

MEERA

Multifunctional Educational Expert in Real-time Assistance

Project report submitted in fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

Computer Science and Engineering / Information Technology

By

Lavish Kumar Jangir (161022)

Prashasy Ashok (161325)

UNDER THE GUIDANCE OF

Dr. Ravindara Bhatt



Department of Computer Science & Engineering and Information Technology

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,

WAKNAGHAT

SOLAN, HP (173234)

DECLARATION

We hereby declare that the work reported in the B. Tech Project Report entitled **MEERA (Multifunctional Educational Expert in Real-Time Assistance)** submitted at **Jaypee University of Information Technology, Waknaghat, India** is an authentic record of our work carried out under the supervision of **Dr. Ravindara Bhatt**. We have not submitted this work elsewhere for any other degree or diploma.



Lavish Kumar Jangir
161022



Prashasy Ashok
161325

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.



Dr. Ravindara Bhatt
Assistant Professor (Senior Grade)
Department of Computer Science & Engineering
JUIT
Date:

ACKNOWLEDGEMENT

We owe our profound gratitude to our project supervisor **Dr. Ravindara Bhatt**, who took keen interest and guided us all along in my project work titled — **MEERA(Multifunctional Educational Expert in Real-Time Assistance)**, till our end semester presentation of our project by providing all the necessary information for developing the project. The project development helped us in research and we got to know a lot of new things in our domain. We are really thankful to him, and we hope the next half of the project also goes as well as this one so that we could complete our project under his superb guidance.

We would also like to show our gratitude towards our project lab supervisor **Mr. Ravi Raina** for his excellent and constant support to provide us with all the necessary equipment to complete this project till now.

Table of Contents

DECLARATION	II
ACKNOWLEDGEMENT	III
TABLE OF FIGURES	VI
ABSTRACT	VII
CHAPTER 1 - INTRODUCTION	1
1.1 Aims & Objectives	2
CHAPTER 2 - LITERATURE SURVEY	4
2.1 Chatbots in Industry	4
2.2 History of Chatbots	6
2.3 Natural Language Understanding Engine & NLP	7
2.4 Artificial Intelligence	9
2.5 Artificial Intelligence Methodologies	10
2.6 Chatbot Architecture	11
CHAPTER 3 – SYSTEM DEVELOPMENT	14
3.1 Project Problem Statement	14
3.2 Proposed Solution	15
3.3 Software Engineering Models	16
3.4 Functional and Non-Functional Requirements	18
3.5 Functional and Non-Functional Requirements	18
3.6 Software and Hardware Specification	20
3.7 Project Planning	20
3.8 Project Architecture	22
3.9 Conversational User Interface Design	28
3.10 Dialog Design	29
3.11 Project Integration And Testing	31
3.12 Deployment Diagram	31

3.13 Implementing BotCopy Chatbot UI	36
3.14 Implementing Web Scraper	38
3.15 Implementing RESTful API	40
3.16 Testing	43
3.17 Git Version Control	44
CHAPTER 4– PERFORMANCE ANALYSIS	45
4.1 Performance Analysis	45
CHAPTER 5– CONCLUSION	48
5.1 Project Outcome	48
5.2 Future Scope	50
5.3 Conclusion	50
REFERENCES	51

Table of Figures

Figure 1 : Representation of chatbot chatting style conversation	1
Figure 2 : Chatbot conversation example	3
Figure 3 : Botsify example	5
Figure 4 : Advanced Chatbot Architecture	13
Figure 5 : Simplified chatbot architecture	13
Figure 6 : Agile model of software development	16
Figure 7 : Agile model of software development	17
Figure 8 : Cahtbot Detailed Architecture	22
Figure 9 : Data flow diagram for fulfillment	24
Figure 10 : Use - Case Diagram	25
Figure 11 : Project Structure and technologies used	25
Figure 12 : Activity diagram for user interactions	27
Figure 13 : Our first prototype	30
Figure 14 : Dialogflow data handling	33
Figure 15 : Botcopy Implementation	38
Figure 16 : Web Scraper Architecture	39
Figure 17 : Web Scraper Implementation	40
Figure 18 : RESTful Web Service Architecture	40
Figure 19 : API Request Handler	41
Figure 20 : API Service Handler	42
Figure 21 : API Service Worker	43
Figure 22 : Git Version Controlling	44
Figure 23 : Firebase analytics report	45
Figure 24 : BotCopy Interaction Analytics Summary	46
Figure 25 : Google clod function Logs	46
Figure 26 : Bot Copy Performance Audit Summary by Lighthouse	47
Figure 27 : Dialogflow consol	48
Figure 28 : Bot implementaion of university website	49
Figure 29 : Bot implementaion of university website	49

ABSTRACT

For the basic day to day interaction nature defines communication to be the default protocol. These days with every mundane routine task being affiliated to digital media like smartphones and computers, people interact with lifeless static image pixels as a mode of communication. This type of query retrieval is outdated and these days we require a new communication model. This opens up the idea of chatbots as an information delivery medium, people enjoy text messaging a lot these days and instead of searching/scavenging websites and applications they tend to love the way a chatbot delivers the required and most appropriate information directly.

Almost every service evolves with better feedback, similarly feedback enabled communication media like chatbots and voice assisted interactions enable for better content delivery and human interactions. The historic norm of engaging with a computer services solely confined to plastic keyboards and lifeless screens are rapidly being outdated. This new way of communication in forms of voice inputs and text messaging are highly reliable and are extensively backed by technologies like Machine Learning and Artificial Intelligence.

This project is aimed to cater to a particular sector for its operations, the market is nowadays flooded with chatbot helping people guide applications and information on various sectors like the IT industry but the education sector is yet to find a good way to implement chatbots. In this project we try to implement chatbots for different educational purposes, also managing potential Student interactions is a major field that this project will address and will help the educational institutions.

A chatbot allow a client to explore an Institute's website without actually navigating through the website. The bot will be capable of delivering appropriate responses. The users will be able to retrieve major information like the history of the institute, the missions of the various departments and various other queries.

The further expansion of this project will address to the hassles of manually structuring conversational flow by eliminating the manual feeding of data. Implementation of Web Crawlers will be used to scrape data from various pages and link from a given domain and will automatically be converted to an appropriate data format required by the training engine of the chatbot to facilitate all different interactions.

Chapter 1 - INTRODUCTION

The new-age millennials are readily encompassing their routine lives with new age technologies and humanoid interactions, this is quite evident in modern times where all tech giants are empowering not only their services but also their existing tech architecture with the artificial intelligence, the most prevalent example are the infamous google assistant, google home, apple home, and everyone’s favorite Alexa. The new age technically sound generation is accepted to be critical and game-changing customers for the business.

The new age demands, on fingertips answers, effortless experience, and increasingly shrewd options, the banking sector is the first ones to adopt such technology. This mix had developed exponentially, assisting banking sector to reach more masses over the world globally.

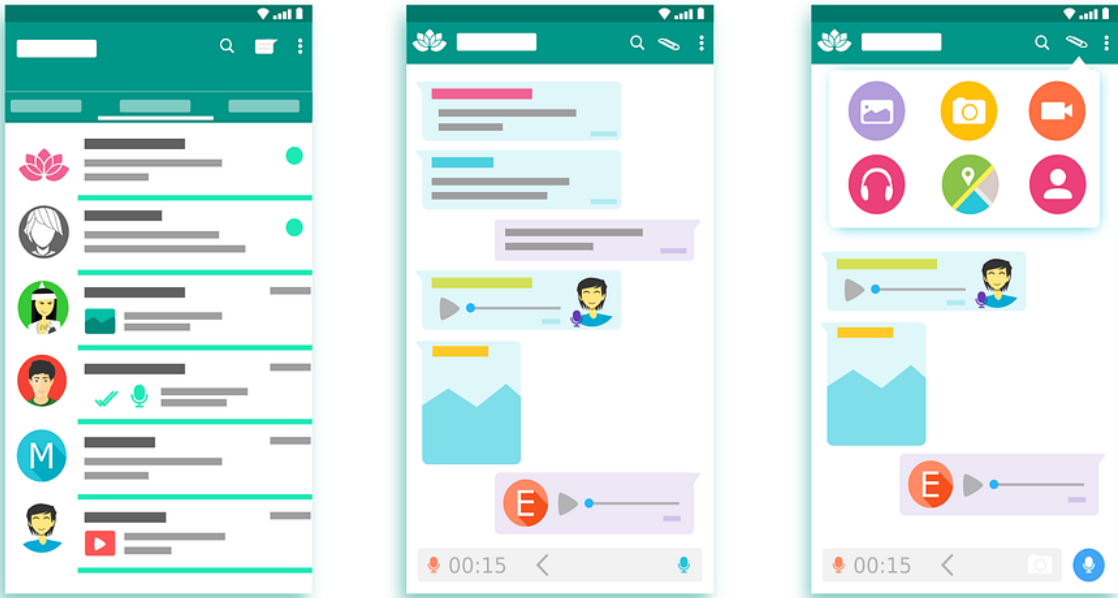


Figure 1 : Representation of chatbot chatting style conversation

The education sector while facilitating this ailment for other industries is yet to implement conversationally engaging helper agents for students, the education sector has integrated a lot of other technologies and has gained a lot of profits from them and are now very keen to implement field agents to attract capable students to right institutes for them. A complex software like chatbot can respond to a variety of customer prompts ranging from basic greetings to complex mood-based enlightening conversations.

We want to use bleeding edge tech like NLP (Natural language processing), Cognitive Learning, Machine learning, Artificial intelligence and modern Web dev technologies to cater to the educational institutes to increase student interactions with the institute's information website/portfolio. The primary objective of our project to is to integrate many new-gen technologies to cater to the new age generations by satisfying their information needs quicker and in a better understandable way.

For example imagine you are browsing for a university that is ranked within top 100 and you scraped their whole website but could not find it, you can ask the chatbot to tell you the rank and give a link to the certificates original URL, this would seem so handy and finally satisfies our aim to increase customer interactions, make the information delivery more efficient and reduce the hassles.

1.1 Aims & Objectives

Our project is aiming to provide a quick and advantageous way to arrange Student – Institute Interactions. This online chatbot will help assist with content delivery more elegantly and efficiently and also to facilitate various user queries.

The application will allow a user to:

- Interact with the bot and retrieve information.
- Interact and engage in conversations
- Interact and review the university details.
- Get references and meaningful insights about the institute.

The application can be used to:

- Gain insights into the potential students
- To track and analyze demographics
- And make interactions and content delivery more efficient.

The chatbot will be implemented with NLP frameworks like Dialogflow, Amazon Lex or Microsoft bot framework whichever suits best to the requirements. Any of these frameworks can be used to encourage AI-ML techniques like Natural language processing, entity acknowledgement to process text and carry operations based on context.

Our long-term objective would be to work on removing the hassles to manually implement these bots by manually feeding and building a conversation flow structure. We will attempt to automate the process so that any institute can easily implement these bots on a single click. Python and its data-rich libraries will be used to scavenge information from an institute's current website and to structure all information in AIML files so that we can use AIML with python and deliver all information readily without any hassles.

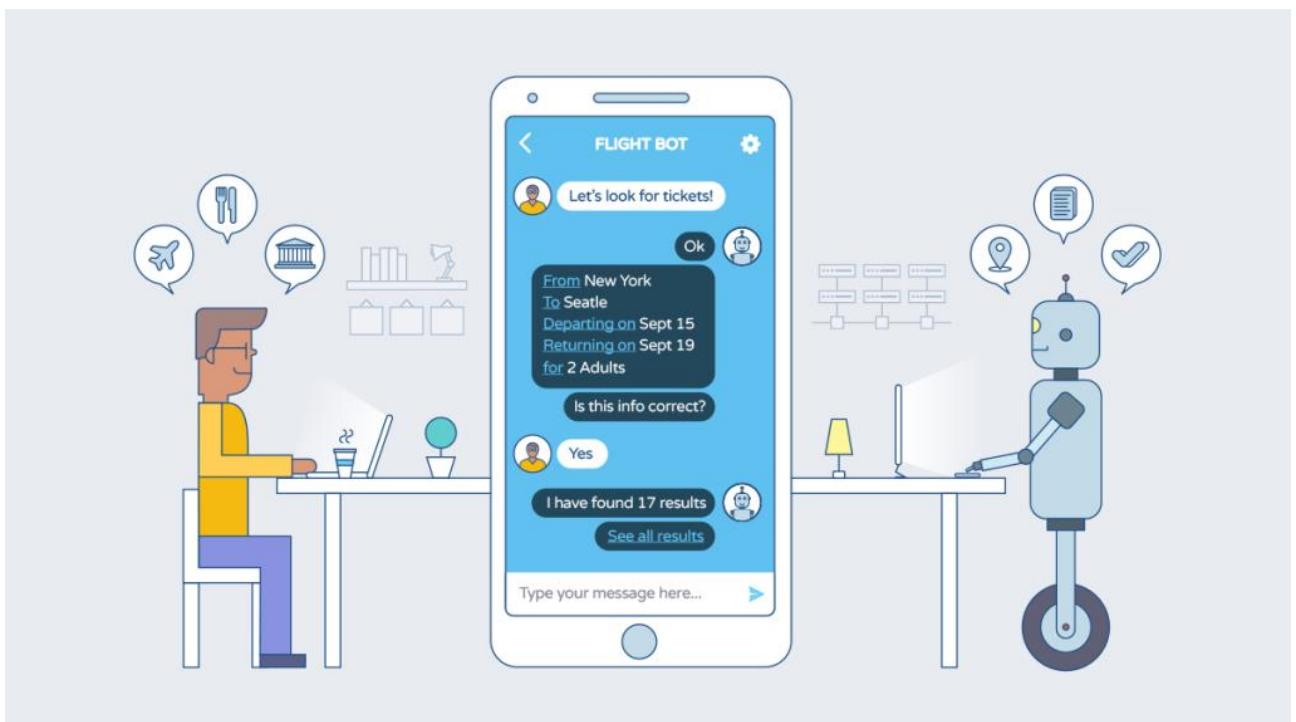


Figure 2 : Chatbot conversation example

Chapter 2 - Literature Survey

The education sector has only its website as its most prominent medium to communicate/advertise courses and achievements to its prospective students. Such smart applications are a major commodity of commerce within the education sector. The expanding utilization of innovation in regular day to day existence is changing how studies learn and retain data. It is a direct result of computerized reasoning that the teachers today can give a customized learning experience to the studies. The scientists have created frameworks that can consequently analyze whether studies can comprehend the study material or not. Chatbots or savvy conversational devices work to improve student's and teacher's joint effort.

2.1 Chatbots in Industry

Learning Through Chatbots

A famous use of man-made consciousness is coaching frameworks that give a customized learning experience to the students by analyzing their reactions and how they experience the learning content. Correspondingly, chatbots with AI can be utilized to teach the students by turning a class lecture in a progression of messages to make it resemble a chat conversation. The bot evaluates the degree of comprehension of the Student and present the following lecture accordingly.

Botsify is an educational chatbot. It displays a particular subject to the students as content, pictures, recordings or a mix of these. In the learning period, students take tests and present the outcomes to their educators. Along these lines, the educators can without much of a hassle filter the students' performance also.

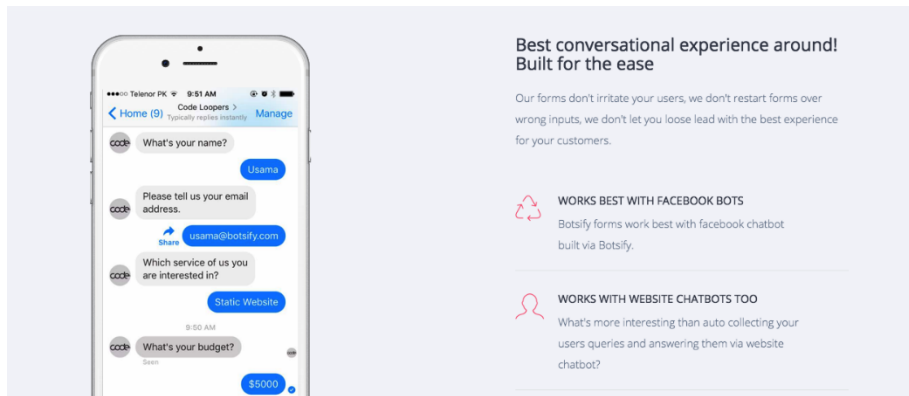


Figure 3 : Botsify example

Improved Student Engagement

The students these days are as of now acquainted with the texting lifestyle and social media life. Regardless of whether they need to speak with one another, examine points or locate the best help, they tend to utilize a virtual assistant to do much of it.

This can be utilized to upgrade the learning procedure and commitment of students in a subject. Imagine chatbots being utilized by educators and students to interact outside the classrooms and offices with alumni. It would turn out to be easier for the students to discover data about the assignments, due dates or some other significant occasion.

CourseQ is a chatbot that is made to support the students, school gatherings, and instructors by giving them a simple platform to interact. The school gathering can utilize it to communicate messages and answer students' inquiries. The students can utilize it to post inquiries from the class and the educators can utilize it to speak with the students, answer queries and settle their doubts.

Up-to-date information for the Institutions

Chatbots are been used beyond just facilitating the students' requests. The chatbots facilitates a lot of data collection. This data contains all the info about students' involvements and their behavior, based on this information the parent institute can implement smart data analysis and come up with better and efficient student compatibility and introduce more loved courses in the curriculum.

The Institute can use this data to track student queries & doubts and then identify the areas of interest where they require improvement. For example, they can analyze if the institute needs to improve the UI/UX of their website or if they should alter/update a web page regarding separate topics. The institution could repeat these processes at regular interval of the year and then adapt accordingly with the rapidly changing requirements and preferences of the students.

2.2 History of Chatbots

Machine Learning and Human & machine interaction are facilitated by a Software concept known as chatbot. It implements Natural Language Processing (NLP) and is used to comprehend the meaning of a conversation happening with a real human. The complexity of implementation depends on the domain of usage.. A relatively open domain demands a considerably big storehouse of knowledge for the bot, while a closed domain can work with a relatively smaller knowledge store.

In the initial years of 1960's chatbot technology was initially implemented chatbots and experimented if a chatbot could be perceived as a human being or not by other humans. During the 1980's the research count elevated particularly in the field of Natural language processing interface which lead to the development of complex chatbot building system like Artificial Linguistic Internet Computer Entity. it's known to be one of the first architectures built specifically to facilitate chatbot integrations in day to day businesses, developed in 1995 by Dr Wallace which is now licensed Open source, A.L.I.C.E stands for Artificial Linguistic Internet Computer Entity. This chatbot also possessed cognitive abilities as it was able to create chatbots

that interacts as well as learns from the past interactions to create a cognitive knowledge base. This knowledge base is saved in modernized and evolved form of Extensible Mark-up language (XML) files also known as AIML (Artificial Intelligent Mark-up Language) (Source: Shawar, B.A and Atwell, E, 2007). There are in general two approaches to implement chatbots; a rule-based decision approach where the bot traverses linearly throughout segmented branches of a decisional trees to emulate an expert system. The rather approach involves implementation of advanced technologies like artificial intelligence and machine learning, this enables the bot to learn, retain and expand on its knowledge, in each interaction the chatbot learns from the conversations, generates appropriate query responses and continuously improves over time (Source: Watson, A. 2017).

For the conversations to be simulated by the chat bot with the user, there can be two methods. The bot can initiate the conversation with the user by perhaps greeting it, called as System Initiated. Whereas, when the user initiates the conversations with the chatbot and directs it, it is called user initiated. There can also be a mixture of both of these approaches, called as mixed initiative chatbot system.(Source: Duijst, D. 2017)

2.3 Natural Language Understanding Engine & NLP

Every ML powered technology requires an engine as its heart and that stands true for chatbots too, the chatbot engine in thought to be one key component in a complex chatbot system, also known as Natural Language Understanding (NLU) engine which is complemented by NLP. Natural Language Processing (NLP) is a key part of man-made brainpower (AI). Most usually, NLP empowers frameworks to traverse common language in archives (for example information articles, tweets, messages) and in this way resolves any hindrance between how individuals talk and what information PC's interpret. While these various territories of NLP researches contribute heavily to modern day AI, with the ascent of chatbots and conversational interfaces, Natural Language Understanding (NLU) has been getting more consideration This subfield of NLP centers around understanding of common language. Among the various ways to deal with NLU, the most well-known one at present depends on characterization calculations to group inputs.

While there are a couple of various ways to deal with NLU, they share regular fragments. As a subfield of NLP, NLU likewise depends on lexical and syntax rules to parse normal language. The parser, alongside semantic theory of comprehension, controls the comprehension of natural language. When the underlying language model is built successfully, it should be adjusted to actually understand some unique contexts. For instance, an expression, for example, "short deal" can have a significant importance in finance while "short sale" while referencing a procedure or a cycle, has a considerably less nefarious significance. NLU models need finessing to have the option to recognize two such articulations.

As chatbots and conversational interfaces are occasion increasingly prevalent, it is imperative to specify that in chatbots, NLU is the motor that extracts the intent and the entity from a client's articulation. Basic NLU organizations essentially use AI driven classifiers to rapidly mark new client expressions as a specific kind of intents. While this is surely helpful, numerous chatbots flop in conveying the appropriate responses that match these aims and all the time, conversational trees become inconceivably confounded thus.

Natural Language Processing is the ability for the conversion of conversations and text dialogs to actions/events in binary form that a machine can understand and execute. NLU engines are known to use eclectic AI-ML methods to interpret the language used in two-way communication systems like chatbots. These Communication methods are provided with Machine Learning and Natural Language Processing capabilities to handle abnormal input articulations by humans, for example spelling mistakes can be taken care of by the bots, also the bots can generate closest context of what the user actually tries to say, you may have seen this implemented by google in their search engine.

Dialogflow an conversational framework developed by google, also known as API.ai, Provides a full-fledged NLP engine which extracts the user utterances followed by derivation of intents along with their context. Various intents and an intent flow describe the behavior and also defines the coherent interactions between the bot and the client. Context and Intents extracted are processed to bridge users input/interaction with corresponding actions to be scheduled by the bot to achieve certain goals specified in the bot's behavior. Contexts are processed to classify and distribute user input into clusters which could have different significance based upon then active conversation.

For example :

Conversation 1 : “Tell me john’s age”

This has no previous current context so we need to tell the subjects name

Conversation 2 : “Who is john” , “What his age”

This conversation has 2 messages so after the first query NLU engine needs the previous context to know whose age to extract from the database.

2.4 Artificial Intelligence

Artificial intelligence has given rise to a lot of ground-breaking applications. A lot of research is being done in this field and it is still a budding technology. New avenues are constantly being identified to employ the powers of artificial intelligence. It can be stated as the capability of a computer based system to execute workloads that are commonly attributed to humans.

Reasoning is of central importance in artificial intelligence. This aspect of AI can be stated as the ability to draw conclusions from a situation that serves a fruitful purpose. Reasoning can be deductive or inductive. While in deductive reasoning the logical base of the situation gives absolute assurance of the conclusion, it only provides support to the conclusion without any guarantee.

The general implementation of AI can be expressed in simple terms and does not require intricate understanding. First and foremost, a numerical presentation of the final outcome is established. Secondly, the data relating to the target outcome is accumulated and various conditions are studied to increase the probability of realizing the target outcome. There are various factors and aspects that play an important role in determining the target outcome. A weight is attached to each of these factors and aspects. Then it is the AI which uses these weights to come up with a prediction formula. Then these formulas are utilized to process the input data and come up with useful decisions.

The terms ‘Strong AI’ and ‘Weak AI’ can be distinguished on the bases of the disparate goals that these two variants try to fulfill. Whereas “Strong AI” attempts to build a type of artificial person, ‘Weak AI’ only attempts to create an information processing system that gives the impression of possessing human mind like abilities. While the former tries to realize mental abilities like consciousness, the later attempts at realizing mental cognition in order to automate mundane tasks.

2.5 Artificial Intelligence Methodologies

Many efforts have been made to come up with methodologies to comprehensively understand and mimic the human language. In order for the machines to be perceived as intelligent, it is imperative that they understand and mimic the human language. This gave rise to Natural Language Processing (NLP) to be developed as a sub-branch of Artificial Intelligence. It allows the machines to acquire the ability to comprehend high level language. In these times, NLP is quickly gaining grounds owing to the substantial developments in computational powers and big data technologies, thus enabling the researchers to use these technologies in disparate fields like biology, finance, international commerce, pharmaceutical industry, economics etc., and obtain some meaningful results.

In order to extract information from some written data, one of the primary tasks is to identify entities. This is precisely the task of Named Entity Recognition (NER). It attempts to identify and separate into different categories various named objects like proper nouns, mathematical figures, organization names, times and the likes. Named Entity Recognition is particularly helpful in providing replies to some real-world inquiries.

Named Entity Recognition is pretty useful in extracting information like names of people and organizations and any 'entities' that appear in the context of a particular input article. This enables to attach various suitable tags to each article thus making the organization of articles in a hierarchical fashion easy. Content of the articles is then easily discoverable on the request of the user.

Stanford University has successfully implemented the Named Entity Recognizer using Java. It is popularly known as a CRFClassifier. It is an implementation of linear chain Conditional Random Field(CRF) sequence models. This is a basic implementation which can be used by anyone to train their own model using labeled data.

For Python users, there is a popular library called Natural Language Toolkit (NLTK). It provides already implemented modules of a range of NLP algorithms.

We also need to extract some subjective information from an article in order to comprehend it better. For this purpose, sentiment analysis is used. It is capable of mining the source article for subjective and contextual information. It is used by corporates to assess the sentiment and brand

image of their product by monitoring social media communications. However, it is usually limited to only basic sentiment analysis metrics. This is just like analyzing the tip of the ice berg. There is a lot of information that can be extracted and comprehended. For this purpose, deep learning methodologies are used.

Part of Speech (PoS) is another methodology that is used to break down an article and comprehend it's meaning. PoS works by breaking each sentence into various lists like lists of tuples, lists of words, list verbs, lists of nouns etc. A tag is used in this context with each word that identifies it as a noun, verb, adverb, adjective etc. It is implemented using the Default Tagger class which accepts a tag as input.

2.6 Chatbot Architecture

Antedated chatbots had support for only linear conversations, but the new complex chatbots are capable of implementing many different adjacency pairs at the same time, this requires state management between multiple different conversations to be implemented. They also require the ability to associate data in different contexts. This is known as the ability to preserve contest & state.

Picking the right Architecture relies upon what specific domain the chatbot will have. For instance, you may ask a chatbot something and the chatbot answers to that. Perhaps in mid-discussion, you leave the discussion, just to lift the discussion up later. In view of the kind of chatbot you decide to develop, the chatbot could possibly spare the discussion history. For narrow domain a pattern matching architecture would be the ideal choice. In any case, for chatbots that manage various contexts or numerous intents, broader domain. In these cases, refined, best in class neural network architecture, for example, Long Short-Term Memory (LSTMs) and fortification reinforcement learning agents are your best option. Because of the variety of chatbot use, the design will change upon the one of a kind need of the chatbot.

A chatbot can be bifurcated to four parts namely

- **Front-end (Client Interface)**

Front end is pretty much self-explanatory, its where the bot and the user actually interact

- **Knowledge base**

Natural language processing engine (Knowledge base) is the core component that extracts what users say along with the context at any given time and translates the language to inputs that the chatbot can further analyze. As our chatbot is domain specific, it must support a number of operations. NLP engine implements advanced ML algorithms to identify the input and then maps to the available intents that the bot supports.

- **Backend (Context and State Management)**

Backend is where the core Natural Language Process engine and context extraction takes place to maintain and manipulate context states.

- **Agent for Dialogue Management (Training Data)**

It deals with the real context of the conversation. For instance, the client may state "He needs to arrange cake" and the bot may take the request. At that point the client may state "Change it to espresso", here the client alludes to the request he has put before, the bot should accurately decipher this and make changes to the request he has set before affirming with the client.

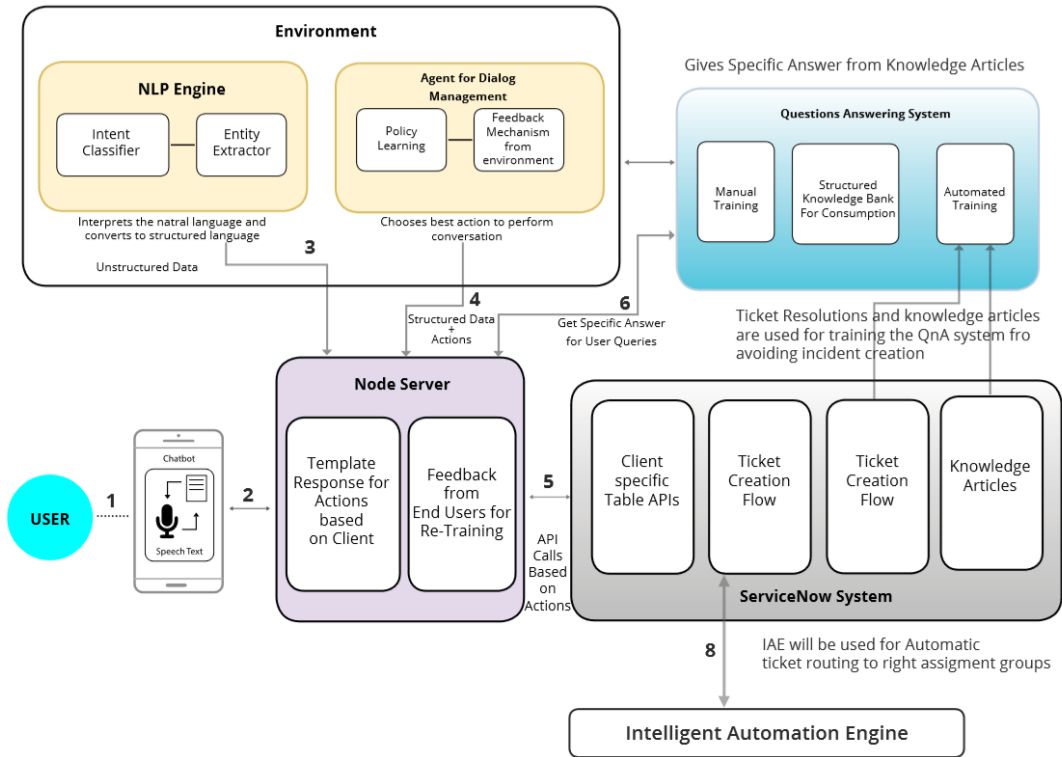


Figure 4 : Advanced Chatbot Architecture

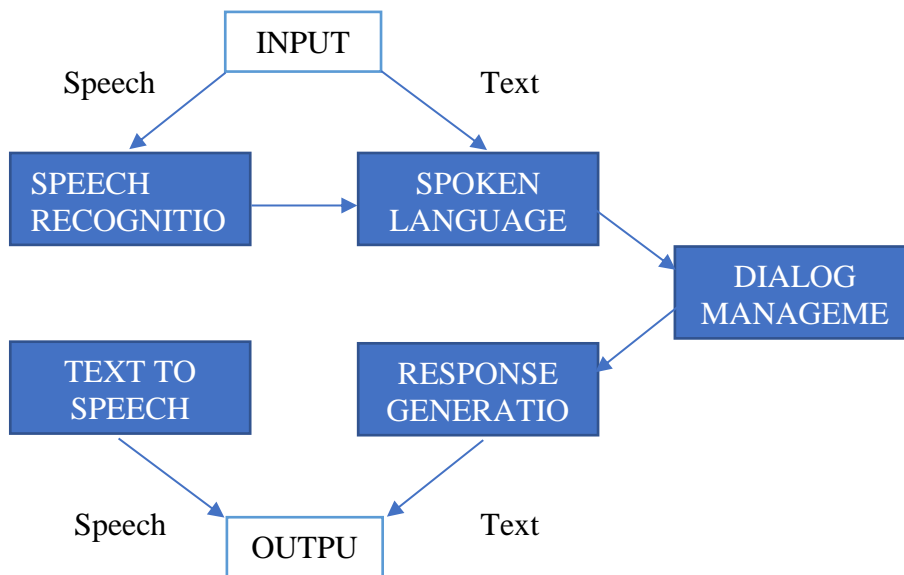


Figure 5 : Simplified chatbot architecture

Chapter 3 – System Development

Clear Analysis and a potential solution is summarized in this upcoming section, this section ahead will list out all the necessary requirements and the ways this chatbot can be implemented.

This includes the following:

- Addressing the problem statement.
- Design Aspects of the project
- Listing out the non-functional requirements
- Listing out the functional requirements
- Hardware and Software used for implementing this project.

3.1 Project Problem Statement

By Our research in Chapter 2(Literature Review) It is quiet conspicuous that the new gen chatbots help us facilitate smooth and nonlinear computer aided conversations, they are efficient to deliver information in a contemporary way, it is also evident that education sector while facilitating modernization of other sectors is now opening up to expand their technologies, there are already some good implementations in the sector using the same or rather similar technologies. We are more focused on a domain specific chatbot that provides users with their browsed information and also to serialize how clients interact and analyze the institute they want to join.

Currently students rely on the knowledge of parents, relatives and acquaintances and obviously the ranking institutes as well questionnaire sites like Quora, to know about any specific institute. It is not practically feasible for the institutes to fix up a real time doubt clearing assistant to assist the admission seekers. This is where our chat bot comes to the aid. It is designed to meticulously help students discover the institute which they desire to go. The bot works on the real time data provided by the institutes itself to increase reliability and increasing transparency for students.

Out chatbot will engineer a way to deliver efficient and personal communication experience between the institute implementing out chatbot and the students/users. Chatbot allows users to be confident and comfortable while conversing irrespective of the user's computer knowledge because a Natural Language in being used within the messages instead of a robotic communication language. The chatbot delivers an efficient, affective and accessible service where all conversation and interactions take place within a single chat window, this reduces users hassle to jump around various website pages through cumbersome navigation. With the powers of speech recognition and optional audio enabled responses, the chatbot interaction is projected to be very efficient for physically challenged users. Chatbot enables the visually challenged to use the website in a very simple way.

3.2 Proposed Solution

We will attempt to implement a Domain Specific Knowledge System to deliver responses to most frequently asked questions related to a particular institution during admission time.

The implementation of this project on a counselling portal will be potentially useful for the already frustrated students who have been browsing information regarding various institution. Even if Universities/Institutions already have website to provide the required information a personal interaction with the user is always preferred as it efficiently clears users doubts and processes queries.

The main goal of this project is to ease the burden of continuous calls and e-mails on institute authorities and at the same time make it easier for the students to conveniently query information without having to look or browse several webpages to fetch answers to frequently asked questions.

Users can query any institute related information in natural language one with the are comfortable with such as asking "What was the last 5-year MHRD rank of the university", "Give me some information of the courses available in the university". Etc.

3.3 Software Engineering Models

Settling on a suitable procedure is vital for the general improvement and development of any product application to guarantee a practical time period for each phase of the project and ensure requirements are clearly sketched out. Different software engineering models will be talked about and considered for the development and plan of this product. This area will feature the various development strategies that are most appropriate to this project.

3.3.1 Waterfall Model

This is one of the ancient ones but still is quiet significant and robust, initially implemented in the initial years of software development, this model works as the basis of all modern development models. The waterfall model is a prescient way to deal with program development that comprises of 5 phases that consist of; requirement gathering, requirement analysis, project designing, development & Integration and finally testing. Each stage is finished in this particular order. A significant disadvantage of the waterfall model is that it is entirely rigid, as the development process is separated into stages. Each stage is given a particular deadline all together, for a deliverable to be created toward the end of each stage. The progress and milestone tagging of the development is estimated from the project deliverables schedule, design and test plans are e key component of this project. As each part of the plan is defined towards the start of the development lifecycle and initial targets have been set it's hard to incorporate new requirements/functional adjustment might have adverse effects on the project flow and deliverables expected date. The waterfall model schedules many high risk and difficult parts towards the end of the project life cycle, which tend to be a major con for this approach.

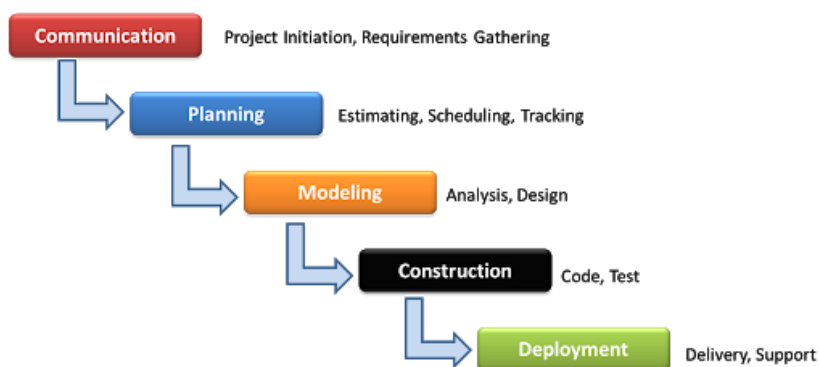


Figure 6 : Waterfall model of software development

3.3.2 Agile/Incremental Model

This product strategy was incrementally derived from the waterfall model. The application is structured, created and tested via an iterative incremental development stage. Toward the finish of each stage a subsystem is created. The project will advance in complexity as new functional requirements are probably going to be found and included in each stable build, developing over the usefulness from the last build resulting in overall improvement of the application.

It is in practice for software development to be bifurcated in stages, it is important that component versions used inside the product are overseen throughout the lifecycle utilizing version control and production tools (DevOPS), for example, GitHub. Each build will exist for couple of weeks before an increment is implemented over the existing build. Input can be given on any existing bugs or newer bugs may arise in the application. Deconstructing the increments of the current build over previous different build's can bring down the dangers related with increment and can help narrow down the parts which cause the errors/bugs. In this way newer requirements are managed into smaller functional components to be implemented and integrated toward the finish of each build.

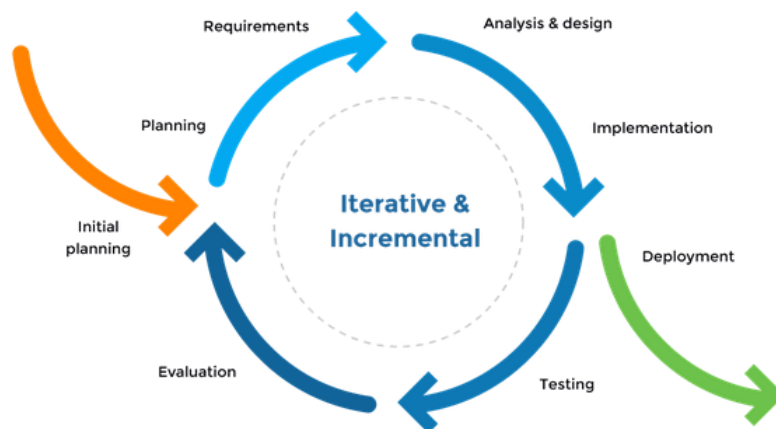


Figure 7 : Agile model of software development

Testing and troubleshooting are made simpler as issues are recognized in the very beginning during small manageable cycles. This philosophy is adaptable during implementation as new requirements are effectively incorporated at each build and a, refreshed implementation is released.

3.3.3 Chosen Methodology

The most appropriate model for our project applications is the Incremental/Agile model , software like these require continuous updates, bug fixes, regular testing and it is very handy to implement version control for such projects, often during the integration phase we see some new requirements come up, the flexibility to handle such situation are the strongest points to adapt the Agile model, this model will implements a iterative build structure which is proven robust in the IT industry for quite a while now.

3.4 Functional and Non-Functional Requirements

The functional requirements and the non-functional requirements are analysed from the data collected in the survey. Functional requirements collectively define a system or its components. These describe the functions and features that the software must perform. A function is the inputs, its behavior, and outputs. These can be a calculation, data manipulation, user interaction, or any other functionality which defines what function a system is likely to perform. It helps to capture the intended behavior of the proposed system.

A non-functional requirement, on the other hand, defines the quality attributes of a system. They represent a collection of specific standards used to determine the various operations of a system. Example, how fast does a website load? These include various parameters such as security, reliability, maintainability, performance etc.

3.5 Functional and Non-Functional Requirements

“The functional requirements and the non-functional requirements are analysed from the data collected in the survey. Functional requirements collectively define a system or its components. These describe the functions and features that the software must perform. A function is the inputs, its behavior, and outputs. These can be a calculation, data manipulation, user interaction, or any other functionality which defines what function a system is likely to perform. It helps to capture the intended behavior of the proposed system.

A non-functional requirement, on the other hand, defines the quality attributes of a system. They represent a collection of specific standards used to determine the various operations of a system. Example, how fast does a website load? These include various parameters such as security, reliability, maintainability, performance etc.

3.5.1 Functional Requirements

- Allow for the users to register for the university notifications if interested
- Provide necessary information queued for
- Allow users to get direct links to the queried information
- Users will be able to converse with the chat-bot through voice or text commands and it will understand what the user is saying through natural language understanding provided through the integration of Dialogflow API.
- The chat bot should be able to maintain the conversational state when the context may be unclear through previous messages and conversations.
- Provide text and audio responses.

3.5.2 Non-Functional Requirements

- The chatbot should be efficient and quick with very little delay in response time to reply to a user message.
- The chatbot must be reliable with almost no faults or bugs
- The database must be scalable to adopt to a growing number of users
- The chatbot must be secure as sensitive data is being used
- Comply with data protection laws such as the Data Protection Act 1198.
- The use of natural language used to interact with the chatbot promotes human computer interaction.
- Provide accurate responses to input.
- Appropriate handling of unexpected input and correctly inform the user if it cannot provide assistance

3.6 Software and Hardware Specification

The various software as well as hardware requirements that must be considered while implementing the chatbot system are mentioned below.

3.6.1 Software Requirements

- Oracle VM VirtualBox
- Node dependency manager
- Nodemon Hot Reload
- Express
- Firebase
- Angular Framework
- JavaScript
- VSCODE
- Dialogflow
- Fetch API
- BotCopy

3.6.2 Hardware Requirements

- PC running Linux-Ubuntu/Windows to act as server to host chatbot locally
- Processor – intel core i5
- Any kind of smart device.

3.7 Project Planning

The collection of project milestones and deliverables are produced projecting the system breakdown to create a management strategy to stick to the project lifecycle, however there is a possibility of some minor alterations in each iteration as some of the functional requirements may need modification to the project structure. Here we list the steps which are required in order to achieve the project deliverables.

3.7.1 Project Milestones and Deliverables

The set project milestones will serve to be helpful in keeping track of the project progress, allocation of schedules for task completion and judging the achievement of various objectives.

Table 1 : Milestones

Milestone	Milestone Description	Completion Date
1	Literature Review	06/Aug/2019
2	Analysis & Requirements Specification	19/Aug/2019
3	Interim Report / Presentation	09/Oct/2019
4	Design	12/Oct/2019
5	Implementation	20/Nov/2019
8	Semester End Presentation	12/Dec/2018
9	Reference Research Papers Review	12/Jan/2020
10	Web Scraper Implementation	18/Jan/2020
11	Mid-Sem Presentation	20/Jan/2020
12	API Development	25/Mar/2020
13	API Implementation	18/Apr/2020
14	API Hosting	8/May/2020
15	Report Finalization	19/May/2020
16	Semester End Presentation	25/May/2020

The following table consists of the project deliverables, the deliverables show us the outcomes after completion of the project.

Table 2 : Deliverables of the project

Deliverable	Description
1	Interim Report
2	Web based Chatbot
3	Final Report
4	Viva Demonstration

3.8 Project Architecture

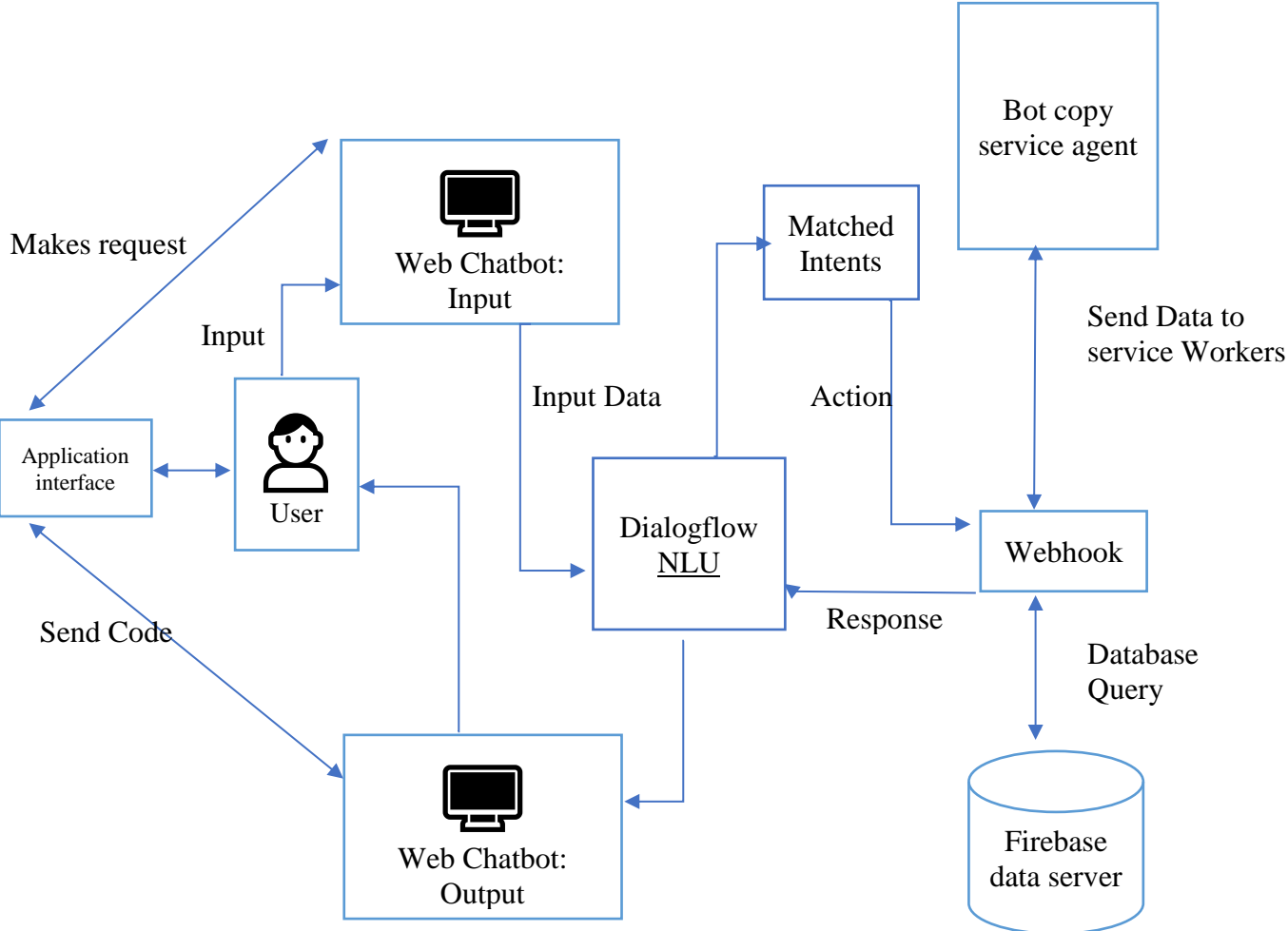


Figure 8 : Cahbot Detailed Architecture

The communication with the chatbot is provided for through a frontend web-based user interface. Users will have the freedom to conduct the interaction with voice or text-based phrases in their natural language. With the integration of DialogFlow users will be served pertinent responses in a rich format such as, cards and images, thus improving the overall user experience for the users because less effort would be required in typing the commands while using the voice-based interaction method to communicate with the chatbot. The integration of DialogFlow allows for easy extension of the chatbot system through the Botcopy interface that can deliver every service in an integrated form.

Our User Interactive Client Interface is designed and implemented using HTML, SCSS, JavaScript and Angular.

This particular framework implements the Model View Control architectural pattern.

Angular helps by compiling and caching view files in order to increase the performance of

the system. Angular also enables the differentiation of view files to be segmented in forms of modules.

Angular Utilizes component inheritance i.e. Angular supports template inheritance. This allows templates to be used within other components, this is a flexible and modular approach that make angular a very suitable platform to design and implement complex UI like this one of a Chat Bot. Ui components can be dynamically reused without replacing code. Views can import components and can render the component conditionally and respective data can be delivered to the component using data binding. Angular is also empowered with inbuilt control structure, for instance “ngif” and “ngif” are few of the most prominent ones.

The Web Client and the Webhook gets the required data in JSON format from the DialogFlow NLU through an HTTP(secure) POST requests. To enable the web route to be implemented as one of the endpoints, the WebHook is defined in the form of controller class. The NLU is designed to post the payload in real time to the system WebHook. On the other hand, the WebHook retrieves data from the DialogFlow NLU upon detecting a trigger of an intent by the user.

There is a fine distinction between the working of a webhook and an API that follows RESTful approach in its design. The data is posted to a webhook from some other module or service upon the occurrence of some event or some update. On the other hand, an API, following a RESTful design, retrieves the required data by making requests. In the project we have used DialogFlow as our Natural Language Understanding Engine, or for short NLU. This is instrumental in enabling the chatbot to get trained in recognizing the various intents and entities that might occur in the user utterances. The design of the chatbot uses mapping of intents in the console of DialogFlow so as to route the utterances of the user to a predefined set of phrases.

The following figure depicts the data flow when the user invokes an intent. It is the responsibility of DialogFlow system to identify the triggering of an intent. This it does through it's training, utilizing a set of typical and well known phrases. The fulfilment webhook receives this data and is responsible for extracting the necessary parameters or entities as well as the action required. Then a custom response is produced and returned back to the user end point in either of the two forms: visual text rendering or in voice format from the webhook. The webhook is tasked with parsing and validating the JSON data received in the payload. It also encodes the custom response produced by the system to the user's end point for both of the formats: text rendering and speech interaction.

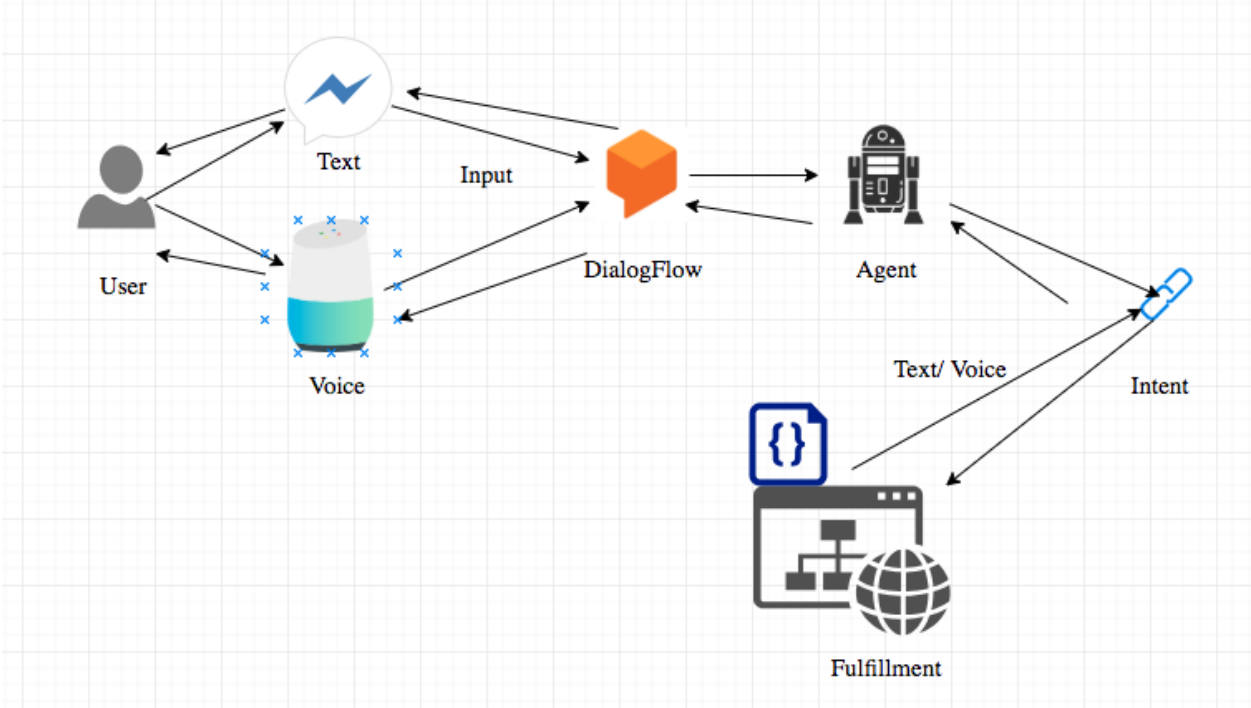


Figure 9 : Data flow diagram for fulfillment

3.8.1 Use Case Diagram

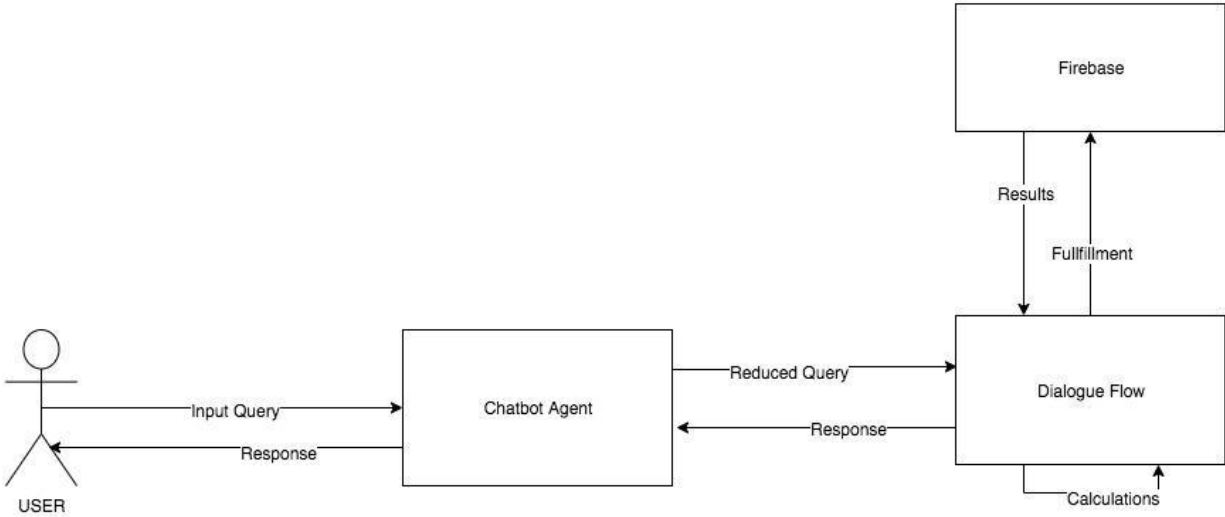


Figure 10 : Use - Case Diagram

3.8.2 Project Structure

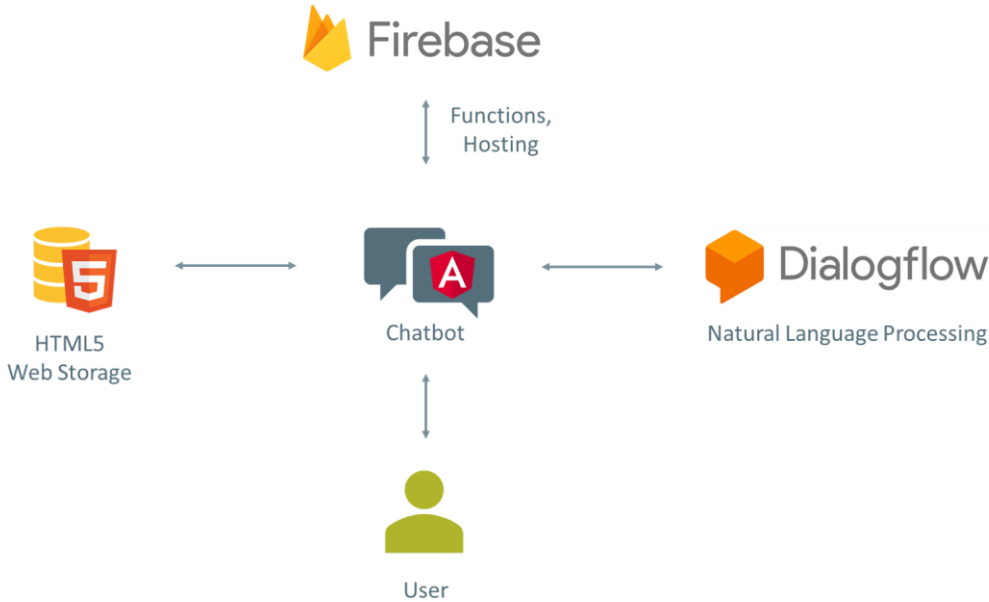


Figure 11 : Project Structure and technologies used

3.8.3 Activity Diagrams

We have drawn the activity diagrams with the aim to clearly depict the flow of information in the form of data and also to depict the major processes involved in the functioning of the chatbot system. These diagrams contain a depiction of the user's behavior when he/she is in interaction with the chatbot as well as the response of the chatbot to the received actions. This diagram helps to simplify and breakdown the various functionalities of the chatbot.

The utterances of the user are posted to the fulfilment webhook as and when the Natural Language Understanding engine analyses the input from the user end and finalizes a context. The endpoints of the webhook are defined as accepting POST requests. Through this route it receives the output of the NLU in JSON format. The webhooks is tasked with finding out the action linked with the intent with which the utterances of the user are mapped to. This action was defined and mapped in DialogFlow console while training. If the webhook is able to find the action in the map then it extracts the various entities in the given utterances of the user and subsequently calls the Fixer.io API. It then reverts back with the pertinent reply in JSON encoded format. In case the NLU is not able to find out the intent of the utterances of the user, then it prompts the user for some more details and information so that it is able to recognize the intent and context of the us

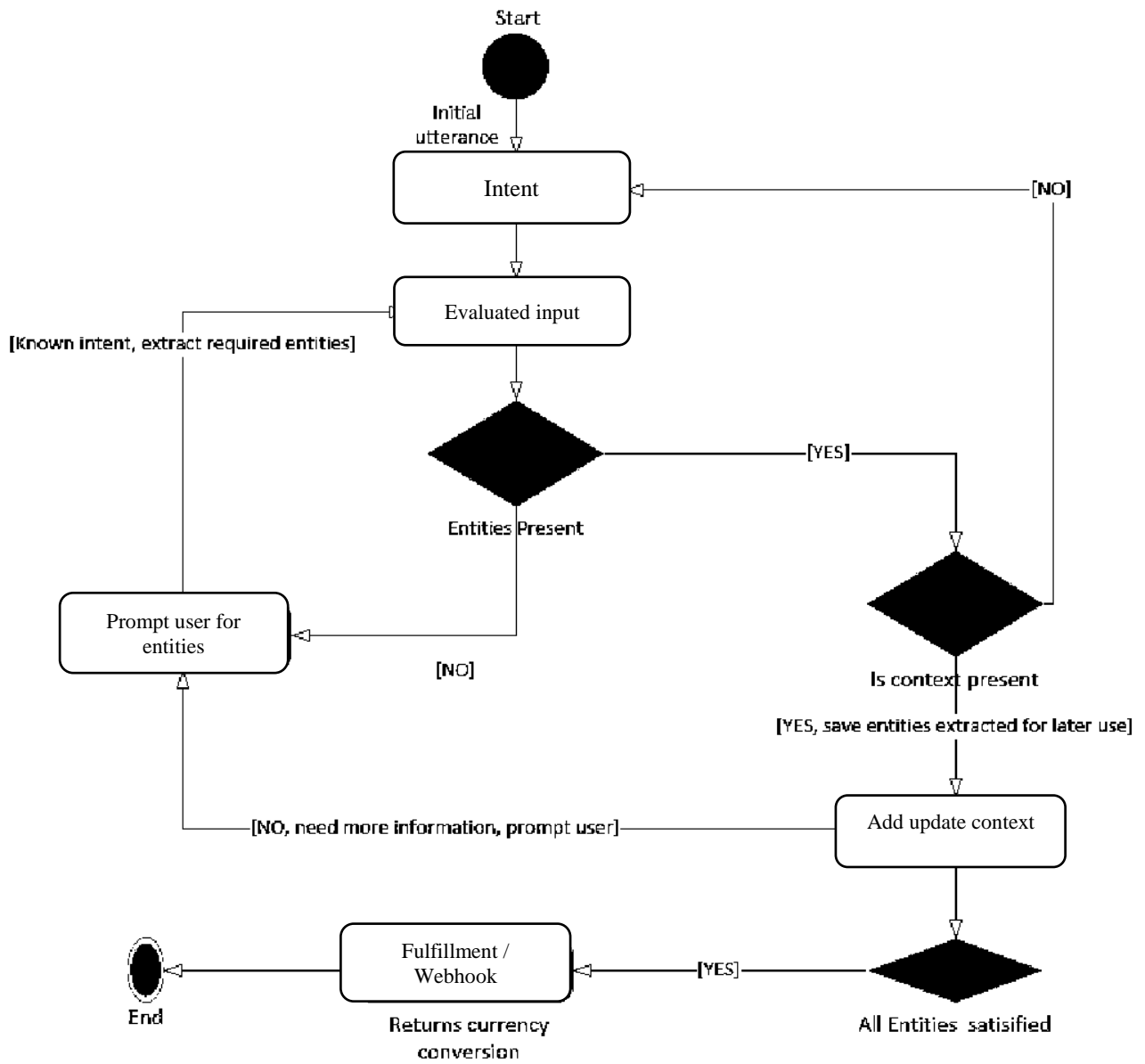


Figure 12 : Activity diagram for user interactions

3.9 Conversational User Interface Design

Interaction will take place within the chatbot in the form of natural language and this will be the focus of design in regards to conversational user interfaces (CUI). “Conversation as a method of interaction is often referred to as the new UI” (Actions on Google, 2018).

(cs.hmc.edu, 2018) define a set of design patterns to adhere to when designing Conversational User Interfaces.

Table 3 : Design Principals

PRINCIPLES		DESCRIPTION
1	Design by personality	The bot personality should provide a natural and unique interaction experience for users. The personality should be designed to guide the users through the interaction which may include being aware of what the user may say or want. The personality is the style and manner the bot holds throughout the interaction.
2	Flexibility in response	Flexible responses should be provided to user input, the bot should have various/random responses to the same user input.
3	Text vs Custom Buttons	It should be clear to the user what input is expected to carry on in the conversation. A fallback response should be used in situations where the input is not recognized such as suggesting other operations that are available. Instead of solely relying on natural text commands from the user, a good UI should render structured options to the user in form of buttons or text.
4	Simplicity in Interaction	Be concise and clear in response to questions. The conversation flow should be straight and not branch out to complex paths.
5	Conversation Flow	The conversation flow is important to ensure that the user is not irritated in situations where the bot may not be able to provide an appropriate answer immediately. The user would then have to undergo a conversation in order to determine the answer to their query.
6	Tasks and duty specifications	The bot should deliver explicit tasks for a particular domain
7	Rigid syntax, then NLP	People are prone to spelling mistakes or using colloquial phrases which may not be understood by the bot.
8	Empathy and emotional state	It is important that the bot builds a rapport with the user and convey emotion in responses based on certain user input.
9	Keep conversation short	Users want to interact with the bot to receive answers or reach a goal quickly. Avoid long winded conversations as it will make the interaction feel burdensome. This helps avoid ambiguity.
10	Triggers and actions	The bot needs to persuade users and seek ways to encourage users to carry out specific actions through the bot.

11	Predict and personalize	Bots can analyse previous input from users for important parameters, in order to anticipate a user's predisposition. It will then provide an answer that would be unique to that particular user
12	Fully/partially automated	NLP should be used for automated tasks
13	Providing a way out	Allow user to begin the conversation again or exit the conversation. The conversation should engage the user throughout the interaction.
14	User boredom	The conversation should engage the user throughout the interaction.
15	Cognitive load	Chatbots should present coherent choices to users to reduce the amount of effort needed for the user to interact with it. This can be achieved through the use of buttons and images.

3.10 Dialog Design

As principle focal point of interaction between the client and the chatbot, which is handled through natural language, the dialog designing is a critical angle to think about when planning the chatbot. To accomplish this; suitable dialog exchange will be structured through the Dialogflow console utility using follow-up and fallback purposes. A specific intent will be set, that once invoked by the client, the chatbot will end the discussion viably paying little importance to the conversation context. The conversation is the assortment of words and expressions used to react to client input. The development will be planned with human like expressions to react to clients with human nature so as to sound progressively natural. Another structure aspect will incorporate fallback intents. In the event that a client expression can't be mapped to an intent, the chatbot will react with a message expressing it didn't match the client input to an intent and brief the client to rephrase the inquiry or give more subtleties to the query. The conversation is additionally structured using follow-up intents, these give a progressive conversational stream while communicating with the chatbot. Follow-up intent separate more data from clients as and when required, for example; a client may state; The chatbot knows it needs more data from the client, for example, date, time and telephone number and will in this manner use follow-up plans to extricate the necessary data from the client.

User: "Can I get some information about the university please"

User: "please tell me the last 5-year rank of the university"

Dialog flow suggest planning a straight discourse for communications where information present in the discussion is extracted to assist the clients to accomplish their objective, this is applied to the design specification and also applied to the chatbot as it is an task-based interactive chatbot that comprise of educational institute domain knowledge. The client input can be conveyed in multiple forms.

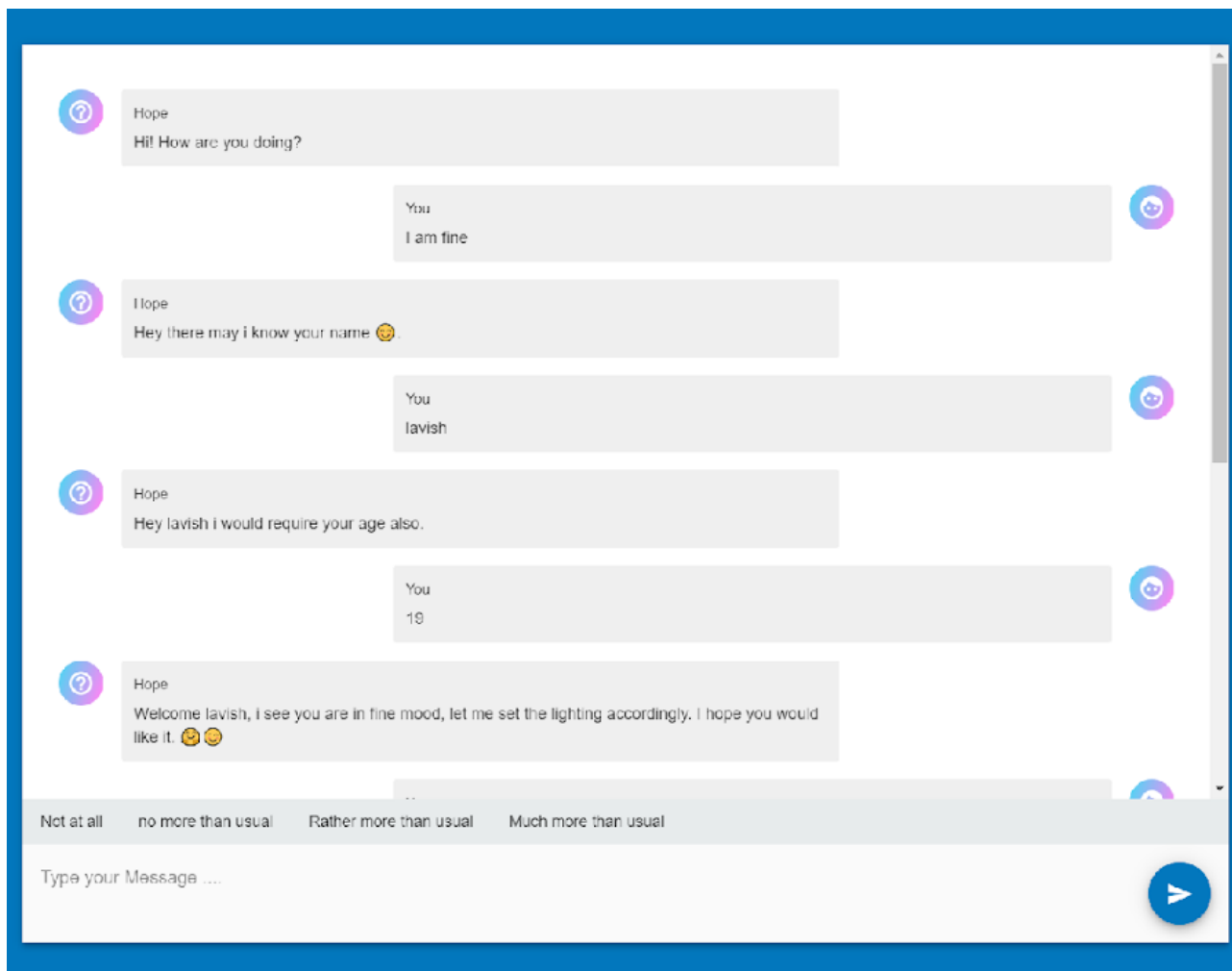


Figure 13 : Our first prototype

The UI of the bot pursues a simplistic format so it is intuitive and productive for clients, in contrast with different types of customary UI. Clients can legitimately pose an inquiry to the chatbot utilizing the text input box or by clicking the Mic icon. When the symbol is clicked first time the browser will request you to enable microphone access, once access is approved device microphone gets activated and the user is notified that he/she has begun Voice Synthesis and the chatbot reaction will be shown on screen in a split second. the infamous google assistant utilizes specific colour for each response. The utilization of various colors will be used to differentiate the chatbot's reaction and the client's input.

User: "Tell me about CSE department?"

User: "What's the camps like"

User: "Tell me about the hostel life"

Clients will have the option to see data all the time without the need to explore/navigate through different website pages.

3.11 Project Integration And Testing

This section describes how different components of the project are integrated and served as a service, the primary architecture of the chatbot is Model View Controller based, and depends on Angular Karma for integration testing. Appropriate Code coverage testing and Chatbot specific testing is also carried out and are presented in Chapter 5(Performance Review).

3.12 Deployment Diagram





Our client end of the chatbot is implemented using vanilla JavaScript and HTML5, we further in our project will be implementing our chatbot using the BotCopy Plugin to enable our legacy website compatible with Rich Media content and a Beautiful customizable chat UI, we also used the Dialogflow SDK for handling queries, the existing SDK was sufficient for most of our interactions furthermore, botcopy enabled us to use voice synthesis for interactions along with configurable styling using SASS(Simply Awesome Style Sheets) and HTML 5. The queries were handled using the Async Post request, an Asynchronous Promise request emits the user input to corresponding Dialogflow API where the request is then interpreted by the dialogflow engine and is mapped to the appropriate intent and then sent to the fulfillment webhook (Fallback Service Worker API) to provide contextual response to the user. The Fallback Service Worker Webhook is again an Asynchronous request to our API host server. The server then returns search results as a JSON Object, this request is made using the Dialogflow console. The console provides us with a NODE JS server to use pattern/entity matching to extract context from text among other Artificial intelligence and machine learning methods.

```
function send() {
  var text = $("#input").val();
  conversation.push("Me: " + text + '\r\n');
  $.ajax({
    type: "POST",
    url: baseUrl + "query?v=20150910",
    contentType: "application/json; charset=utf-8",
    dataType: "json",
    headers: {
      "Authorization": "Bearer " + accessToken
    },
    data: JSON.stringify({ query: text, lang: "en", sessionId: "somerandomthing" }),
    success: function(data) {
      var respText = data.result.fulfillment.speech;
      console.log("Response: " + respText);
      setResponse(respText);
      speak(respText);
      $("#response").scrollTop($("#response").height());
    },
    error: function() {
      setResponse("Internal Server Error");
    }
  });
}
```

Figure 14 : Dialogflow data handling

App component to facilitate the main component of the angular application

```
import { Component, OnInit, HostBinding, OnDestroy } from '@angular/core';
import { ThemingService } from './core/theming.service';
import { Subscription } from 'rxjs/Subscription';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent implements OnInit, OnDestroy {
  themingSubscription: Subscription;

  constructor(private theming: ThemingService) { }

  @HostBinding('class') public cssClass;

  ngOnInit() {
    this.themingSubscription = this.theming.theme.subscribe(theme => {
      this.cssClass = theme;
    });
  }

  ngOnDestroy() {
    this.themingSubscription.unsubscribe();
  }
}
```

The main ask bot function that facilitates user queries, delivers users queries to the Dialogflow Api and the also renders a successful interrupt returned but the backend.

```
private initial() {
  this.getResponse('Hallo').subscribe((r: any) => {
    this.addToChat({ text: r.result.fulfillment.speech, bot: true });
    this.possibleAnswers.next(['Awesome', 'great']);
  });
}

askBot(input) {
  this.possibleAnswers.next([]);
  this.addToChat({ text: input, bot: false });
  this.getResponse(input).subscribe((r: any) => {
    r.result.fulfillment.messages.forEach(message => {
      if (message.speech) {
        this.addToChat({ text: message.speech, bot: true });
      }
      if (message.payload) {
        if (message.payload.response.types) {
          const list = message.payload.response.types;
          this.addToChat({ text: message.speech, bot: true, selectionList: list });
        }
        if (message.payload.response.possibleAnswers) {
          this.possibleAnswers.next(message.payload.response.possibleAnswers);
        }
      }
    });
  });
  this.checkAction(r.result.action, r.result.parameters);
}
```

3.13 Implementing BotCopy Chatbot UI

About Botcopy

Bot copy is rich chat UI that bridges Google Dialogflow agent with content-rich, custom media Web Chat User Interface, We can implement dialog flow agents to our legacy websites using JavaScript injection, no need to implement a custom chat UI, BotCopy injects a rich UI without any hassle, this is a primary feature we need as most of university's have a legacy website that does not implement our modern Front end libraries like Angular. We can control the look of the Chat UI and how it delivers content using the BotCopy config generator, rarely requires any sort of coding.

Botcopy client plugin is a customizable playground, edit and serve as per needs.

Its config generator enables us to choose custom avatar, color scheme and font face to seamlessly integrate with our existing website. The team can also be updated in realtime.

Most importantly BotCopy Enables us to implement Rich Media content delivery, Rich Media like suggestion chips, card layout, carousel, embedded video, and images are a prominent feature of Google Dialogflow and BotCopy enables our UI to benefit with such rich interactive media.

Absolutely no code required!

Currently Bot copy successfully supports Cards, videos, gifs, voice, carousels and lastly the most important for our bot suggestion chips. Many more Rich Media Components are as of now being worked , Interactive List view is one of the most prominent and aspired Rich Component under development right now.

What problems does Botcopy solve?

Until Botcopy, there's been no easy way to host Dialogflow agents on your website in a rich, branded way that you control. This is a huge problem because you're forced to run your bot through Facebook's messenger plugin, which requires a login and isn't branded at all and looks tacky on your website.

Or, you can use a Drift or Intercom-type service that offers a nice looking web chat, but doesn't have the NLP and machine learning that Dialogflow offers. Now, at long last you can have both, brains and beauty, on any website, when you run Botcopy with Dialogflow.

For whom your Product/ Service is for?

- Developers
- Dialogflow users
- Bot agencies
- Bot freelancers
- Bot companies
- AI-first companies
- Tech companies
- Bot designers

Deployment code for the chatbot widget on any website.

```
• <script type="text/javascript"
•   id="botcopy-embedder-d71cfheammjct"
•   class="botcopy-embedder-d71cfheammjct"
•   data-botId="5ddf94512d639aaf3c36b7dc" |
• >
•   var s = document.createElement('script');
•   s.type = 'text/javascript'; s.async = true;
•   s.src = 'https://widget.botcopy.com/js/injection.js';
•   document.getElementById('botcopy-embedder-d71cfheammjct').appendChild(s);
• </script>
```

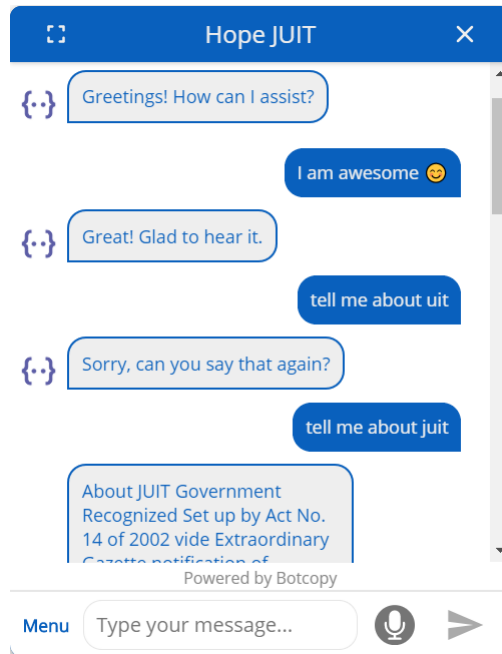


Figure 15 : Botcopy Implementation

3.14 Implementing Web Scraper

In order for the chat bot to provide the user with pertinent results, it is paramount that the bot has all the informational resources at it's service. It needs a database of information on which it performs specific operations to arrive at the optimum response for the user. This knowledge base ,in our case, is the informational repository of the university website. We need to represent all the information present on the website in Java Script Object Notation format in order for the bot to perform search operations on it using the input search query from the user. Creating JSON data of this voluminous data manually seems an infeasible idea. So, we felt the need to automate this task.

In order to automate the informational resource creation process, we wrote a web scraper that would crawl over each link, starting from the homepage given as initial starting input, and scrape each web page for relevant information. We used Python 3.7 to implement the scraper and used BeautifulSoup module to parse the website responses received.

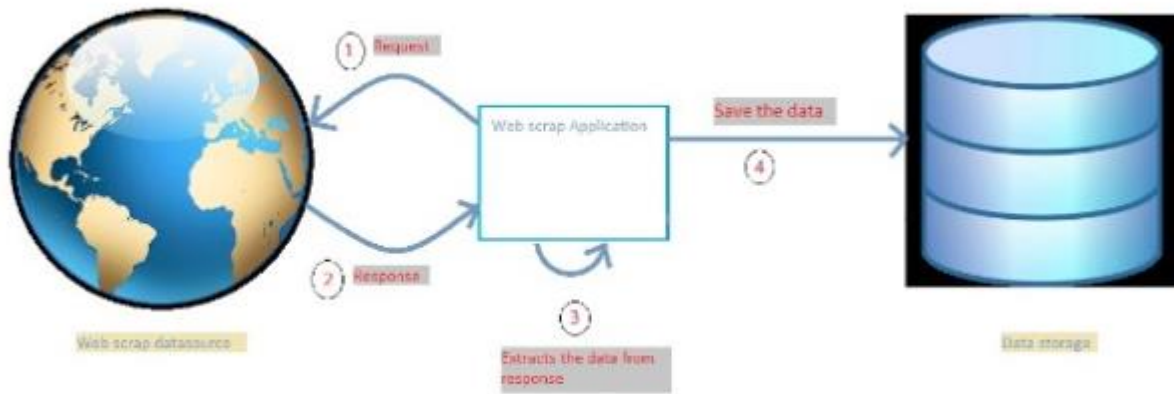


Figure 16 : Web Scraper Architecture

. It would label different information scraped from the websites using predefined rules and populate the JSON data object. The basic structure of the JSON object is given as under.

```
{
  "headings": "",
  "linkTitle": "",
  "linkUrl": "",
  "pageAbout": "",
  "pageTitle": "",
  "relatedLinks": [{
    "linkTitle": "r",
    "linkUrl": ""
  },
  {
    "linkTitle": "",
    "linkUrl": ""
  },
  {
    "linkTitle": "",
    "linkUrl": ""
  },
  {
    "linkTitle": "",
    "linkUrl": ""
  },
  {
    "linkTitle": "",
    "linkUrl": ""
  },
  {
    "linkTitle": "",
    "linkUrl": ""
  }
}
```

```

}
]
}

```

The data from each web page of the university web site is scraped, labeled and populated in this given format. This is a time taking process and takes 2 to three hours to complete, depending on the website structure and content.

```

url=url+part+"/"
url=url+elem
try:
    if(sem==1):
        if(tag[tag_len-1]=="Done"):
            x={"Department":tag[tag_len-2],"Link":url,"Paper":course,"Semester":"","Year":tag[tag_len-3].replace("Done, ", "")}
        else:
            x={"Department":tag[tag_len-1],"Link":url,"Paper":course,"Semester":"","Year":tag[tag_len-2]}
    elif(tag[tag_len-1]=="Done"):
        x={"Department":tag[tag_len-2],"Link":url,"Paper":course,"Semester":sem,"Year":tag[tag_len-3].replace("Done, ", "")}
    else:
        x={"Department":tag[tag_len-1],"Link":url,"Paper":course,"Semester":sem,"Year":tag[tag_len-2].replace("Done, ", "")}
except:
    print("FILE ERROR : {}".format(elem))
    print("Tag->{}".format(tag))
data.append(x)

```

Figure 17 : Web Scraper Implementation

3.15 Implementing RESTful API

The service would be called by various clients from varied platforms (android, ios, web etc) which required the bot response generating service to be deployed on a server as an API. This would make our app very portable, modular and adaptive to changes and improvements. Since this service would not be handling any state changes, we opted for a RESTful design.

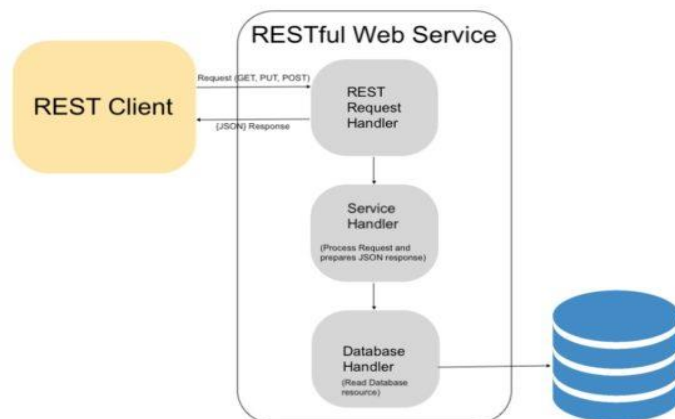


Figure 18 : RESTful Web Service Architecture

The Concept of Representational State Transfer was first given by Roy T. Fielding in his Ph.D dissertation [Reference 17]. In a RESTful architecture the server is not concerned with maintaining any state of the interactions with the clients.

We implemented our API using the RESTful Architecture. The API listens to the port 3000 on the relative URL '/search/'. The client makes a request to this public, non-restrictive, endpoint using the GET request and supplies the search string to the API. Any text appearing in the request URL after 'search/' is treated as the search query string.

```
1  const http = require('http')
2  const port = process.env.PORT || 3000
3
4
5  const app = require('./app')
6
7
8  const server = http.createServer(app)
9  server.listen(port);
10
```

Figure 19 : API Request Handler

The gateway passes the search query string to the API service handler. The API service handler then performs some basic formatting of the input search query string and invokes a worker which is tasked with fetching the stored data of the university website, as provided by the web scraper, in Java Script Object Notation (JSON) format.


```
const express = require('express')
const router = express.Router()

router.get('/:searchQuery', (req, res, next) => {
  const query = req.params.searchQuery;
  const searchResult = search(query.trim())
  res.status(200).json({
    message: 'ET VOILA',
    query : query,
    result: searchResult
  })
})

module.exports = router;
```

Figure 20 : API Service Handler

The worker thereafter performs the search on the fetched data. The searching mechanism is the same as developed by us previously. It performs a fuzzy search on the collected JSON data using the input search query string. We have the option of adjusting the amount of responses generated by the worker. Currently it is set to three responses for each search query.

```

query = query.toLowerCase();
var terms = query.split(/\s+/);
var fuzz = 0.7;
console.log('got query: ', query);

for (var j = 0; j < data.length; ++j) {
  data[j].score = {'linkTitle': 0, 'pageTitle': 0, 'pageAbout': 0, 'headings': 0, 'total': 1};
  for (var t = 0; t < terms.length; ++t) {
    var maxTermScore = 0, maxField;
    for (var f = 0; f < fields.length; ++f) {
      var termFieldScore = data[j][fields[f]].score(terms[t], fuzz);
      if (termFieldScore > maxTermScore) {
        maxTermScore = termFieldScore;
        maxField = f;
      }
    }
    data[j].score[fields[maxField]] += maxTermScore;
  }
  for (var f = 0; f < fields.length; ++f) {
    data[j].score.total += Math.pow(Math.max(data[j].score[fields[f]], 0.1) - 0.1, 1);
  }
}

data.sort(function(a, b) {
  return b.score.total - a.score.total;
});

var results = data.slice(0, 3);
console.log('results: ', results);

return results;
};

module.exports = search;

```

Figure 21 : API Service Worker

The worker produces the result in JSON format and returns to the API request handler which returns the results to the requesting client via request response.

\

3.16 Testing

Testing is surely a very integral as well as important part in the entire lifecycle of software development. In this process various operations and procedures are undertaken in order to interpret any of the software limitations. This process helps in identifying and documenting various errors and bugs that exist in the system. This process is instrumental in improving the overall quality, functionality and standard of the chatbot system.

3.17 Git Version Control

```
.env.example  
.gitattributes  
.gitignore  
Procfile.txt  
README.md  
Vagrantfile  
after.sh  
aliases
```

Figure 22 : Git Version Controlling

Throughout the project there was a git repository maintained to store and version control the application for easy and rapid Agile development. As the application was being iterated git commits were made on various milestones to cement the progress till then a different branch of the GitHub repository

Chapter 4– Performance Analysis

4.1 Performance Analysis

The backend implemented by us is quite efficient in analyzing the performance of our system, this backend can also collect a huge amount of data for various analytics by the Institute.

The Functions tab of our firebase console displays all the interactions and the data flow between those interactions, this is also a good play ground to test the working of the chatbot.

We are able to see the error logs which give a detailed description of the error.

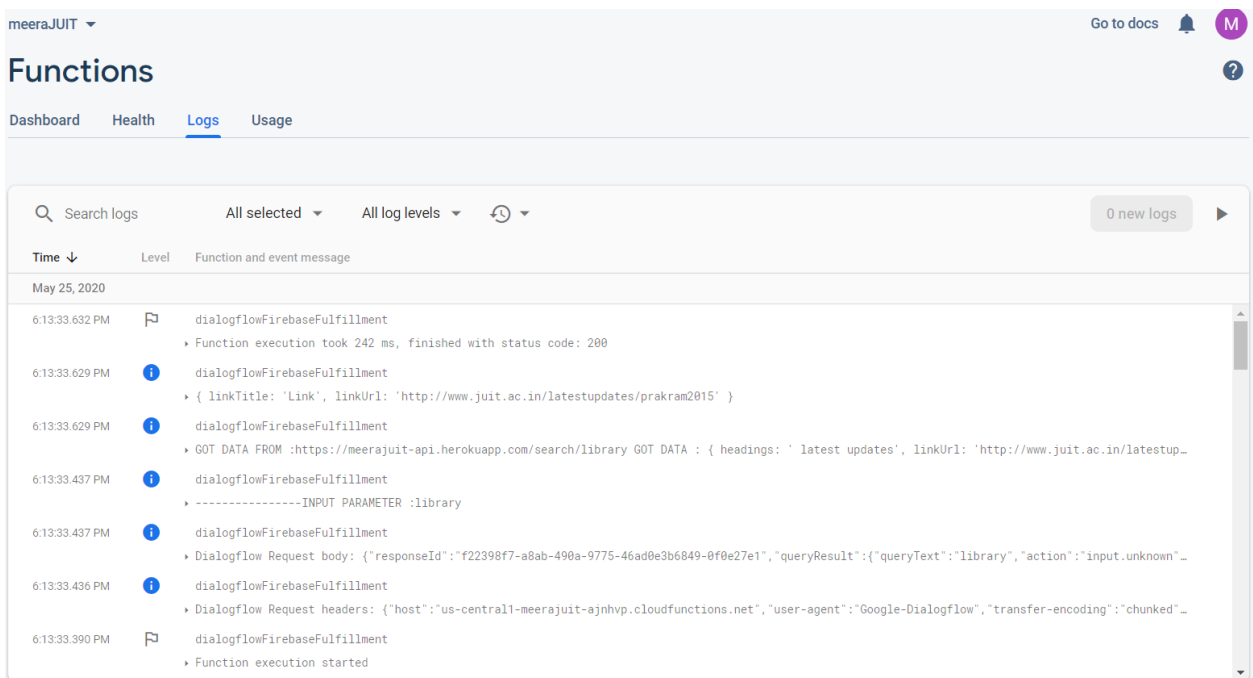


Figure 23 : Firebase analytics report

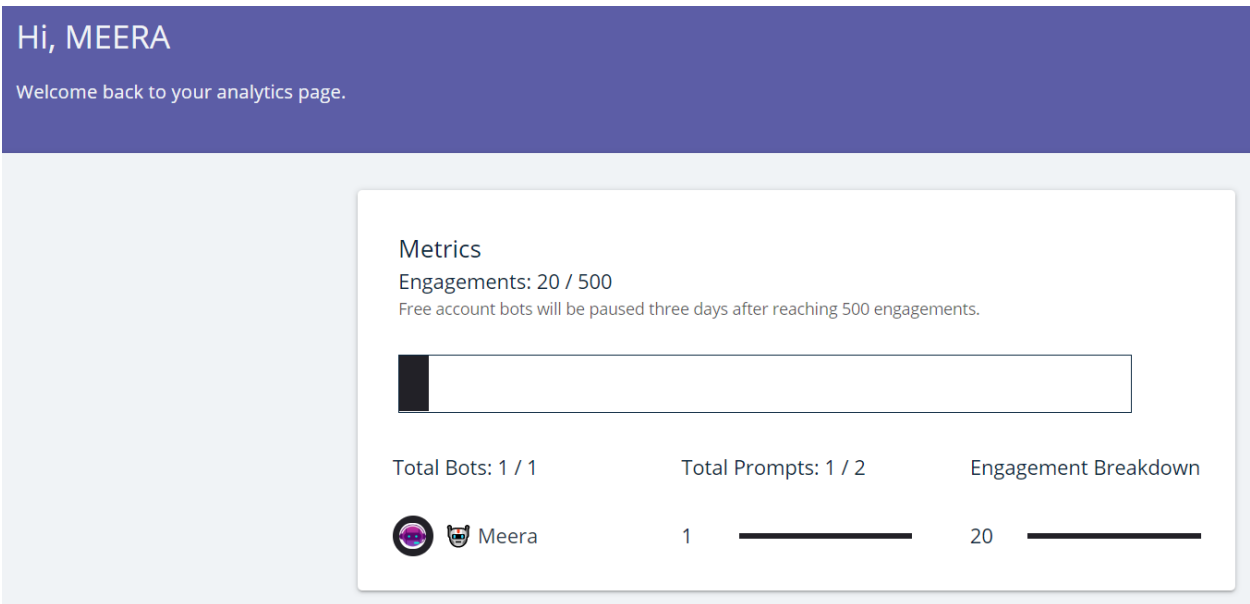


Figure 24 : BotCopy Interaction Analytics Summary

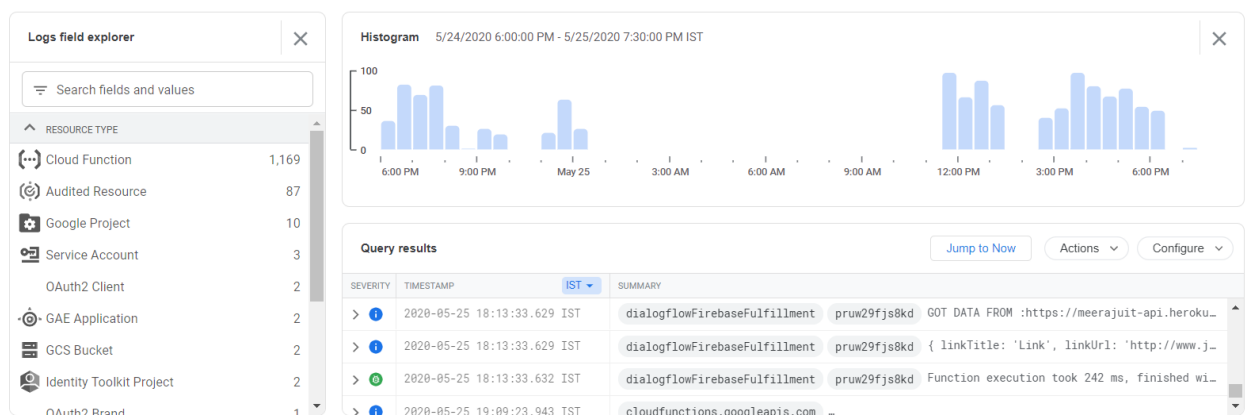
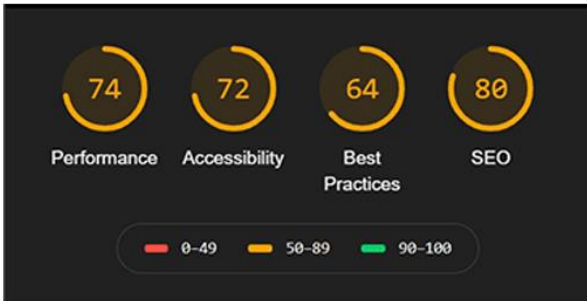


Figure 25 : Google cloud function Logs

In order to measure the site performance, we used the 'Lighthouse' feature of the popular internet browser, Google Chrome. It executes a performance audit of the website against pre set web metrics. The results indicate an average performance for our web service.

Performance Index before the bot was implemented on the website



Performance Index after the bot was implemented on the website

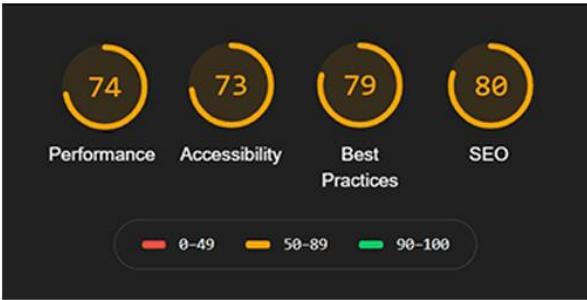


Figure 26 : Bot Copy Performance Audit Summary by Lighthouse

The metrics taken into consideration by Lighthouse are:

First Contentful Paint : It represents the time when the first text or image is rendered on the screen.

Speed Index : It commensurates how quickly the elements of the web page are populated.

Time to Interactive : It shows the time taken by the page to become fully interactive.

First Meaningful Paint : It measures the time when some primary elements of the web page are rendered.

First CPU Idle : It represents the time when for the first time the main thread of the page is too full to handle any further input.

Maximum Potential First Input Delay : It represents the time duration of the longest task performed by the web page.

Chapter 5– Conclusion

5.1 Project Outcome

We were successful in implementing a test chatbot that uses data from our own colleges websites and then delivers the relative content successfully, while keeping track of all the interactions.

The figure below shows the dialogflow console of our JUIT BOT. We see a list of intents and actions that are structured to be delivered to the user in a structured way.

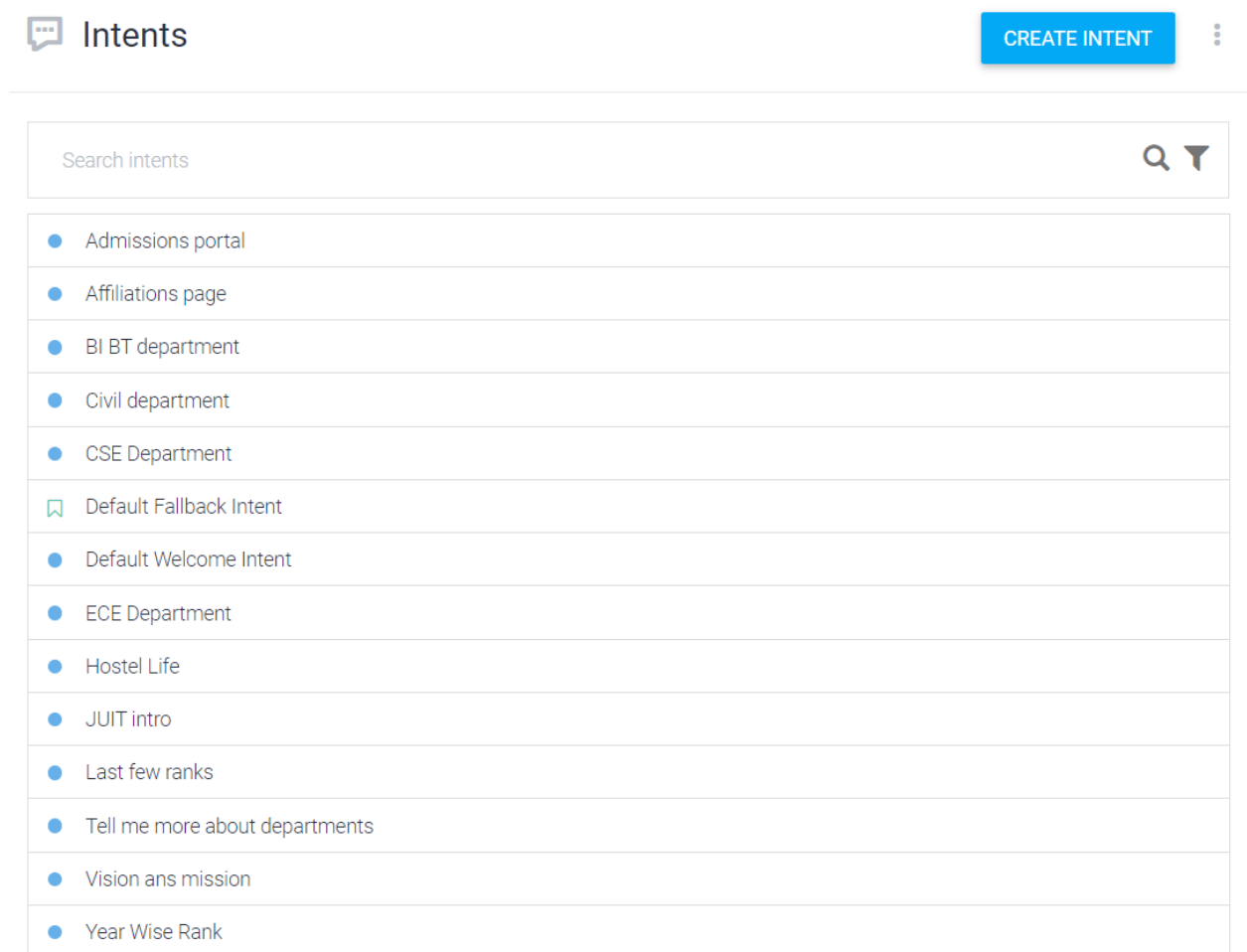


Figure 27 : Dialogflow consol

The following shows our successful implementation of Bot Copy agent on a snapshot of our university website.

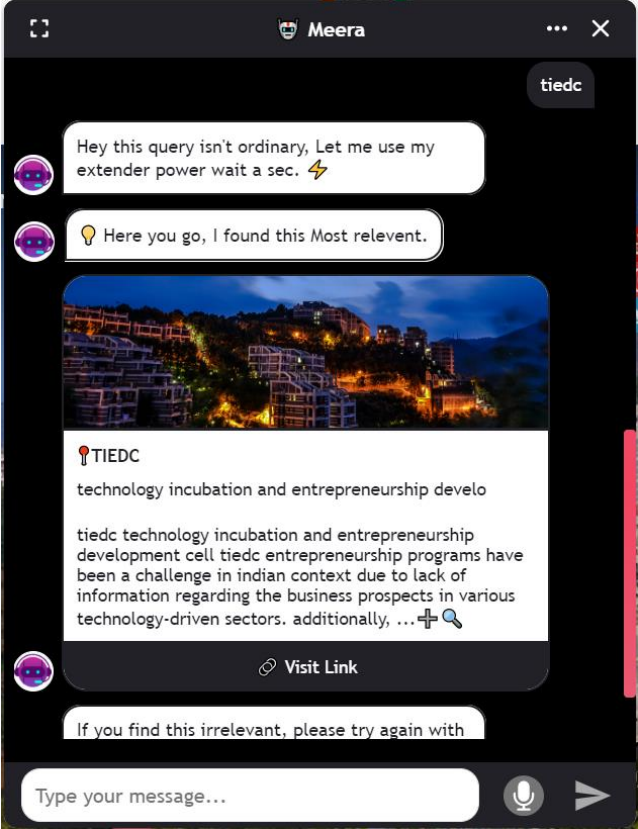


Figure 28 : Bot implementation of university website

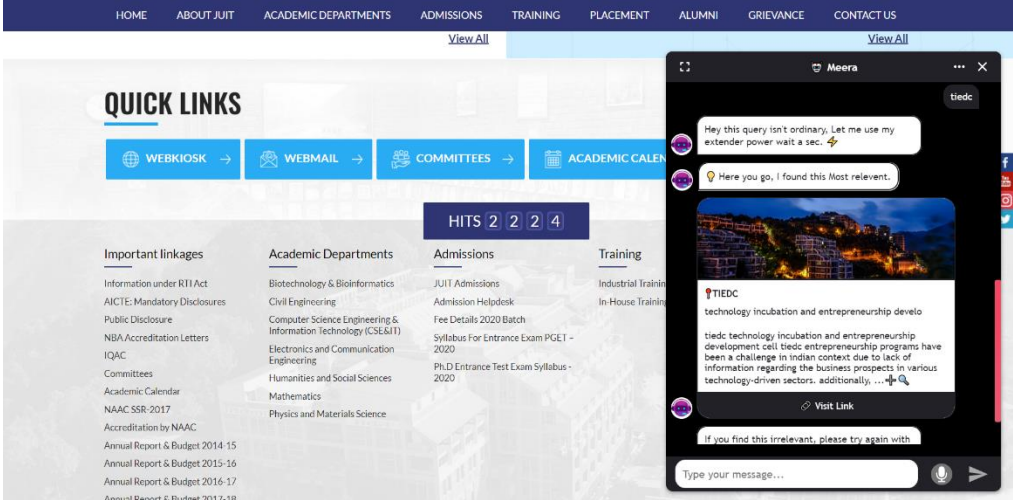


Figure 29 : Bot implementation of university website (View 2)

5.2 Future Scope

We have made a basic implementation of the idea. This serves as a starting point and direction for further development. There is a lot of scope for future development of work. We are planning to develop a single knowledge store based on which the chat bot can cater to the queries of users about many universities and colleges. It will aggregate the information about many universities in a similar fashion as implemented in this project.

The process we use right now to implement a prototype of the chatbot for our own university is an outsource free service that integrates all the technologies so that we can hassle free focus on the content delivery. Right now, we use the BOTCOPY free tier service to implement our chatbot. Our most recent improvement scope is to create a tool that replaces the BOTCOPY service and still renders our project to function in the same way.

Another area of future development can be a self-made and performance efficient user input handling, intent matching and response generation mechanism. This can help bring down the operational cost of the system over a long period of time.

5.3 Conclusion

The first implementation of the chatbot was successful. The bot API is hosted on heroku server and the implementation of the bot on a copy of the university website is hosted on github pages. The performance analytics can be obtained from the Google DialogFlow dashboard.

References

- [1] <https://kevincurran.org/dissertations/2018%20Thesis%20Dana%20Doherty%20-%20Chatbots.pdf>
- [2] Watson, H. J. (2017). 'Preparing for the Cognitive Generation of Decision Support', MIS Quarterly Executive, 16(3), pp. [Online]. Available at: <http://www.misqe.org/ojs2/index.php/misqe/article/viewFile/705/465>
- [3] Duijst, D. (2017). 'Can we Improve the User Experience of Chatbots with Personalisation?', University Of Amsterdam, (), pp. 3 [Online]. Available at: 10.13140/RG.2.2.36112.92165
- [4] Watson, A. (2010). How to succeed with NLP: Go from good to great at work using the power of neuro-linguistic programming. Oxford: Capstone
- [5] KingING, W. B. (2017). Year of the chatbot: Credit unions gearing up for artificial intelligence. Credit Union Journal, 21(4), 18-18 .
- [6] Dole, A., Sansare, H., Harekar, R. and Athalye, S. (2015). [online] www.ijettcs.org. Available at: [http://www.ijettca.org/Volume4Issue5\(2\)/IJETTCS-2015-10-09-16.pdf](http://www.ijettca.org/Volume4Issue5(2)/IJETTCS-2015-10-09-16.pdf)
- [7] <https://drive.google.com/file/d/0B-tCvLzyt01FYlQ2dVRBWetTNkE/view>
- [8] <https://drive.google.com/file/d/0B-tCvLzyt01FblRwc01TVlY0ZUU/view>
- [9] <https://medium.com/swlh/how-to-build-a-chatbot-with-dialog-flow-chapter-1-introduction-ab880c3428b5>
- [10] https://miro.medium.com/max/1200/0*yBKnMaAHhDCfMepU.
- [11] https://miro.medium.com/max/941/1*tZKE2D6wLbgOdWMuNQ4CiQ.png
- [12] <https://www.thestartupinc.com/startup/botcopy/>

- [13] https://www.google.com/imgres?imgurl=https%3A%2F%2Fmiro.medium.com%2Fmax%2F1024%2F1*e8v1xC0NTgoduh_ei9F7Pw.png&imgrefurl=https%3A%2F%2Fchatbotslife.com%2Fchatbots-are-the-future-of-marketing-31fd285f37d9&docid=PWHf-7heN8lsDM&tbnid=1WMBRrXcrt4uzM%3A&vet=10ahUKEwiSI7mm-ZLmAhVXSX0KHeEeACYQMwh8KAIwAg..i&w=1024&h=573&bih=754&biw=1536&q=chatbots%20&ved=0ahUKEwiSI7mm-ZLmAhVXSX0KHeEeACYQMwh8KAIwAg&iact=mrc&uact=8
- [14] <https://blog.vsoftconsulting.com/hs-fs/hubfs/chatbot%20architecture.png?width=1683&name=chatbot%20architecture.png>
- [15] <https://www.attivio.com/blog/post/what-natural-language-understanding>
- [16] <https://blog.vsoftconsulting.com/blog/understanding-the-architecture-of-conversational-chatbot>
- [17] https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- [18] <https://plato.stanford.edu/entries/artificial-intelligence/#StroVersWeakAI>
- [19] <https://www.britannica.com/technology/artificial-intelligence>
- [20] <https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1>
- [21] <https://www.geeksforgeeks.org/nlp-part-of-speech-default-tagging/>

ORIGINALITY REPORT

13%

SIMILARITY INDEX

12%

INTERNET SOURCES

3%

PUBLICATIONS

11%

STUDENT PAPERS

PRIMARY SOURCES

1	www.scribd.com Internet Source	6%
2	www.thestartupinc.com Internet Source	1%
3	www.geeksforgeeks.org Internet Source	1%
4	Submitted to University of Ulster Student Paper	1%
5	www.guru99.com Internet Source	1%
6	Submitted to VIT University Student Paper	1%
7	Dana Doherty, Kevin Curran. "Chatbots for online banking services", Web Intelligence, 2019 Publication	1%
8	Submitted to University of Wisconsin System Student Paper	1%
9	www.ng-conf.org	

MEERA 26 05 20

by Lavish Kumar (161022) Prashasy Ashok (161325)

Submission date: 26-May-2020 04:17PM (UTC+0530)

Submission ID: 1332082244

File name: MEERA_Report_Group_39.pdf (2.43M)

Word count: 9488

Character count: 50925

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 19-07-2020

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

Name: Prashasy Ashok Department: CSE & IT Enrolment No 161325

Contact No. +91 72756 97998 E-mail. prashasyashok@gmail.com

Name of the Supervisor: Dr. Ravindara Bhatt

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): MEERA :
MULTIFUNCTIONAL EDUCATIONAL EXPERT IN REAL-TIME ASSISTANCE

UNDERTAKING

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

Complete Thesis/Report Pages Detail:

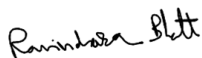
- Total No. of Pages = 59
- Total No. of Preliminary pages = 7
- Total No. of pages accommodate bibliography/references = 2



(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at¹³.....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.



(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

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

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

Checked by
Name & Signature

Librarian

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

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 19-07-2020

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

Name: Lavish Kumar Jangir Department: CSE & IT Enrolment No 161022

Contact No. +91 78918 87989 E-mail. lavishkumar12@gmail.com

Name of the Supervisor: Dr. Ravindara Bhatt

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): MEERA :
MULTIFUNCTIONAL EDUCATIONAL EXPERT IN REAL-TIME ASSISTANCE

UNDERTAKING

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

Complete Thesis/Report Pages Detail:

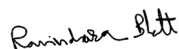
- Total No. of Pages = 59
- Total No. of Preliminary pages = 7
- Total No. of pages accommodate bibliography/references = 2



(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at¹³.....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.



(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

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

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

Checked by
Name & Signature

Librarian

.....

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