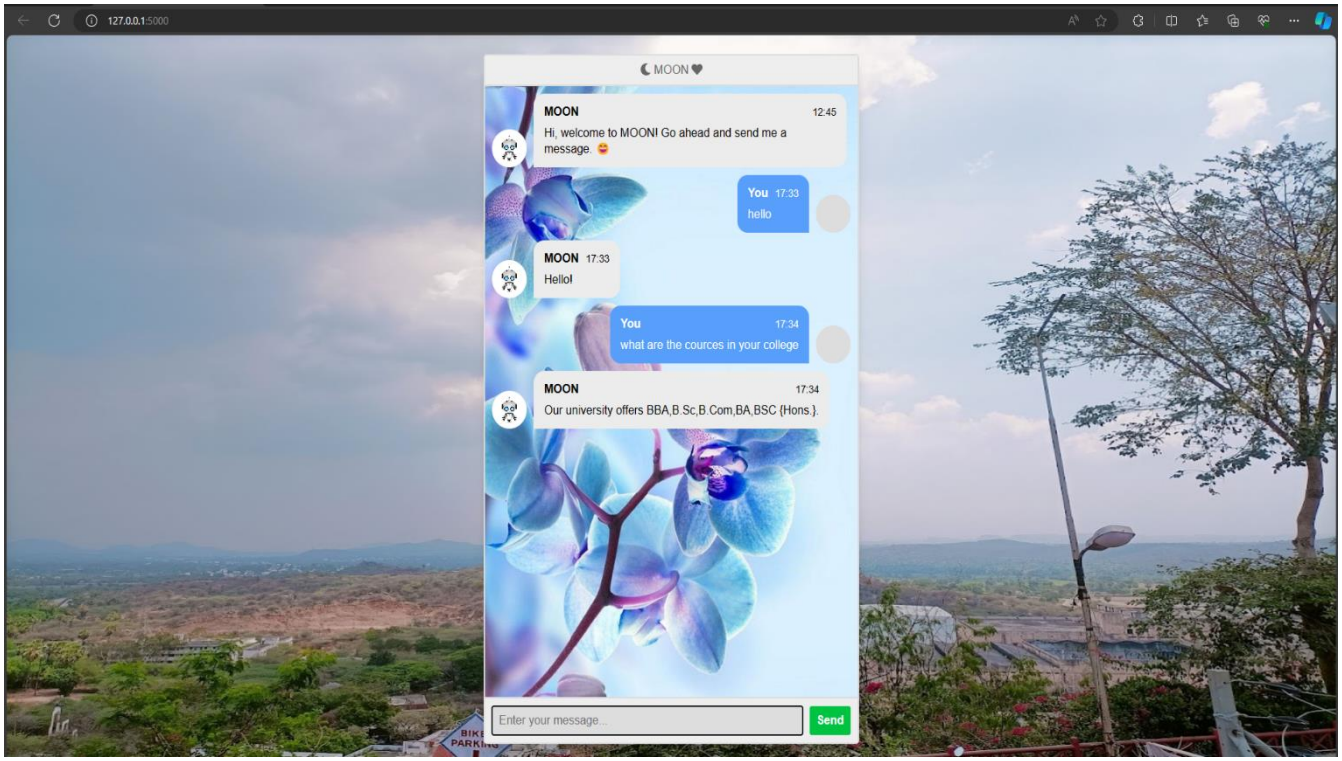
11.3 Final Output



Screenshot.11.1



Screenshot.11.2



Screenshot.11.

References

Websites:

1. https://europepmc.org/An overview of chatbot
2. https://www.tidio.com/blog/chatbot-types/ Types of chatbot
3. Science direct – general architecture of the chatbots
4. https://scholar.ppu.edu/bitstream/handlechatbot learning
5. https://www.frontiersin.org/articles- A systematic literature review of chatbots in education system
6. http://en.wikipedia.org/wiki/Chatterbot - Chatbot information
7. https://www.ibm.com/topics/chatbots- chatbot topic

Books:

1. Chatter Bot : Build a Chatbot With Python by Martin Breuss.
2. Building Chatbots with Python: Using Natural Language Processing and Machine Learning by Sumit Raj.
3. Getting Started with Chatbots: learn and create your own chatbot with deep understanding of Artificial Intelligence and Machine Learning by Akhil Mittal.