Clark University

Clark Digital Commons

School of Professional Studies

Master's Papers

12-2020

A web-based AI assistant Application using Python and JavaScript

Viet Le
Clark University, vile@clarku.edu

Tej Bahadur
Clark University, tsaru@clarku.edu

Jainee Shah
Clark University, jshah@clarku.edu

Roushan Ara
Clark University, rara@clarku.edu

CLARK UNIVERSITY

CHALLENGE CONVENTION.
CHANGE OUR WORLD.

School of Professional Studies

Research Paper

A web-based AI assistant Application using Python and JavaScript

Team member: Viet Le
Tej Bahadur
Jainee Shah
Roushan Ara

Table of Contents

1. Introduction

1.1 General Introduction

Our research is mainly based on a chatbot which is powered by Artificial Intelligence. Nowadays, Artificial Intelligence assistants such as Apple's Siri, Google's Now and Amazon's Alexa are currently fast-growing and widely integrated with many smart devices. These assistants are built with the primary purpose of being personal assistants for every individual user in certain contexts. In this research, we would highlight the development process of the chatbots, features, problems, case studies and limitations.

This research delivers the information, helps developers to build answer bots and integrate chatbots with business accounts. The aim is to assist users and allow transactions between client companies and their customers. As a result, users can accomplish results to queries as well as clients can grow their business.

1.2 Research Problem

In this research, we found the following problems implementing an AI chatbot:

- Sometimes chatbots have the same answer for different queries and answers to some questions are not responded as per user's expectation.
- If anyhow hackers got access to the chatbot, they pose as a client company, to strike up conversations with the in-house personnel of business.
- Chatbot acts like a mechanical robot. It is pre-programmed by developers and can handle queries from humans only if the overall conversation flows in the expected path.
- The system could be slowed down when a large number of users access simultaneously.
- Natural Languages are changing and quite irregular: Current natural language processing algorithms may attempt to catalog the different rules of natural languages, but natural

languages eventually could get replaced by new rules or grammars after a period of time, then become a new language.

1.3 Rationale for Research Project

As new learners, we had interest in working in the machine learning area and we found that AI chatbots are a hot commodity right now constituting a fertile area of research for machine learning. Besides that, in future if we want to start a career as a user experience practitioner, this would be the best way to utilize our research opportunity on some of the latest technologies which are already in the market and have enhanced capabilities. Working on AI chatbot would cover some aspects such as AI chatbot development process, technologies, limitations, possibilities and different types of development methods. To get some deep knowledge about these aspects, we decided to start with the research about AI chatbot.

1.4 Key Terminology

AI chatbot is a system that is able to understand and respond to spoken inputs, to a somewhat broader end-user level. Now chatbot is one of the most accessible ways to get connected without wasting time on roaming over the pages of a complicated website. Some companies have even gone more ahead of this traditional thing and have started using bots as their minor level of customer service. For any chatbot the most important key terminologies are:

1. Utterance: The entire message which a user says.
2. Intent: An intent is the user's intention.
3. Conversational UI: User interface that allows users to interact with a chatbot. Conversational UIs include components such as text area, buttons, links or other graphical elements.
4. Automatic Speech Recognition (ASR) which is a computer technology that is used to identify and process the human voice.

5. Natural Language Understanding (NLU): It deals with machine reading comprehension and behaves like a subtopic of natural-language processing in artificial intelligence.
6. Natural Language Processing (NLP):It deals with interaction between computers and humans using natural language. It is a branch of artificial intelligence. The ultimate objective of NLP is to read, decipher, understand and make sense of the human languages in a manner that is valuable.
7. Text-to-Speech (TTS): It is a type of assistive technology that reads digital text aloud.
8. API: Is a computing interface that defines interactions between various software intermediaries. It defines the kinds of calls or requests that could be made using data formats and conventions that need to be followed.

2. Hypothesis

2.1 Theoretical Foundations

To build this AI chatbot, we have gone through several researches about a basic chat app web-based and machine learning. Then we have found the following theoretical foundations:

Automatic Speech Recognition (ASR): is a technology that converts spoken words into text. With ASR, voice technology can detect spoken sounds and recognize them as words. ASR is the cornerstone of the entire voice experience, allowing computers to finally understand a natural form of communication: speech.

Natural Language Understanding (NLU): the comprehension by computers of the structure and meaning of human language, allowing users to interact with the computer using natural sentences. NLU is artificial intelligence that uses computer software to interpret text and any type of unstructured data. NLU can digest a text, translate it into computer language and produce an output in a language that humans can understand.

Text-to-Speech (TTS): Text-to-speech (TTS) is a type of assistive technology that reads digital text aloud. It’s sometimes called “read aloud” technology. With a click of a button or the touch of a finger, TTS can take words on a computer or other digital device and convert them into audio. TTS is very helpful for kids who struggle with reading. But it can also help kids with writing and editing, and even focusing.

Client-server model: Client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs, which share their resources with clients. A client does not share any of its resources, but it requests content or service from a server. Clients, therefore, initiate communication sessions with servers, which await incoming requests

JavaScript: JavaScript is a scripting or programming language that allows to implement complex features on web pages. JavaScript is involved with displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes in the websites..

Python: is a programming language shining for data science, machine learning, systems automation, web and API development.

Google Assistant API: is an API of an AI assistant built by Google that allows you to manually register or update a device using a JSON file and the REST API. It provides a way to define and register the device model. Google developers provides APIs, software development tools, and technical resources that allow communication with Google Services and their integration to other services. The Google Assistant SDK service exposes a low-level API that lets you directly manipulate the audio bytes of an Assistant request and response for languages like Node.js, Go, C++, Java on all platforms that support gRPC.

2.2 Literature Reviewed, Discussed and Applied

In technical terms, a chatbot is a program of continuous commands which is able to receive an input data, then process and return a corresponding output data through the user interface at a given time. The input and output data are now in voice format, therefore they are appropriately processed under various technologies such as Automatic Speech Recognition (ASR), Natural Language Understanding (NLU) and Text-to-Speech (TTS). Basically, ASR is used to convert the words a user has spoken into text in several types of conversations such as speech-to-text for in-person conversation, captioning for online meetings and telephone conversations. Then, NLU is primarily focused on machine reading comprehension to understand the content of text to issue corresponding commands based on comprehending text. Finally, TTS reads a sentence or a paragraph of text aloud to the users such as reading e-news or stories from e-books. In this paper, we would focus on learning Google Assistant as an AI assistant for chatbot perspectives.

The process of a chatbot is similar to an online chat system by implementing a client server approach to acquire the signal and stream it to a server. The server module first waits for the client to send a connection request. If the connection is granted, a client can mutually communicate with the server.

In a chatbot case, a web browser plays a client role for an input mechanism to acquire a signal. Then this signal is sent to the server to process, analyze and generate responses. Server response generation can be broken down into two categories: data retrieval and information output. The core focus of this paper is to implement JavaScript on the client side to exchange input and output data between client and server, apply python on the server side to process requests from clients, access google assistant API for generating a response that is relevant to the request.

3. Methods

3.1 Study Method

Before implementing any chatbot, we need to know which chatbots are available now. We found the following chatbots which are most popular and used to accelerate the business automating the customer experience, as answer bot, guide bot, chatbot API, advanced agent AI bot, customizing bot solution etc.

After selecting the right chatbots, we faced the issue of the design and development involving a variety of techniques. We understood what the chatbot will offer and what category falls into helps us pick the algorithms or platforms and tools to build it.

The requirements for designing a chatbot included accurate knowledge representation, an answer generation strategy, and a set of predefined neutral answers to reply when user utterance is not understood. The first step in designing the system was to divide it into constituent parts according to a standard so that a modular development approach can be followed. We introduced a general chatbot architecture.

The process starts with a user's request, after the chatbot receives the user request, the language understanding component parses it to infer the user's intention. Once the chatbot reaches the best interpretation it can, it determines how to proceed. When the request is understood, action execution and information retrieval take place. The chatbot performs the requested actions or retrieves the data of interest from its data sources, which may be a database, known as the Knowledge Base of the chatbot, or external resources that are accessed through API calls or can be defined in JavaScript functions.

Upon retrieval, the Response Generation Component used Natural Language Generation (NLG) to prepare a natural language human-like response to the user based on the intent and context information returned from the user message analysis component. The appropriate responses were produced by one of the three models as, rule-based, retrieval based, and generative model . We could develop Chatbots in two ways:

i. Using any programming language like Java, Clojure, Python, C++, PHP, Ruby, and Lisp or using state-of-the-art platforms. We distinguished six leading NLU cloud platforms that we can use to create applications able to understand natural languages: Google's DialogFlow, Facebook's wit.ai , Microsoft LUIS, IBM Watson Conversation, Amazon Lex , and SAP Conversation AI . All these platforms are supported by machine learning. They share some standard functionality (cloud-based, support various programming and natural languages) but differ significantly in other aspects.

ii. Using other known chatbot development platforms such as, RASA, Botsify, Chatfuel, Manychat, Flow XO , Chatterbot , Pandorabots, Botkit, and Botlytics.

After choosing a language, We need to choose the correct development process. There are two development process we could choose for chatbot, reasons are:

1. Building a chatbot with web API and node.js. Web browsers are familiar with the Web Speech API, which allows users to integrate voice data in web apps. At first, we need to install node.js in our machines, then to build the web app, we need to follow the following steps, as:

a. Use the Web Speech API's SpeechRecognition interface to listen to the user's voice.

b. Send the user's message to a commercial natural-language-processing API as a text string.

c. Once the API returns the response text back, uses the SpeechSynthesis interface to give it a synthetic voice.

2. Building a simple chatbot creating index.html file, including JavaScript and CSS . Defining the questions and answers in the backend, it works with limited vocabulary. So the responses can be similar sometimes and the system can get new questions which are beyond its capacity.

3.2 Sample to Be Used in the Study

We have chosen to work on “answer bot” . This chatbot is mainly followed by the Rule-based model which has been built with, like numerous online chatbots. It is based on a fixed predefined set of rules, based on recognizing the lexical form of the input text without creating any new text answers. The knowledge used in this chatbot is hand-coded and is organized and presented with conversational patterns. Creating a comprehensive rule database allows the chatbot to reply to more types of user input. This model is not robust to spelling and grammatical mistakes in user input. It responds to selection for single-turn conversation, which only considers the last input message. Usually, in more human-like chatbots, multi-turn response selection takes into consideration previous parts of the conversation to select a response relevant to the whole conversation context. We started with the English language and for development we chose a python framework called Django. For further functionalities we used JavaScript and for user interface, used HTML and CSS.

3.3 Explanation of Measurements, Definitions, Indexes, etc. and Reliability and Validity of Study Method and Study Design

Measuring certain parameter of the chatbot, we adopted three evaluation metrics:

- Dialog efficiency in terms of matching type.
- Dialog quality metrics based on response type.
- User satisfaction assessment among our group members.

The chatbot required tests in the following four areas: Conversation design testing, entity testing, fulfilment testing and user acceptance testing. The conversation design testing consist of following key areas:

Conversation Flow: a process where two entities exchange communications between themselves through meaningful sentences.

Intent matching and responses: the goals or the purpose of the user's input, or we can call it a "collection of sentences."

Small talk: basically a casual conversation between user and the bot.

Fallback: a process to measure the response of chatbot on unmatched inputs.

Entity testing is when the user speaks or types, chatbot will look for the value of the various entities it needs from the context of the conversation. We use entities to automatically extract the information from what the user says.

The fulfilment testing includes the bot needs to request the information to fulfil the user's request once requirement from the user is received.

The main priority is to test that the chatbot functionality is as per requirements and goals. It is very important that user testing is done on a chatbot before releasing it to the market.

3.4 Description and Justification of Analytical Techniques Applied

In order to measure the quality of each response we classify responses according to an independent group member evaluation of "reasonableness": reasonable reply, non-reasonable, understandable or nonsensical reply. After analysis, it was found that nonsensical responses are more likely than reasonable or understandable but unusable answers. We found the conversation less repetitive and more interesting which measured the satisfaction to be good.

Our chat bot answers questions based on a set of predetermined answers on which it was initially set. Instead of that, we could use Google Assistant API to deliver automated responses to user inputs, then generate many different types of responses. This feature allows developers to build chatbots using python that can converse with humans and deliver appropriate and relevant responses.

3.5 Assumptions and Implied Limitations of Study Method and Design

We found a number of assumptions in the course of our experiment which show that our chatbot would be sufficient to successfully conduct a QA conversation. We developed a process step-by-step to ensure accuracy and efficiency of the chatbot.

- Users would use the system to obtain information, thus most of their utterances would be questions or information requests.
- Users can easily deal with the requests from the system to send their previous utterances, if the system misunderstood them.
- The requests for clarification from the user will be few: in fact, our response format provides more detail than explicitly needed, and this has been shown to be an effective way to reduce the incidence of requests for clarification.

Here are some of the limitations that we found during our research.

i. The most significant limitation of the chatbot is that it doesn't understand human context. It gives the answer based on its training data. It cannot understand the context of humans, and this is a massive gap that can frustrate customers. The AI-powered smart-bots can understand the general context, but 40 out of 100 cases are not related to the general context (Vishal, 2019).

ii. Chatbot is poor in decision-making. It can’t differentiate between the good and the bad.

iii. Chatbots have no emotions, and they cannot relate to any low situation. It can never establish a good connection with the customer, which can be crucial for the growth of the business.

4. Findings

4.1 Brief Overview of Research Project

As we described above, chatbots are automated running programs that can interact and communicate with humans in a common language through texts or voice-based audio conversation. Like a normal chat application, a chatbot includes an app layer, a database and APIs to call functions. To have a better understanding of a chatbot, we will give an overview of a basic chatbot application as below:

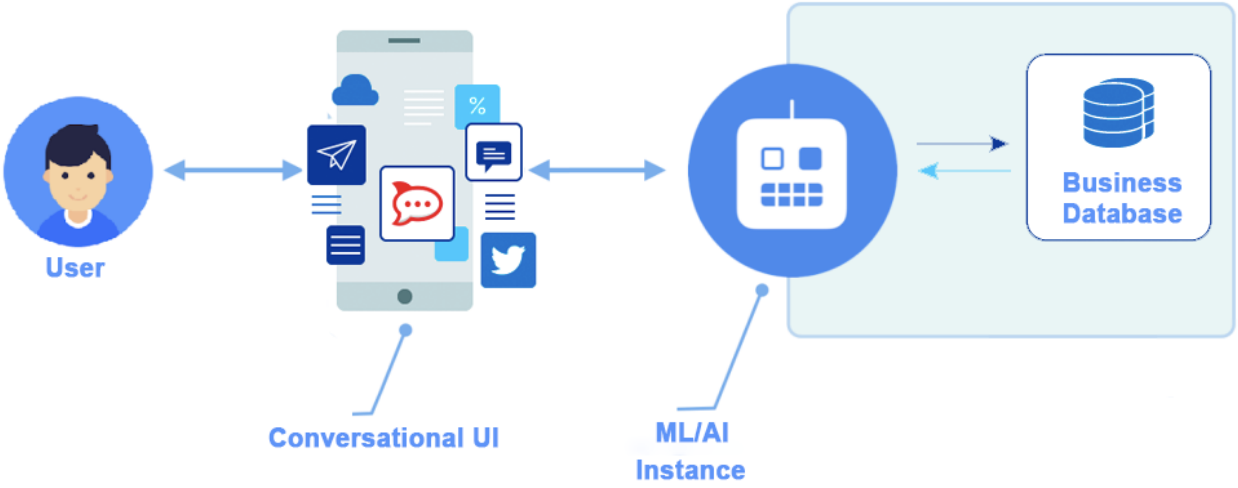


Figure 1. The Overview of an AI chatbot application.

Users can easily send a request on a chatbot application through Conversation UI, then ML/AI Instance is responsible for analyzing and processing the user's request, accessing database or calling APIs methods if necessary, and finally generating proper response to the user.

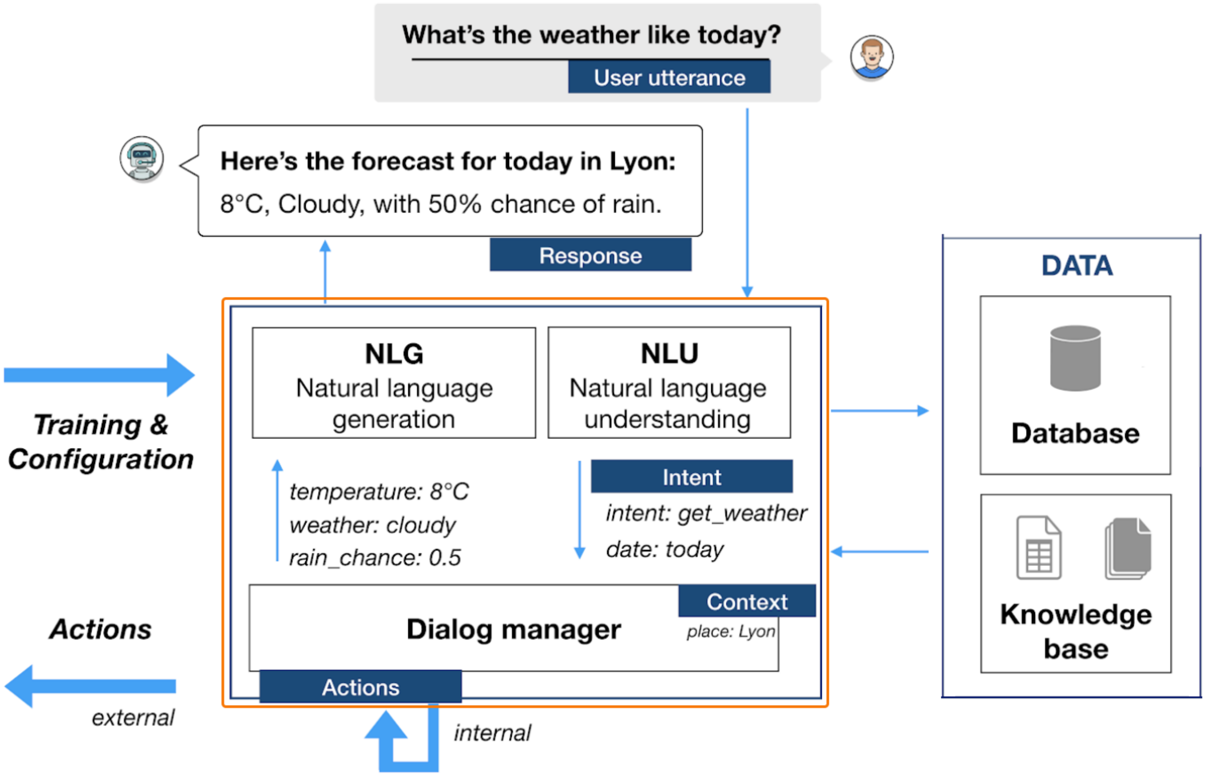


Figure 2. Machine Learning/Artificial Intelligence (ML/AI) Instance.

ML/AI Instance places all processes related to Natural Language Processing (NLP). NLP mainly focuses on examining an utterance and extracting the intent and entities. In order for the NLP unit to know how to map utterances to intents, it is trained and configured with a dataset of examples of utterance-intent mappings. Intents may have parameters, so-called slots (e.g., the date of a weather forecast), and the language understanding part of the NLP must be able to infer their values from utterances (e.g., date: today). Once an intent is identified, the dialog management component enacts an appropriate action, i.e., a specific operation serving the intent (e.g., perform a call to the weather API). To disambiguate similar intents or infer values of slots, additional information, the dialog context, may be used (e.g., if the chatbot already knows the location of the user, it does not need to ask for it in order to provide a localized weather forecast). The dialog control is designed either explicitly by defining conversation flows or derived from previous conversations, or using a combination of both techniques, said Preez (2019).

In this paper, we do not go into details of how to train or configure a model with machine learning. Instead, we will focus on design and research the implementation of a chatbot as a web app using an AI assistant API. Therefore, to solve a problem about chatbots implementation, we treat a chatbot as a web application and divide it into two parts: Front-end UI and Back-end Brain.

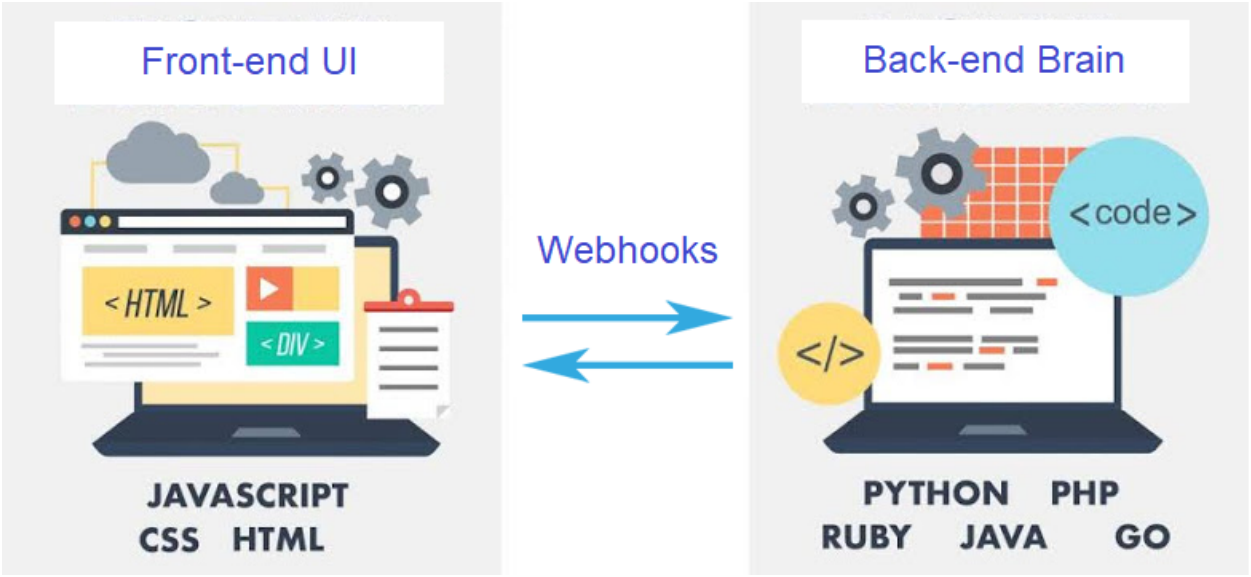


Figure 3. Chatbots web-based application development.

Front-end UI (User Interface): the interface the app where the user interacts with directly, including: text colors and styles, images, graphs and tables, buttons, colors, and navigation menu. HTML, CSS, and JavaScript are the languages used for Front End development. The structure, design, behavior, and content of pages when apps are opened up. Responsiveness and performance are two main objectives of front-end UI.

Backend Brain: is the server side of the app. It defines how data is stored and arranged, and also makes sure everything on the client-side of the app works fine. This part of the app is behind which users cannot see and interact with, but indirectly access through a front-end UI.

Chatbots can be built in basically any programming language that allows you to make a web API, especially Java and Python offer a ton of libraries supporting bot API. The backend receives the user's request, analyses or processes it to generate a corresponding response, and returns it to the user.

The web application server will then have to setup webhooks - URL-based connections between the bot and the chat platform. Webhooks will allow users to securely send and receive messages via simple HTTP requests.

API serves as a communication link between backend and frontend which allows multiple applications to fetch and send data from connected sources. API conveys your message or requests to the source or application and sends the response back to you. It has been known that every industry that is serving customers via digital channels is making use of API to get and send user data to relevant teams. Using dialog Flow developers can build engaging voice and text based conversational interfaces powered by AI.

Dialog management employs natural language processing techniques to extract meaning from a user's messages. For example, it may identify a symptom from a request by extracting named entities and employ the inference engine to capture the relationships between the entities and to invoke a specific rule to infer a symptom.

The inference engine receives requests from the dialog manager and interacts with the knowledge base to infer an appropriate intent based on a user's request. In addition, it may infer entities from the user's request to map them to appropriate actions.

The purpose of the inference engine is to identify a closest possible intent for a given request. In several cases, the initial corpus required to train a classifier could be small. Thus, the intent inference could have either low recall or low precision.

The knowledge base represents the chatbot knowledge, including inputs received from a user and about available services and how they relate to the domain. The knowledge base includes a knowledge graph constituting a set of entities and their relationships, and a set of rules that make inferences based on entities and their relationships. The chatbot designers need to populate the knowledge base with expert knowledge

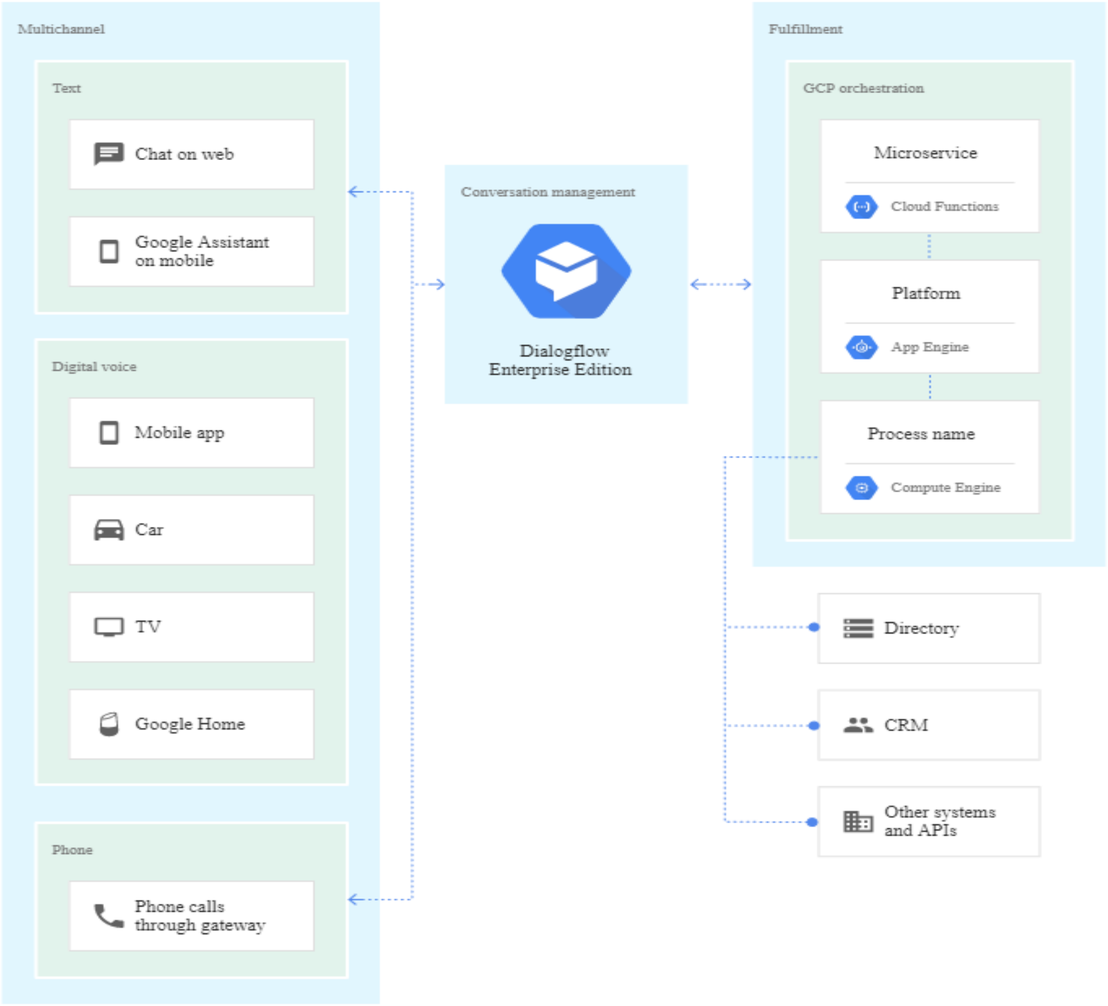


Figure 4. A typical conversational bot system architecture, Google (2020).

The most important component of an AI chatbot is ML/AI Instant. Even though we do not build a ML/AI model, we give some background information about a specific AI model being available with API, Google Assistant API. An API (Application Programming Interface) is a computing interface that defines interactions between various software intermediaries. It defines the kinds of calls or requests that could be made using data formats and conventions that need to be followed. The Google Assistant API allows you to manually register or update a device using a JSON file and the REST API. It provides a way to define and register the device model which includes get a device model, list the device model, update a device model and delete a device model. Google developers provides APIs, software development tools, and technical resources that allow communication with Google Services and their integration to other services. The Google Assistant SDK service exposes a low-level API that lets you directly manipulate the audio bytes of an Assistant request and response. Bindings for this API can be generated for languages like Node.js, Go, C++, Java for all platforms that support gRPC.

The main reason why we choose Google Assistant is its advanced features. Users can use it to open apps, send messages, make calls, play a specific song on Tidal, check the weather, and many other things without touching the screen of their phones. Of course, Assistant isn't available just on phones. It is now easily found on smart speakers, smartwatches, headphones, and a plethora. Some of the advantages are highlighted such as more action oriented (helps users buy products, book appointments, find stores, and more), voice commands (digital assistants are changing the way consumers interact with businesses), and anytime access (assistant helps people get things done all day—at home, on the go, and in the car).

JavaScript: Front-end Development

It is comparatively time saving to build a JavaScript based Chatbot using web speech API. The Web Speech API enables to incorporate voice functionality easily into web applications. Combined with basic existing web technologies (HTML, CSS), developers are able to design and develop the user interface of any web app. The HTML file defines how the app looks.

Google Assistant API:

Google Assistant helps millions of people around the world get things done more easily every day. Assistant works with content and services to help people find just what they are looking for. Assistant finds the best response for the user's request by asking for more information to clarify the user's intent, show user follow-up suggestions (on devices with screens), or let the user know about related things they can ask. It helps in enabling discovery by offering businesses and developers new ways to reach more users.

Python: Back-end Development

To develop chatbot, Python is the most loved language of developers. The main reason is the presence of Artificial Intelligence Markup Language (AIML) that makes it less demanding for engineers to compose syntax. The effortlessness removes the coding difficulties and diminishes the likelihood of errors.

As we mentioned above, API is a standard that facilitates intercommunication between two or more computer programs, the aim is to help developers easily access predefined functions from other applications. Developers can use the standard request module to communicate with an API, specifically a python web API, then get a response in a JSON format (JSON is a way to store data in an organized, logical manner). The JSON module makes the data simpler to work with and it is also a built-in python module.

For deeper understanding, we could discuss a few points of using an API with python follows these steps:

- First, get an API key (a unique string of letters and numbers).
- Add an API key to each request so that the API can identify you.
- Once we get an API key by using API endpoints we can check if everything is working as expected.

- After we checked the endpoints and everything works as we expected, we can start creating the application, including calls to the necessary API.

From our findings, we could conclude a few points as API acts as a layer between your application and external service. You do not need to know the internal structure and features of the service, you just send a certain simple command and receive data in a predetermined format. It uses HTTP requests for communication with web services. API comply with certain constraints which are as follows:

1. Client-server architecture – the client is responsible for the user interface, and the server is responsible for the backend and data storage. Client and server are independent and each of them can be replaced.
2. Stateless – no data from the client is stored on the server side. The session state is stored on the client side.
3. Cacheable – clients can cache server responses to improve performance.

We determined some types of HTTP request methods and characterize some types of action:

- GET: retrieve information (like search results). This is the most common type of request. Using it, we can get the data we are interested in from those that the API is ready to share.
- POST: adds new data to the server. Using this type of request, you can, for example, add a new item to your inventory.
- PUT: changes existing information. For example, using this type of request, it would be possible to change the color or value of an existing product.
- DELETE: deletes existing information

After embedding Google Assistant with our web application, we could interpret the following points: - the Service name embeddedassistant.googleapis.com is needed to create RPC client stubs. Some methods that are developed within API:

CreateDevice: Register one device, return the device if succeed or error if the device already exists or failed.

CreateDeviceModel: Create a 3p device model.

DeleteDevice: Deletes a specific device user owns.

DeleteDeviceModel: Delete a 3p device model.

Get Device: Get setting info of a specific device user owns, return the device if succeeded or if failed.

It exposes a low-level API that lets the user directly manipulate the audio bytes of an assistant request and response. Bindings for this API can be generated for languages like Node.js, Go, C++, Java for all platforms that support gRPC. Some features of gRPC are it is a modern open source high performance RPC framework that can run in any environment. It can efficiently connect services in and across data centers with pluggable support for load balancing, tracing, health checking and authentication. In addition, we need to understand that we are not integrating commercial devices with Google Assistant SDK. The Google Assistant SDK lets you add voice control, natural language understanding and Google's smarts to your ideas.

We could say JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. The structure and style to web pages is provided by HTML and CSS languages. JavaScript gives web pages interactive elements that engage a user. Incorporating JavaScript improves the user experience of the web page by converting it from a static page into an interactive one. To recap, JavaScript adds behavior to web pages. Developers can also use JavaScript to build simple web servers and develop the back-end infrastructure using Node.js. JavaScript can be used as a client-side programming language and helps web developers to make web pages dynamic and interactive by implementing custom client-side scripts. At the same time, the developers can also use cross-platform runtime engines like Node.js to write server-side code in JavaScript. It helps to make the web pages interactive, respond to users instantaneously and create richer user interfaces, without less server interaction and reduces server load. It helps to simplify complex web development applications as developers can use libraries to create shadow DOM boundaries. The shadow DOM further makes web browsers deliver documents with widely used HTML tags like div, select and input. The four components of MEAN stack are MongoDB, Express.js, Angular.js and Node.js. They enable developers to write both the front-end and back-end of a website in JavaScript. However, components of MEAN stack are still interoperable. MongoDB is a modern and schema less NoSQL database, whereas Angular.js is a popular JavaScript framework supported by Google. At the same time, Node.js is a cross-platform and server-side runtime environment, whereas Express.js is designed with features to simplify development of single-page and multiple-pages websites. The fully-featured JavaScript libraries like AngularJS and Ember enable JavaScript web developers to add functionality to complex web applications without writing additional code. At the same time, lightweight JavaScript libraries like React.js make it easier for programmers to accomplish specific tasks. We could say the modern web developers must use JavaScript to make their web applications deliver optimal user experience across various devices, operating systems, and browsers. However, they must be familiar with various JavaScript libraries, frameworks, and tools. They can even combine multiple libraries and frameworks to enhance and extend JavaScript according to requirements of large projects.

4.2 Results of the Method of Study and Any Unplanned or Unexpected Situations that Occurred

To create our AI chatbot we found multiple methods/procedures. As for customizing Google API, developing new API. Developing HTML, CSS, JavaScript and integrating with Django framework (to explain Google API and its functions). Furthermore, Our implementation was using HTML, CSS, JavaScript and integrating with the Django framework (to explain how above tools were used for our system).

While developing our AI chatbot initially some unexpected results we faced where it was not working properly on different web browsers, not responding to user queries as the user expected the answer, security could be implemented and web application needs to be integrated with different cultural languages so it could have high demand in industry.

Some ways to build conversational AI are as follows: Rule-based, Retrieval-based, generative-methods, ensemble methods, grounded learnings, Interactive learning. A short experiment was conducted to demonstrate working of anthropomorphic AI based chatbot. The hypotheses were tested by means of a randomized online experiment in the context of a customer service chatbot for online banking that provides customers with answers to frequently asked questions. Instead of rule-based systems in experiments and practice we used IBM Watson Assistant cloud service with the required AI-based functional capabilities for natural language processing. As such, participants could freely enter their information in the chat interface, while the AI in the IBM cloud processed, understood and answered the user input in the same natural manner. At the same time, these functionalities represent the necessary narrow AI that is especially important in customer self-service contexts to raise customer satisfaction. By means of the IBM Watson Assistant we extracted intentions and emotions from natural language in user statements. Once the user input has been processed, an answer option is automatically chosen and displayed to the user.

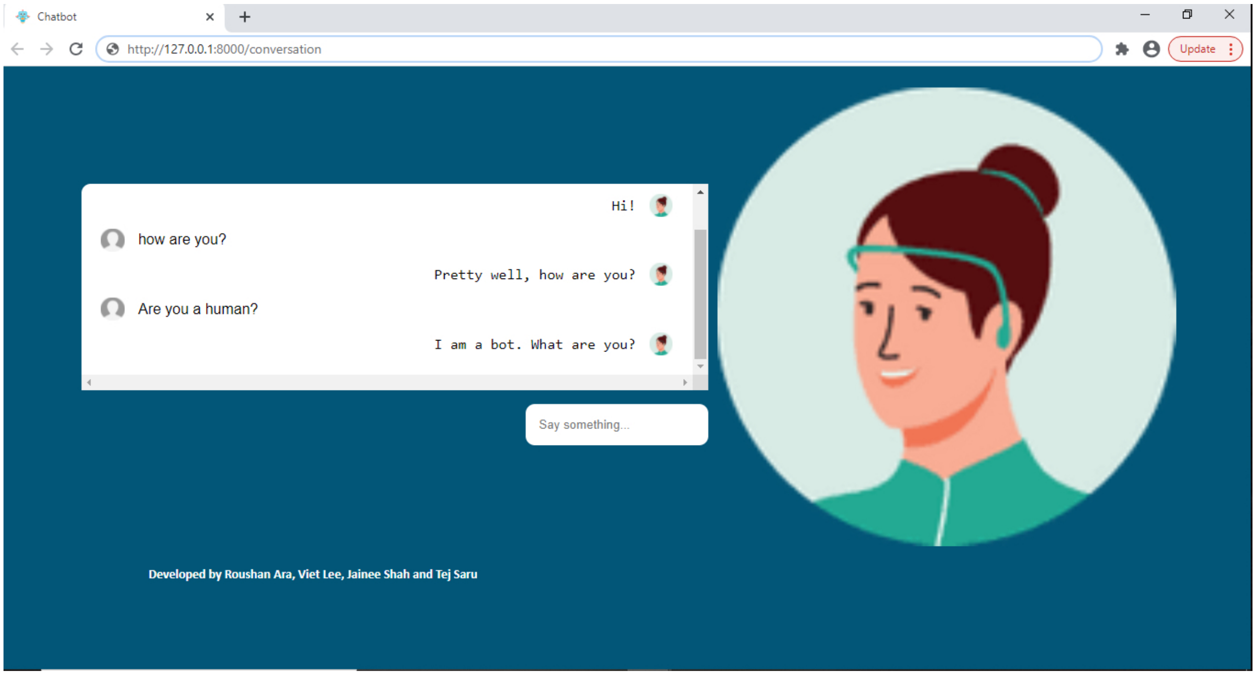
AI is always working and constantly learning, which then requires somebody to be available to supervise it. You can't just let your AI run wild and expect it to handle everything on its own; you'll need somebody who can guide and direct it, and step in if something goes wrong. That means having a system in place where specific employees will be contacted and expected to help out if the AI requires help all hours of the day, which is called human-in-the-loop.

If you don't have a supervisor keeping an eye on your AI, you run the risk of it going rogue. There won't be a robot uprising, but it could result in unhappy customers or lost leads, depending on what your chatbot is doing. to the "always-working chatbot." If the power goes out where the chatbot is housed — whether it's in a server in the office or across the globe — your chatbot goes with it. Similarly, Internet outages still happen occasionally, and natural disasters can cause connection issues affecting an entire country.

4.3 Explanation of the Hypothesis and Precise and Exact Data

Some hypotheses that are proposed are as follows:- Users are more likely to comply with a chatbot's request for service feedback when it exhibits more verbal anthropomorphic design cues. The social reaction to computers depends on the kind and number of ADCs. Usually, the more cues a CA displays, the more socially present the CA(customer agent)will appear to users and the more users will apply and transfer knowledge and behavior that they have learned from their human-human-interactions to the HCI(Human computer interaction).

Social presence will mediate the effect of verbal anthropomorphic design cues on user compliance. The study of how an anthropomorphic recommendation agent had a direct influence on social presence, which in turn increased trusting beliefs and ultimately the intention to use the recommendation agent. Thus, we argue that a chatbot with ADCs will increase consumers' perceptions of social presence, which in turn makes consumers more likely to comply with a request expressed by a chatbot.



Hi!

how are you?

Pretty well, how are you?

Are you a human?

I am a bot. What are you?

Developed by Roushan Ara, Viet Lee, Jainee Shah and Tej Saru

Figure 5. An example of implementing an AI chatbot.

Following the mentioned steps we have developed a demo chatbot above. In this chatbot, users can ask questions by text and can get responses in both voice and text formats together. Users can view the chat history using the slide bar. The chatbot is able to deliver the user question answers successfully.

5. Discussion

5.1 Brief Overview of Material

Designing the chatbot conversation follows the following steps, as:

1. Being specific, If needed. If there is a need to ask a user a question, it should be very specific. But when in doubt, there should not be worries to show that the system is not sure of what the user's answer is. This is the opportunity to jump from a Natural Language format to specific choices. It helps the user get back on track with their experience.
2. Presenting the user with specific options. Avoiding open ended questions, which cause frustration. Being binary in the decisions if needed to be asked the users to face. The art is ensuring that it is given the user the right, personalized options.
3. Not mixing information into one sentence and options.
4. Let the user know what is going on so there is no cognitive dissonance around the expectation of what they are getting from the app
5. Keeping the narrative simple: If there is a need to switch topics, making sure going onto links in nicely.

5.2 Results and Implications

In order to evaluate the implementation of a chatbot, a list of metrics were applied to measure its performances. There are tools and methods which were used to track the overall performance of a chatbot. The first metric is "**comprehension capabilities**", this means a good chatbot should ensure comprehension capabilities of a good texting and error free experience for the user. When a user types a spelling mistake or makes an error in a sentence, the chatbot should enable the 'auto-correct' feature. To achieve this metric, a chatbot firstly should have 'text-based understanding' to quickly understand the questions or a command from the user. For example, if the user texts “I would like to order a pizza”, the chatbot can be aware of the difference between a question and an order. Secondly, chatbot should be capable of ‘balanced text-use’. This allows a chatbot to be able to use a combination of both short descriptions and engaging content to hold the user’s attention. The second metric is “**user engagement**”, so the chatbot can initiate conversation and interact with the users to share and deliver meaningful messages, take direct orders from users, and navigate to layouts and more. The outstanding function of a chatbot is frequently asked questions (FAQs) of users to increase productivity. Of course, “**response speed**” is an indispensable metric as the chatbot purpose is to serve the users instantly. Quality chatbots can deliver responses immediately for effective interactions. Next, chatbots must include basic functionalities with a variety of well-designed functionalities such as onboarding, rich media use, and navigation that lead to a great conversational flow. For instance, chatbots should welcome users with a series of onboarding steps, use engaging rich media images with text to get the user’s attention, and provide navigation tools to help the users with the layout.

Just to focus on the implications of chatbot usage, let's have a glance at the chatbot future. It has been predicted by business insiders that by 2020, 80% of enterprises will use chatbots. In addition by 2022 banks can automate up to 90% of their customer interaction using chatbots and 40% of large companies employing more than 500 people plan to implement one or more intelligent assistant or AI-based chat robot. According to drift, 27% of adult clients in the United States are ready to purchase basic goods through a chatbot, 13% of adults in the US have at least once bought expensive items using chatbots. By 2021 researchers say 4.5 billion dollars will be invested in chatbots. The following graph shows chatbot usage as per company size:

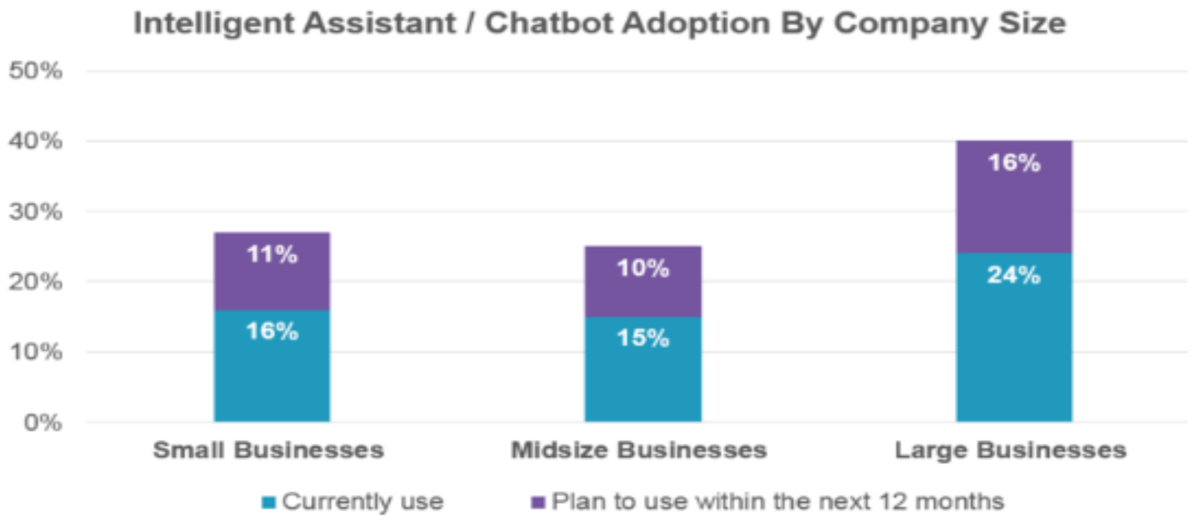


Figure 6. A bar graph showing usage of intelligent assistant as per company size.

5.3 Hypothesis and Findings

We have presented the approaches to develop a chatbot which connected with multiple technologies as API, web platform, scripting language etc. Different development methods follow different strategies to response user's answers. According to business purpose development companies choose which method is more effective and cost minimizing. Whatever method we choose, the purpose is the same which is responding to more accurate answers to consumers. All methods follow the same steps, but they choose different platforms for fetching data such as choosing Google API , defining hard code embed answers in JavaScript function, getting data from database or updated technologies using machine learning language. All methods follow the below steps:

1. Libraries & Data
2. Initializing Chatbot Training
3. Building the Deep Learning Model
4. Building Chatbot GUI
5. Running Chatbot

6. Conclusion

In this paper, we researched about a web based AI chatbot which is based on Google Assistant API, JavaScript, Python, CSS and HTML. We created text based input and output web interface and code pattern uses Google API Dialog Flow to control the conversation flow between user and chatbot. Our recommendation to disclose to customers is that they are interacting with a non-human interlocutor. By showing that CAs can be the source of persuasive messages, we provide evidence that attempting to fool customers into believing they are interacting with a human might not be necessary nor desirable. The focus should be on employing strategies to achieve greater human likeness through anthropomorphism by indicating, for instance, identity, small-talk, and empathy, which we have shown to have a positive effect on user compliance. When employing CAs, and chatbots in particular, providers should design dialogs as carefully as they design the user interface. Besides focusing on dialogs that are as close as possible to human-human communications, providers can employ and test a variety of other strategies and techniques.

AI-based CAs have become increasingly popular in various settings and potentially offer a number of time- and cost-saving opportunities. However, many users still experience unsatisfactory encounters with chatbots (e.g., high failure rates), which might result in skepticism and resistance against the technology, potentially inhibiting that users comply with recommendations and requests made by the chatbot. In our study we designed a web-based AI application to show how Machine Learning, Python & JavaScript techniques increase user compliance with a chatbot's request for service feedback. Our study is thus an initial step towards better understanding how AI-based CAs may improve user compliance by leveraging the effects of Machine learning and AI as the need to stay consistent in the context of electronic markets and customer service. Consequently, this piece of research extends prior knowledge of

CAs as anthropomorphic information agents in customer-service. We hope that our study provides impetus for future research on compliance mechanisms in CAs and improving AI-based abilities in and beyond electronic markets and customer self-service contexts.

The paper provides several avenues for future research focusing on the design of chatbot as information agents and may help in improving the interaction of AI-based CAs through user compliance and feedback. Moreover, with the rise of AI and other technological advances, intelligent CAs will become even more important in the future and will further influence user experiences in, for example, decision-makings, onboarding journeys, and technology adoptions. Since the study was conducted in an experimental setting with a simplified version of an instant messaging application, future research needs to confirm and refine the results in a more realistic setting, such as in a field study. In future research we need to examine how to influence users to start the chatbot interaction but who just simply end the questionnaire after their inquiry has been solved, a common user behavior in service contexts that does not even allow for the emergence of survey requests. Furthermore, our sample consisted of only German participants, so that future researchers may want to test the investigated effects in other cultural contexts. Users might get used to the presented cues and will respond differently over time, once they are acquainted to the new technology and the influences attached to it.

7. References

7 Steps for Migrating Sensitive Data to the Cloud - A Guide for Data Teams (n.d). Retrieved from https://www.kdnuggets.com/2020/05/build-deploy-machine-learning-web-app.html

7 Characteristics of a Great Chatbot (n.d). Retrieved from https://www.bluewolf.com/bluewolf-now/7-characteristics-great-chatbot

Adam M., Wessel M., Benlian A. (March 2020). Retrieved from https://link.springer.com/article/10.1007/s12525-020-00414-7

Azvine B., Djian D., Tsui Ching K., Wobcke W (n.d). Retrieved from http://www.cse.unsw.edu.au/~wobcke/papers/ia.pdf

Bisen V. (2019 July). Retrieved from https://medium.com/vsinghbisen/what-ai-techniques-are-used-in-chatbots-explained-with-examples-7956ab8a74ec

Chatbot Learning: Everything you need to know about machine learning chatbots (2020). Retrieved from https://www.whoson.com/chatbots-ai/chatbot-learning-everything-need-know-machine-learning-chatbots/

Fourault S. (2017 August). Retrieved from https://chatbotsmagazine.com/the-ultimate-guide-to-designing-a-chatbot-tech-stack-333eceb431da

Fakhruddin H. (June 2017). The Other Side Of Chatbots: Risks & Disadvantages. Retrieved from https://teks.co.in/site/blog/the-other-side-of-chatbots-risks-disadvantages/

How to use an API with Python(Beginner's Guide). Retrieved from https://rapidapi.com/blog/how-to-use-an-api-with-python/

Introduction To Machine Learning using Python (January 2019). Retrieved from https://www.geeksforgeeks.org/introduction-machine-learning-using-python/

Jain M., Kumar P., Ramachandra K., Patel S. (June 2018). Evaluating and Informing the Design of Chatbots. Retrieved from https://ubicomplab.cs.washington.edu/pdfs/chatbot_study.pdf

Jain M.,Kumar P., Kota R., Patel S. (June 2018). Retrieved from https://ubicomplab.cs.washington.edu/pdfs/chatbot_study.pdf

Killer (January 2020). Retrieved from https://killernewtech.blogspot.com/2020/01/front-end-and-back-end-development.html

Ilias M.(May, 6 2020). Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7256567/

M.Mirthula (n.d). Retrieved from https://towardsdatascience.com/how-to-build-your-own-ai-personal-assistant-using-python-f57247b4494b

Markov D.(2016,November). Retrieved from https://tutorialzine.com/2016/11/introduction-to-chatbots

Mindfire Solutions (June 2017). Retrieved from https://medium.com/@mindfiresolutions.usa/how-important-is-javascript-for-modern-web-developers-2854309b9f52

Mostaco G., Campos L., Souza R., Cugnasca C. (June 2018). Retrieved from https://www.researchgate.net/publication/327212062_AgronomoBot_a_smart_answering_Chatbot_applied_to_agricultural_sensor_networks

Milhorat P., Schlogl S., Chollet G. (March 2014). Retrieved from https://www.researchgate.net/publication/264516372_Building_the_Next_Generation_of_Personal_Digital_Assistants

Nasirian F., Ahmadian M., Lee D. (May 2017). Retrieved from https://www.researchgate.net/publication/322665841_AI-Based_Voice_Assistant_Systems_Evaluating_from_the_Interaction_and_Trust_Perspectives

Preez S., Lall M., Sinha S. (June 2009). Retrieved from https://www.researchgate.net/publication/224564336_An_intelligent_web-based_voice_chat_bot

Sacred Heart University Library (2020). Retrieved from https://library.sacredheart.edu/c.php?g=29803&p=185916

Schar A., Hermannstrasse., Winterthur. (May 2018). Retrieved from https://digitalcollection.zhaw.ch/bitstream/11475/12450/1/final_Bachelor%20Thesis%20Chatbots%20A.%20Sch%C3%A4r.pdf

Stafford G. Retrieved from https://programmaticponderings.com/2018/09/24/integrating-search-capabilities-with-actions-for-google-assistant-using-gke-and-elasticsearch-part-2/

Stavan H.(December,2017). Retrieved from https://uxdesign.cc/case-study-designing-a-chatbot-for-a-design-studio-part-1-fff81deeadd

Telang P., Kalia A., Vukovie M., Pandita R., Singh M. Retrieved from https://www.csc2.ncsu.edu/faculty/mpsingh/papers/columns/IC-22-06-Chatbots-NWI-18.pdf

Vishal V. (May 2019). Retrieved from https://www.smatbot.com/blog/what-are-the-limitations-of-chatbots/

Viswanathan V. (June,2018). Retrieved from https://vigilviswanathan.medium.com/what-is-a-chatbot-and-how-can-you-use-it-for-business-6461753e08f9

Vyas B. (November 2017). Retrieved from https://chatbotslife.com/6-key-metrics-to-measure-the-performance-of-your-chatbot-5fd0adfd0b5b

Xavier P. (February 2019). Retrieved from https://medium.com/quick-code/absolute-beginners-guide-to-slaying-apis-using-python-7b380dc82236

Xu A., Liu Z., Akkiraju R. (May 2017). Retrieved from http://edithlaw.ca/teaching/cs889/2018/reading/Asking/Paper9.pdf

Brain[Brn.AI] Code for Equity (April 2019). Retrieved from https://chatbotsmagazine.com/chatbot-report-2019-global-trends-and-analysis-a487afec05b