TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A

PROJECT REPORT

ON

CHATBOT FOR PULCHOWK CAMPUS

SUBMITTED BY:

BEENAY BASNET (PUL075BCT021)

DILIP KR. JAYSWAL (PUL075BCT033)

DIMPLE SARAOGI (PUL075BCT034)

EROSHA PAUDEL (PUL075BCT035)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

April, 2023

Page of Approval

TRIBHUVAN UNIVER
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certifies that they have read and recommended to the Institute of Engineering for acceptance of a project report entitled **"Chatbot for Pulchowk Campus"** submitted by **Beenay Basnet**, **Dilip Kr. Jayswal**, **Dimple Saraogi**, **Erosha Paudel** in partial fulfillment of the requirements for the Bachelor's degree in Electronics and Computer Engineering.

.....
Supervisor
Prof. Dr. Ram Krishna Maharjan
Assistant Professor
Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

.....
Internal examiner
Nischal Acharya
Assistant Professor
Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

.....
External examiner
Assistant Professor
Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

Date of approval:

Copyright

The author has agreed that the Library, Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purposes may be granted by the supervisors who supervised the project work recorded herein or, in their absence, by the Head of the Department wherein the project report was done. It is understood that the recognition will be given to the author of this report and to the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering in any use of the material of this project report. Copying or publication or the other use of this report for financial gain without approval of to the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering and author's written permission is prohibited.

Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to:

Head
Department of Electronics and Computer Engineering
Pulchowk Campus, Institute of Engineering, TU
Lalitpur,Nepal.

Acknowledgments

We would like to express our deepest gratitude and appreciation to our respected HoD and Supervisor ***Prof. Dr. Ram Krishna Maharjan*** sir, who involved in the development of the chatbot for Pulchowk Campus using Dialogflow.

First and foremost, we would like to thank the Department of Electronics and Computer Engineering, faculty members and staff of Pulchowk Campus who provided invaluable insights and guidance throughout the development process. Their input and feedback helped ensure that the chatbot was tailored to the specific needs of the campus and its students.

We would also like to extend our thanks to the development team who worked tirelessly to build and refine the chatbot. Their dedication, expertise, and creativity were instrumental in bringing the project to fruition.

Furthermore, we would like to acknowledge the contribution of the Dialogflow platform, which provided the foundation for the chatbot's natural language processing and understanding capabilities. Its advanced features and integration capabilities enabled us to build a chatbot that was both intelligent and user-friendly.

Last but not least, we would like to express our appreciation to the students of Pulchowk Campus for their participation and feedback during the testing phase of the chatbot. Their input was invaluable in ensuring that the chatbot met their needs and expectations.

In conclusion, we are grateful for the collaborative effort and support of all involved in the development of the chatbot for Pulchowk Campus using Dialogflow. We believe that the chatbot will be a valuable resource for the campus community and look forward to its continued success.

Abstract

Our project focuses on creating a chatbot for students to use to easily get answers to their questions from the college website. A chatbot is a program that can engage in real conversations with textual methods. Using artificial intelligence (AI), chatbots can simulate human conversations. There are two categories of chatbots. One category is command-based chatbots, where the chatbots are based on a database of responses and heuristics. The user must be very specific when asking the questions so that the bot can answer. Therefore, these bots can answer a limited set of questions and cannot perform functions outside of code. The other category is based on AI-based chatbots or automatic apprenticeship algorithms, these robots can answer ambiguous questions, which means that the user does not have the right pose questions. Therefore, these bots create responses to user queries using natural language processing (NLP).

AI powered chatbots are driven by the need for traditional websites to provide a chat feature where a bot is needed to be able to chat with users and resolve queries. Where a live agent can only handle two or three trades at a time, chatbots can trade without an upper limit, which really increases trades. Also, if a school or business gets a lot of inquiries, having a chatbot on a website takes the burden off the support team. Having a chatbot clearly improves response rate compared to a human support team. Also, since millennials prefer live chats to phone calls, they find a chatbot, which provides an engaging and highly interactive marketing platform. Also, a chatbot can automate repetitive tasks. There may be scenarios where a business or school receives the same queries multiple times during a day and the support team has to respond to each query repeatedly. Finally, the most important benefit of having a chatbot is that it is available 24/7. No matter what time it is, a user can solve a query. All these benefits of a chatbot are the motivation to implement a College Inquiry Chatbot. The college query system will provide the answer by summarizing the query and then generating the answers, it also provides selective information about what the user wants. A college system will dispense with all answers related to areas such as admission, exam cell, notice board, attendance, placement cell, and other miscellaneous areas.

Contents

List of Figures

1. Introduction

Task automation has grown rapidly due to an increase in technological resources and tools. One of these areas is automation of customer service. Many companies now have automated response chatbots to provide extension Customer service. The main intention of using chatbots is to provide support around the clock 24/7. They are highly accessible, customizable and provide instant answers to Username. They can be customized to perform actions based on the inputs given to it .

As we all know, chatbots measure a very current approach to communication between the user, the textual interface of the system. We tend to use this system to set up a college website for additional business communication between the student and the administration. Students use the college website's chatbot to ask questions instead of going directly to work or the government for information. Our project is to create a chatbot for a university website that will undoubtedly have a great impact as it could change the approach of interaction with students.

1.1 Problem Motivation

AI-powered chatbots are driven by the need for traditional websites to provide a chat facility where a bot is needed to be able to chat with the user and resolve queries. When Live agent can only handle two or three trades at a time, chatbots can work without an upper limit that actually increases operations. Also, if a school or business is getting a lot of inquiries, having a chatbot on a website takes the burden off of the support team. Having a chatbot definitely improves response rate compared to a human support team. Also, since millennials prefer live chats to a phone call, they find a chatbot that offers a highly interactive and engaging marketing platform. Also, a chatbot can automate repetitive tasks. There may be scenarios where a company or the school receives the same requests several times a day and the support team must respond to them each request repeatedly. Finally, the most important benefit of having a chatbot is that it is available 24/7. No matter what time it is, a user can solve a query. All these benefits of a chatbot are the motivation to implement a College Inquiry Chatbot.

So summing up the points are as follows:

To make traditional websites more interactive and help resolve queries and chat with the user by providing a chat area, AI-enabled chatbots have been promoted.

Chatbots can handle more than two or three operations at a time, unlike the live agent

on traditional websites. It is a blessing to increase operations by reducing your load.

This relieves the support team and improves the response rate compared to a human support team.

1.2 Project Objective

a. To provide the most convenient way to efficiently search for the most relevant queries from students.

b. To provide support that can help the user with the most basic queries 24/7.

c. Ease of work to find distinctive features on our college website.

d. Makes it easier for parents and students by providing additional technical support.

e. It proves to be a time saving app for students as well as teaching and non-teaching.

f. Reduce manual effort

1.3 Scope of Project

The main objective of this project is to create a chat-bot for the college website that can be used by students and other people to get their queries in a more efficient and easy way.

• This will give a quick response for courses and other details like student and faculty details.

• This will calculate the fee based on the student's contribution, answer frequently asked questions, and provide addresses, contacts and location.

• This will provide the link to academic calendar and event information from various departments.

• The chat-bot will make conversations more friendly and efficient.

1.4 Background

Chatbots like ELIZA and PARRY were the first attempts to create programs that could, at least temporarily, trick a real person into thinking they were having a conversation with someone else. PARRY's effectiveness was compared to that of the early 1970s using a version of the Turing test; the testers only correctly identified a human versus a chatbot to a level consistent with random guesses. Chatbots have come a long way since then. Developers build modern chatbots using AI technologies, including deep learning algorithms, NLP, and machine learning (ML). These chatbots require massive amounts of data. The more an end user interacts with the bot, the better it predicts the right responses. There are various chatbots like Amazon Shopping App, Alexa, Bank of America (Erica bot) and CNN news bot. To understand the requirement for a chatbot, consider an example of the Amazon Shopping app. In this app, when a customer purchases an item, they have no information on how to return it. To obtain this information, the customer must call and wait a long

time to speak with the customer representative. However, this whole process is tedious for a client. Therefore, Amazon created a chatbot to answer simple customer queries. The use of chatbots is increasing in business and consumer markets. As chatbots improve, consumers have less reason to argue when interacting with them. Between cutting-edge technology and a social transition to more passive text-based communication, chatbots are helping to fill a niche previously occupied by phone calls.

1.5 Chatbot

Instead of providing direct contact with a live agent, we use Chat-bot as an application to conduct an online conversation in the form of text or text to speech. These are designed to mimic the way a person behaves as a chat partner. These systems often require continual tweaking and testing to pass a standard Turing field test.

A chat bot (by many names, including talkbots, bot, IM bot, chatterbot, interactive agent, or artificial conversational entity) is an artificial intelligence or computer program that provides a means of conversation through audio or text.

Programs like chat-bots are designed to mimic how a person behaves as a conversation partner, or equivalent to passing the Turing test. Chatbots are used for a variety of purposes, including customer service or information acquisition in chat systems. Some chatterbots analyze the keywords in the input and then extract the response with the most relevant keywords, or similar keyword patterns, from the database, and some use native language processing systems.

In 1994, Michael Mauldin coined the term "Chatterbot" to describe chat systems. Today, assistants like Google Assistant and Amazon Alexa make content or information available through messaging apps like WhatsApp, Facebook Messenger, WeChat, Telegram, or through apps and websites of other individual organizations. The categorization of chatbots is as follows:- Chat Commerce (E-Commerce for Chat), Analytics, Communication, Customer Support, Construction, Development Tools, Entertainment, Finance, Food, Sports, Health, HR, Marketing, News, Staff, Manufacturing, Shopping, Social, Sports , education, tourism and resources.

If the bot can pass the test, that is, the Turing test, the person or user may not even know that they are in contact with an artificial intelligence or a computer program.

Chatbot assistants are increasingly being used to handle simple tasks, freeing human agents from the need to focus on high-quality service or marketing cases. The integration of chat-bots generates cost savings because we know that employees are more expensive.

Additionally, it enables organizations to provide a level of customer service when live agents are not available.

1.6 Dialogflow

Dialogflow is a cloud-based natural language processing (NLP) platform developed by Google. It enables developers to build conversational interfaces for a variety of applications, such as chatbots, voice assistants, and messaging platforms. Dialogflow uses machine learning to understand user input and generate appropriate responses.

Dialogflow offers a range of features to enable developers to build conversational applications quickly and easily. Some of the key features of Dialogflow include:

Intent Detection: Dialogflow uses machine learning algorithms to understand the user's intent based on the input provided. The platform identifies the relevant intent from a predefined set of intents defined by the developer.

Entity Recognition: Dialogflow uses entity recognition to extract relevant information from the user input. This enables the chatbot to provide more personalized responses and improve the overall user experience.

Dialog Management: Dialogflow provides a pre-defined dialog flow to determine the appropriate response based on the user input and the context of the conversation.

Multilingual Support: Dialogflow supports more than 20 languages, making it a powerful tool for building multilingual conversational applications.

Integration: Dialogflow can be integrated with a variety of platforms, such as Facebook Messenger, Slack, and Google Assistant, to enable users to interact with the chatbot through their preferred channel.

Analytics: Dialogflow provides analytics tools to help developers understand how users are interacting with the chatbot, what questions they are asking, and how successful the chatbot is in meeting user needs.

Dialogflow offers both a standard and an enterprise edition, with the enterprise edition providing additional features and support for larger-scale applications. Overall, Dialogflow is a powerful tool for building conversational applications that can understand natural language and provide personalized responses, making it a popular choice for developers building chatbots and voice assistants.

1.6.1 Installing Dialogflow

- To install the Dialogflow CLI, the user will need to have Node.js and npm installed on their computer. Once they have Node.js and npm installed, they should open a terminal window and run the command **"npm install -g @google-cloud/dialogflow-cli".**

- Next, the user should create a Dialogflow agent by going to the Dialogflow website and creating a new agent. They should give the agent a name and select a default language

and time zone.

- In order to use the Dialogflow CLI, the user will need to authenticate with Google Cloud Platform. They can do this by running the command "gcloud auth application-default login" in the terminal window. This will open a browser window where they can sign in to their Google account and grant permissions to the Dialogflow CLI.

- To deploy the agent to Google Cloud Platform, the user will need to create a service account key. They can go to the Google Cloud Console, select their project, and navigate to the IAM and admin ¿ Service accounts page. They should click "Create Service Account", give it a name, and grant it the "Dialogflow API Admin" role. Then, they should click "Create Key" and download the JSON file.

- In the terminal window, the user should set the following environment variables:

  export GoogleApplicationCredentials="[PathToYourServiceAccountJsonFile]"
  export ProjectId="[YourProjectId]"

- They should replace "**[PathToYourServiceAccountJsonFile]"** with the path to the JSON file they downloaded earlier, and replace "[YourProjectId]" with their Google Cloud Platform project ID.

- Finally, to deploy the agent to Google Cloud Platform, the user should run the command "dialogflow deploy". This will upload the agent to Google Cloud Platform and make it available for use. The user can now start building their chatbot or voice assistant using Dialogflow.

1.6.2 Dialogflow Commands

Dialogflow provides a range of commands that developers can use to build and manage their conversational applications. Here are some of the key commands in Dialogflow:

Create Agent: This command creates a new agent in Dialogflow. An agent is the core component of a conversational application and contains the NLP models, intents, entities, and other components that make up the application.

Create Intent: This command creates a new intent within an agent. An intent represents a specific action that the user wants to perform, such as making a reservation or ordering food.

Create Entity: This command creates a new entity within an agent. An entity represents a specific object or concept that the user may refer to in their input, such as a date or location.

Fulfillment: This command enables developers to add custom code or scripts to their conversational application to provide more advanced functionality, such as accessing external APIs or databases.

Testing: Dialogflow provides a range of testing commands that developers can use to test their conversational application and ensure that it is working as expected.

Export and Import: These commands enable developers to export and import their conversational application to and from Dialogflow. This can be useful for backing up the application or transferring it to a different environment.

Training: Dialogflow uses machine learning algorithms to train the NLP models that power the conversational application. The training commands enable developers to monitor and optimize the training process to improve the accuracy and performance of the application.

1.6.3 Actions Supported:

Dialogflow supports a wide range of actions that developers can use to build complex conversational experiences.

These actions include:

- Text Response - The chatbot responds with text.
- Image Response - The chatbot responds with an image.
- Audio Response - The chatbot responds with an audio clip.
- Video Response - The chatbot responds with a video clip.
- Webhook Response - The chatbot makes a request to an external service and responds with the result.
- Custom Payload Response - The chatbot responds with a custom payload that can be processed by the client application.

1.6.4 Dialogflow Architecture

The architecture of a Dialogflow-based conversational application typically includes the following components:

User Interface: The user interface is the part of the application that the user interacts with, such as a chatbot on a website or a voice assistant on a smart speaker. The user interface sends the user's input to Dialogflow for processing.

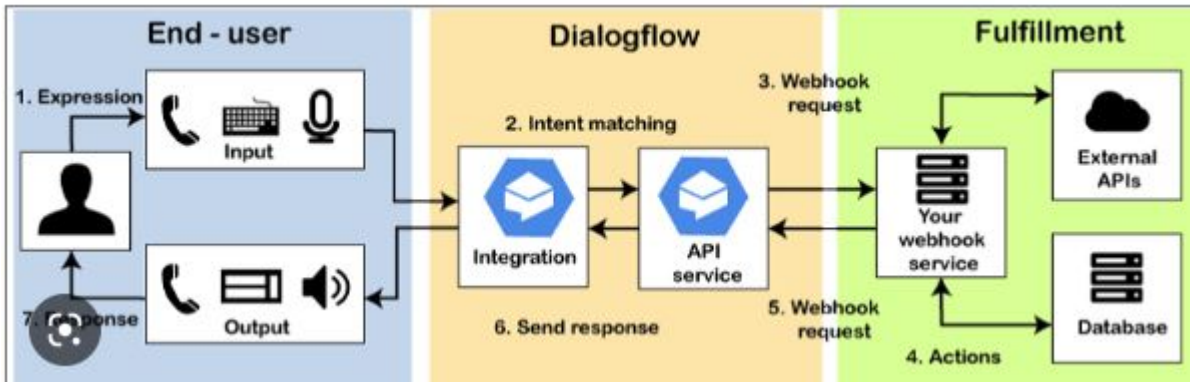

Figure 1.1: Dialogflow architecture

Dialogflow: Dialogflow is the NLP platform that processes the user's input and generates appropriate responses. Dialogflow uses machine learning algorithms to understand the user's intent and extract relevant information from their input.

Fulfillment: The fulfillment component is responsible for performing actions based on the user's request. For example, if the user asks to book a hotel room, the fulfillment component would handle the booking process and send a confirmation to the user. Fulfillment can be implemented using custom code or scripts.

Backend Services: Backend services are used to perform any tasks required by the fulfillment component, such as accessing a database or API. Backend services are typically implemented using a cloud-based platform, such as Google Cloud Platform or AWS.

Analytics: Analytics tools are used to monitor the performance of the conversational application and gather insights into user behavior. Dialogflow provides built-in analytics tools, but additional tools can also be used, such as Google Analytics or Mixpanel.

The architecture of a Dialogflow-based conversational application can be customized to meet the specific needs of the application. For example, additional components can be added to handle complex tasks or integrate with other systems. Overall, the architecture of a Dialogflow-based application provides a powerful and flexible platform for building conversational interfaces that can understand natural language and provide personalized responses.

1.6.5 Dialogflow Training

Training Dialogflow's Natural Language Understanding (NLU) models is a critical step in building a conversational application that can understand user inputs and provide relevant responses. Here are some steps to train Dialogflow's NLU:

Define Intents: An intent represents a specific action that the user wants to perform, such as ordering food or booking a hotel room. Define a list of intents that your application

will support.

Add Training Phrases: For each intent, add a list of training phrases that users might use to express that intent. Dialogflow uses these training phrases to understand the user's intent and extract relevant information.

Define Entities: Entities represent specific objects or concepts that the user may refer to in their input, such as a date or location. Define a list of entities that your application will support.

Add Entity Values: For each entity, define a list of values that the entity can take. For example, a date entity might have values such as "today," "tomorrow," or "next week."

Enable Contexts: Contexts allow Dialogflow to understand the context of the user's input and provide more accurate responses. Define a list of contexts that your application will support.

Test the Model: Once you have defined intents, training phrases, entities, and contexts, test the model to see how well it performs. Use the Dialogflow console or API to send sample inputs to the model and verify that it understands the user's intent and extracts relevant information correctly.

Optimize the Model: If the model is not performing as expected, optimize it by adjusting the training phrases, entities, and contexts. You can also use Dialogflow's machine learning features to improve the model's accuracy over time.

Training Dialogflow's NLU models is an iterative process that requires ongoing testing and optimization. By following these steps and continuously improving the model, you can build a conversational application that can understand natural language inputs and provide relevant and accurate responses.

1.6.6 Pipeline in Dailogflow

In Dialogflow, a pipeline refers to the flow of user input and the processing steps that are performed on that input to generate a response. The pipeline typically includes the following steps:

User Input: The user inputs a text or voice command into the conversational interface.

Natural Language Understanding (NLU): The user input is processed by Dialogflow's NLU system, which uses machine learning algorithms to extract the user's intent and entities from the input. Dialogflow also considers any context that is present in the conversation to improve the accuracy of its understanding.

Fulfillment: Once the user's intent and entities have been identified, the fulfillment component of the pipeline takes over. This component is responsible for processing the user's request and generating a response. Fulfillment can be implemented using custom code or

pre-built integrations with external services.

Response Generation: The response generation component takes the output of the fulfillment component and generates a response that is sent back to the user. The response can be in the form of text, speech, or visual output, depending on the type of conversational interface being used.

Analytics: Analytics tools can be used to monitor the performance of the pipeline and gather insights into user behavior. Dialogflow provides built-in analytics tools, but additional tools can also be used, such as Google Analytics or Mixpanel.

The pipeline in Dialogflow is highly customizable and can be adapted to meet the specific needs of the conversational application. For example, additional processing steps can be added to handle complex tasks or integrate with other systems. Overall, the pipeline in Dialogflow provides a powerful and flexible platform for building conversational interfaces that can understand natural language and provide personalized responses.

2. Literature Review

2.1 Related work

1. "Campus Buddy" by Krea University: This chatbot helps students navigate campus life by answering questions about events, facilities, and academic programs. It uses Dialogflow to interpret user input and provides responses through a web-based interface.

2. "UniBot" by the University of Edinburgh: This chatbot helps prospective students learn about the university's programs, admissions process, and campus life. It uses Dialogflow to handle natural language input and provides responses through Facebook Messenger.

3. "Guru" by Marist College: This chatbot provides personalized academic advice to students, such as course recommendations and scheduling assistance. It uses Dialogflow to understand user input and provides responses through a web-based interface.

4. "Alfred" by the University of Melbourne: This chatbot provides support to students with disabilities, such as scheduling accommodations and providing campus navigation assistance. It uses Dialogflow to interpret user input and provides responses through a custom-built app.

5. "Buddy" by the University of Wisconsin-Milwaukee: This chatbot helps students with academic advising and registration, as well as providing information about campus resources and events. It uses Dialogflow to process user input and provides responses through a web-based interface.

These are just a few examples of college chatbots using Dialogflow. Each project has its own unique features and functionality, but they all demonstrate the power of natural language processing in improving the student experience.

2.2 Summary of RELEVANT Papers

Introduction to Microsoft Bot, and Google Dialogflow, Abhishek Singh et al., 2019 discuss how to create a chatbot framework for in-house conversations in natural language and conversational skills. Creating a solution from scratch has the benefits we discussed earlier. However, there are scope of use and scope where it can be easier, faster, and cheaper to use the separation of existing objectives with chat management frameworks to build your chat client.

Using Chatbots to Assist Communication in collaborative networks , Christian Frommert et al., 2018 indicates that the power of social networks has yet to be used. Although systems are interconnected with each other, and create unconditional data exchange conditions, there is a lack of flexibility to use data stock within the network efficiently and easily. Chatbots provide opportunities to meet this challenge. Nowadays, chat rooms are used to improve customer communication and simplify the day-to-day process in the lives of customers. Within collaborative networks, their use and benefits are not yet fully available. This paper examines current chatbot technologies and uses the type of litigation that is used to demonstrate the benefits of chatbots within social (internal) networks by intelligently integrating data across all collaborative networks.

A Survey Paper on Approaches of Natural Language Processing (NLP) Samta Tembhekar et al., 2017 discusses that with the increasing usage of the Web and other text application areas, the demands in both text mining and natural language processing (NLP) have been increasing. Researchers in text mining have hoped that NLP can provide useful improvements to text mining applications of all kinds. The primary goal of NLP is to implement within computers the skill to understand a normal human language or natural language. It is related to the field of computer and human interaction.

Using Chatbots to Assist Communication in collaborative networks , Christian Frommert et al., 2018 indicates that the power of social networks has yet to be used. Although systems are interconnected with each other, and create unconditional data exchange conditions, there is a lack of flexibility to use data stock within the network efficiently and easily. Chatbots provide opportunities to meet this challenge. Nowadays, chat rooms are used to improve customer communication and simplify the day-to-day process in the lives of customers. Within collaborative networks, their use and benefits are not yet fully available. This paper examines current chatbot technologies and uses the type of litigation that is used to demonstrate the benefits of chatbots within social (internal) networks by intelligently integrating data across all collaborative networks.

A Review of the Neural History of Natural Language Processing, Kamper H

and Ruder 2018 discusses that intelligent chatbots are popular issues in the application fields of robot systems and natural language processing. As the development of natural language processing and neural network algorithms, the application of artificial intelligence is increasing in Chatbot systems, which are typically used in dialog systems for various practical purposes including customer services or information acquisition. This paper designs the functional framework and introduces the principle of Rasa NLU for the chatbot system, then it integrates Rasa NLU and neural network (NN) methods and implements the system based on entity extraction after intent recognition. With the experimental comparison and validation, our developed system can realize automatic learning and answering the collected questions about finance. The system analysis of two methods also validate that Rasa NLU outperforms NN in accuracy for a single experiment, but NN has better integrity to classify entities from segmented words.

An Intelligent Chatbot System Based on Entity Extraction Using RASA NLU and Neural Network , Anran Jaio, 2020 showed the approaches to perceiving the accuracy and integration of a business or sentence, and also ensured an improved system in a realistic way. In the future, this system will be further enhanced by the accuracy of business extensions for longer sentences and more complex organizations. The methods in this paper are used for academic research and not for commercial purposes.

3. Project Architecture

The chatbot is built using Dialogflow, which is a natural language processing (NLP) platform that enables developers to build conversational interfaces for a wide range of applications.

The chatbot architecture is designed to enable users to ask questions in natural language and receive accurate and relevant responses in real-time. The architecture comprises several components, including the Dialogflow agent, fulfillment server, and third-party integrations.

The Dialogflow agent is responsible for processing user input and generating responses. It uses NLU algorithms to understand the user's intent and extract key entities from the input. The Dialogflow agent is trained on a corpus of college-related data, such as course information, admission requirements, and campus facilities, to enable it to provide accurate and relevant responses to user queries.

The fulfillment server is responsible for processing user requests and generating responses based on the information provided by the Dialogflow agent. It acts as an intermediary between the Dialogflow agent and external services, such as the college's database, web services, or third-party APIs. The fulfillment server is implemented using custom code that can be hosted on a cloud platform, such as Google Cloud or Amazon Web Services.

Third-party integrations enable the chatbot to connect with external platforms, such as messaging apps, social media platforms, and voice assistants. This enables users to access the chatbot from multiple channels and devices.

The chatbot architecture is designed to be scalable, flexible, and secure. It can be easily customized to meet the specific needs of the college, and can be updated over time to incorporate new features and functionality. By leveraging Dialogflow's NLP capabilities and the fulfillment server's ability to integrate with external services, the chatbot provides a user-friendly and efficient way for students and visitors to interact with the college website and find the information they need.

3.1 Working of Dialogflow

Dialogflow is a natural language processing (NLP) platform that enables developers to build conversational interfaces for a wide range of applications, such as chatbots, voice assistants, and messaging apps. The platform works by processing user input and generating responses based on the user's intent and context.

Figure 3.1: Working of Dialogflow

Here's an overview of the working of Dialogflow:

- User Input: The user provides input to the Dialogflow agent, such as a text message or voice command. The input can be in natural language and can include multiple intents or entities.
- Natural Language Understanding (NLU): The Dialogflow agent processes the user input using NLU algorithms to understand the user's intent and extract key entities from the input. The NLU algorithms are trained on a corpus of data, such as FAQs, product descriptions, and customer support queries, to enable the agent to provide accurate and relevant responses to user queries.
- Intent Recognition: The Dialogflow agent identifies the user's intent based on the input provided and maps it to a specific intent in the agent's knowledge base. The intent defines the action the agent should take in response to the user's request, such as providing information, scheduling an appointment, or making a reservation.
- Response Generation: The Dialogflow agent generates a response based on the user's intent and context. The response can be in the form of text, voice, or rich media, depending on the platform used to access the agent.

- Fulfillment: The response generated by the Dialogflow agent can be fulfilled using custom code or pre-built integrations with external services, such as databases, web services, or third-party APIs. The fulfillment component is responsible for processing the user's request and generating a response.

- Integration: The Dialogflow agent can be integrated with a wide range of third-party platforms, such as chatbots, voice assistants, and messaging apps. This enables developers to build conversational interfaces that can be accessed from multiple channels and devices.

- Analytics: Dialogflow provides built-in analytics tools that enable developers to monitor the performance of their conversational interface, such as user engagement, retention, and satisfaction. These insights can be used to optimize the conversational interface and improve the user experience.

Overall, the working of Dialogflow is designed to enable developers to build conversational interfaces that can understand natural language and provide personalized responses. By leveraging Dialogflow's NLU capabilities and the fulfillment server's ability to integrate with external services, developers can build chatbots, voice assistants, and other conversational interfaces that can improve customer engagement, reduce costs, and drive business growth.

3.2 Proposed Model

The system we propose will receive the query from the end user, who can be a student or a parent, and the system will process and respond to the query raised. The system will compare the query provided by the user with the existing knowledge base and try to find a suitable answer. The system can also respond to general inquiries from the student or parent. The complete system algorithm proposed by us can be understood with the help of the following flowchart. The user will enter the query first, then the query will be compared against the database. If there is matching data from the query, it will go to the knowledge base, if not, it will go to the predefined answer, therefore, the user will clarify the query back to the chat bot. If there is a result that matches the query, it will respond accordingly. And that's the flow of the chart.

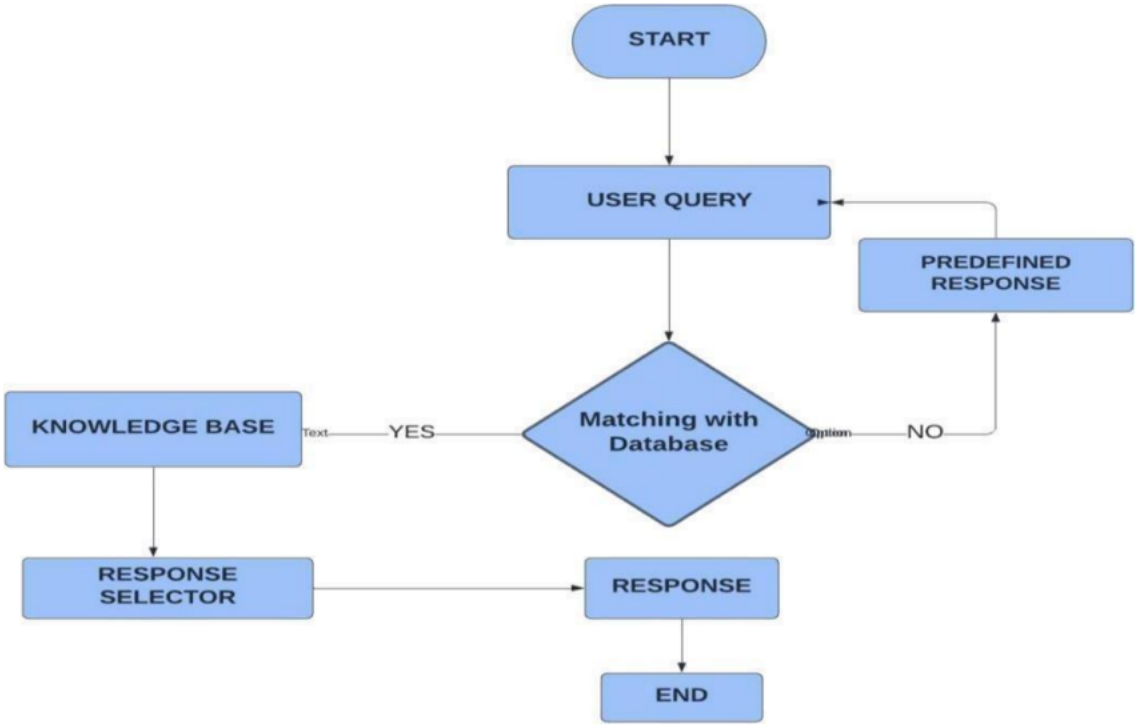

Figure 3.2: Proposed Model

4. System design

4.1 System Architecture

Figure shows the system architecture for the required chatbot system. When the user inputs its query, it is sent to the reasoning component where the reasoning component finds the best match from the knowledge base according to the query. After finding the best match , selection is done and sent back to the reasoning component which responds through the web interface.

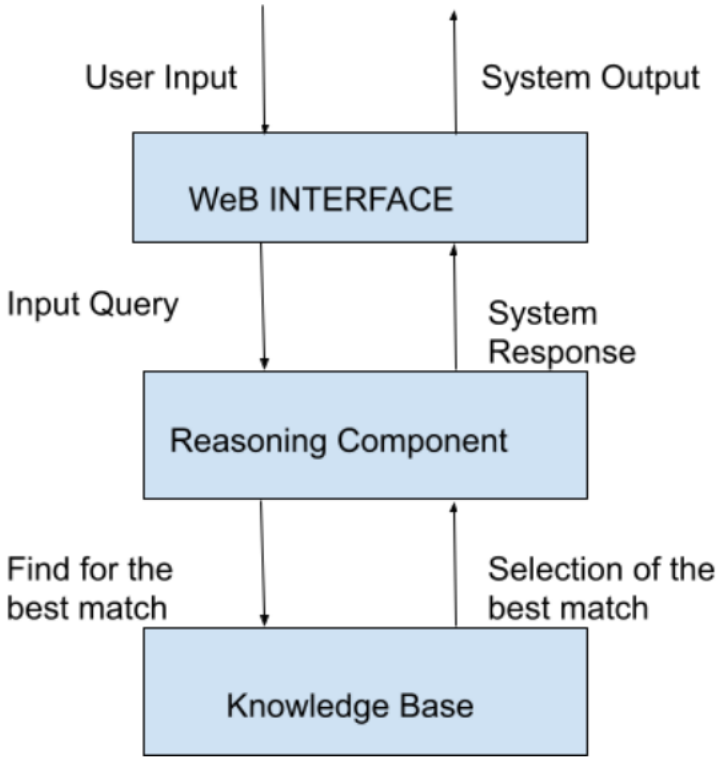

Figure 4.1: System architecture

4.2 Data flow diagram

Figure explains that the user will first type the chatbot query and then extract the query. The query retrieval will be done using Dialogflow-NLU and then the pattern matching will be repeated. With the help of Natural Language Processing (NLP for short, is broadly defined as the automatic manipulation of natural language, such as speech and text, by software), you will go to the knowledge base. If there is matching data from the query, it will go to the knowledge base; otherwise it will go to the default answer. So the user will clarify the query back to the chatbot and then reply back to the user.

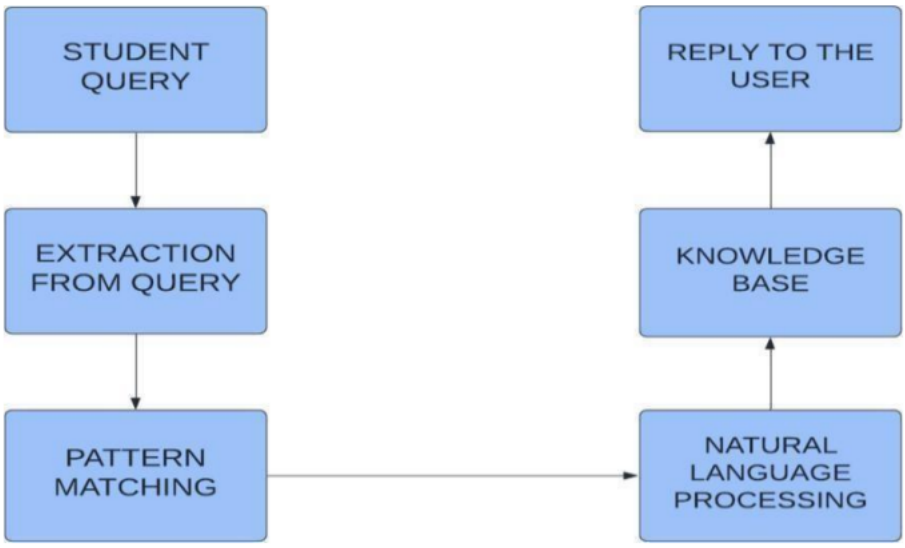

Figure 4.2: Data flow diagram

5. Implementation And Results

5.1 Methodology

The chatbot for Pulchowk Campus built using Dialogflow follows a structured methodology. The first step is to identify the user needs and requirements, followed by designing the conversation flow using intents, entities, and responses. The chatbot is then trained using machine learning algorithms to understand user input and respond appropriately. It is then integrated with the Pulchowk Campus website and other relevant systems. Testing and validation are performed to ensure that the chatbot is functioning correctly and meeting user requirements. Finally, the chatbot is launched and continuously monitored to improve its performance and functionality over time.

1. Define the Problem: First, identify the problem that the chatbot is intended to solve. For example, in the case of Pulchowk Campus, the chatbot could be designed to answer common questions about the campus, such as the location of various buildings, the hours of operation for different facilities, and so on.

2. Define the Use Cases: Next, define the specific use cases that the chatbot will support. This will involve identifying the types of questions that users are likely to ask, as well as the information that the chatbot will need to provide in response.

3. Design the Dialog Flow: With the use cases defined, you can begin designing the dialog flow for the chatbot. This involves mapping out the various paths that the conversation might take, based on the user's input and the chatbot's responses.

4. Develop the Chatbot: Once the dialog flow is defined, you can begin developing the chatbot using Dialogflow. This will involve creating intents, entities, and responses, as well as training the chatbot to recognize different types of user input.

5. Test and Iterate: With the chatbot developed, you can begin testing it to ensure that it works as intended. This will involve simulating different user scenarios and verifying that the chatbot provides accurate and helpful responses. If there are any issues or areas for improvement, you can iterate on the chatbot design and development as necessary.

6. Deploy the Chatbot: Finally, once the chatbot has been thoroughly tested and refined, you can deploy it to your desired platform, such as a website or messaging app. You may also need to integrate the chatbot with other systems or APIs as necessary, such as a database of campus information.

5.2 Software and Hardware Requirements

5.2.1 Hardware Requirements

Dialogflow is a cloud-based service, which means it does not have any specific hardware requirements. Instead, it relies on the hardware resources of the cloud platform it is hosted on. Therefore, as a user of Dialogflow, we do not need to worry about hardware requirements or provisioning any specific hardware resources.

However, one may need a computer or mobile device to access and use the Dialogflow console or to build integrations with Dialogflow APIs. In this case, the minimum hardware requirements for your computer or mobile device will depend on the operating system and browser you are using. Here are the minimum requirements for some common operating systems and browsers:

- **Windows 10:** 1 GHz processor, 1 GB of RAM, and 16 GB of free hard disk space
- **macOS:** Intel-based processor, 2 GB of RAM, and 10 GB of free hard disk space
- **iOS:** iPhone 5S or later, iPad Air or later, or iPod touch 6th generation or later, and the latest version of iOS
- **Android:** Android 5.0 or later and the latest version of the Chrome browser or Web-View

In addition, if we are using Dialogflow in conjunction with other services, such as a database or web server, the hardware requirements will depend on the specific requirements of those services.

5.2.2 Software Requirements

As a team developing a chatbot using Dialogflow, we need to ensure that we have the necessary software requirements to use the platform. These include a modern web browser such as Chrome, Firefox, Safari, or Edge, a stable and reliable internet connection, and a Google account to sign in.

If we plan to use advanced features of Dialogflow, such as integrating with other Google Cloud services or deploying the chatbot to other platforms, we will also need a Google Cloud Platform (GCP) account.

In addition, if we plan to develop custom code or integrations with Dialogflow, we may need an Integrated Development Environment (IDE) or code editor such as Visual Studio Code, as well as the appropriate programming language installed on our systems.

It's also important to note that if we plan to integrate Dialogflow with third-party services or platforms, we may need to meet additional software requirements based on the specific integration. For example, integrating with a messaging platform such as Facebook Messenger or Slack may require us to create accounts on those platforms and install their respective SDKs.

5.3 Assumptions and Dependencies

The project is based on the assumption that the chatbot can answer all university-related queries very efficiently.

No additional dependencies are needed for the development of this project other than the Python packages listed above.

5.4 Implementation Details

5.4.1 Interface Snapshots

The figure depicts a few list of intents . The intents are organized under different categories such as Admissions, Academics, Student Life, and Support. Each intent is designed to understand specific user queries related to the category, allowing the chatbot to provide accurate and relevant information to the user. By organizing the intents under different categories, the chatbot can easily direct users to the right information and help them quickly find the answers they need.

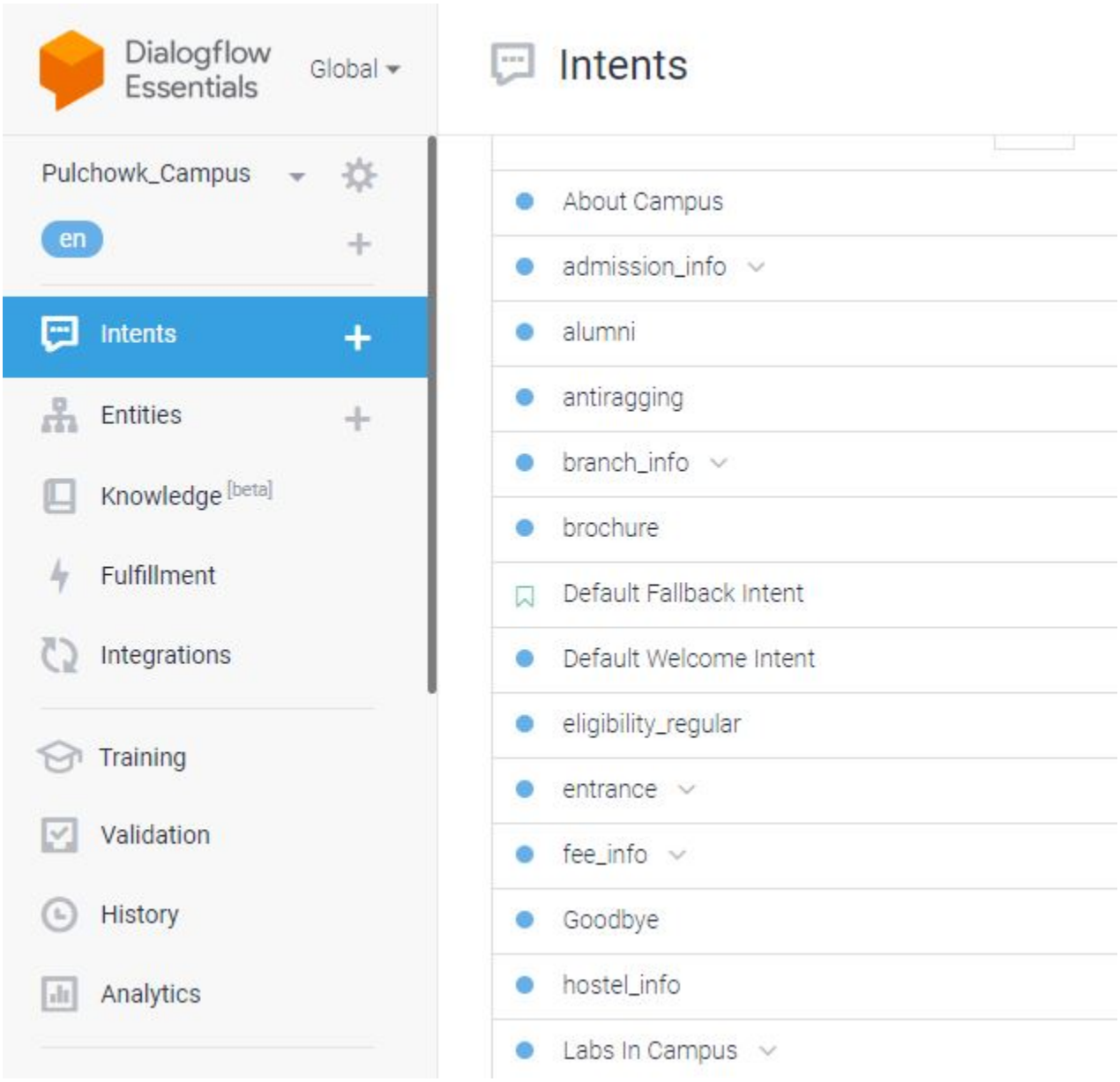

Figure 5.1: Intents

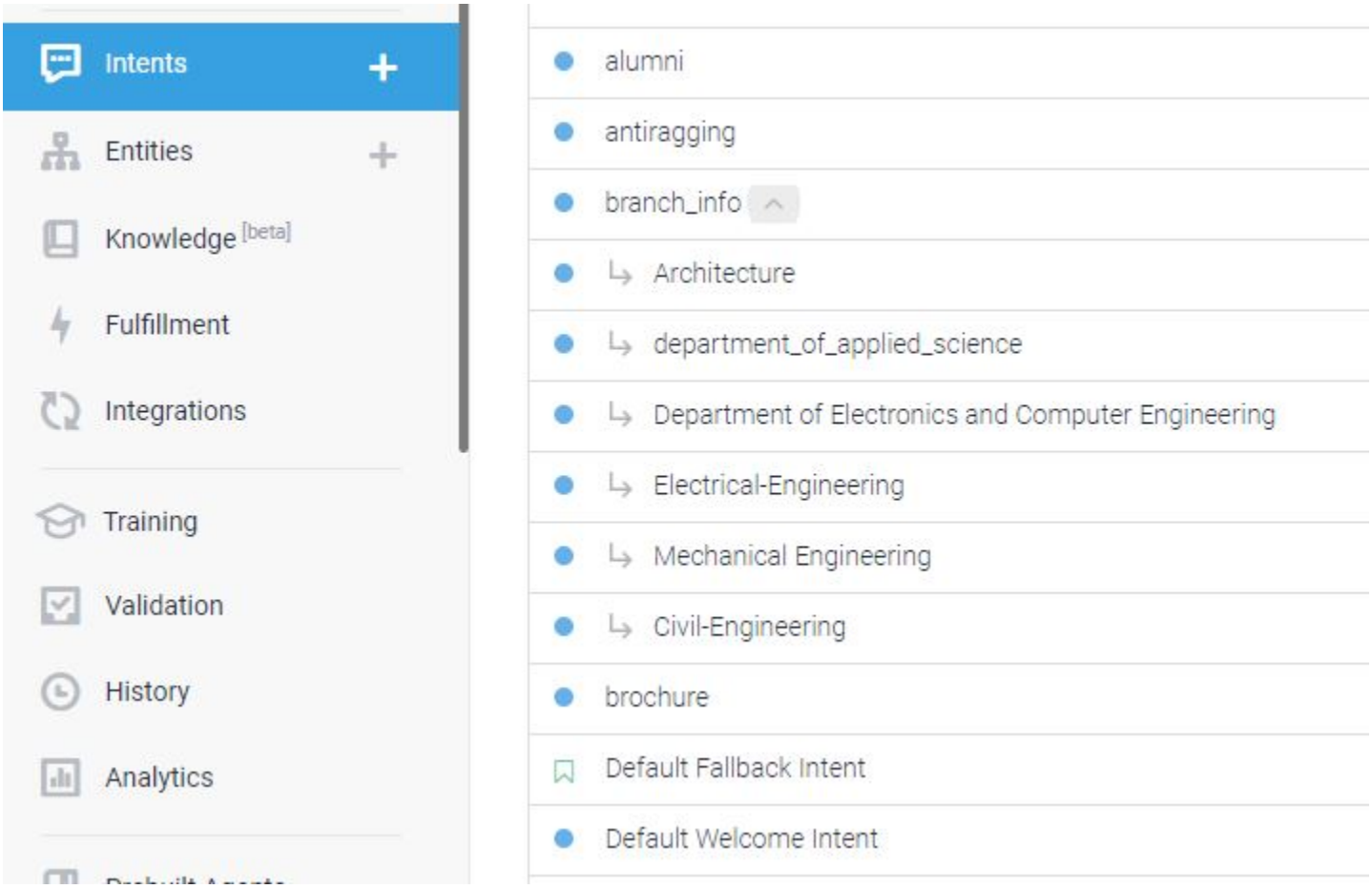

Figure 5.2: Followup-intents

The image depicts various follow-up intents. These follow-up intents are related to different branches of study offered by the college, including Architecture, Department of Applied Science, Department of Electronics and Computer Engineering, Electrical Engineering, Mechanical Engineering, and Civil Engineering. Each follow-up intent is linked to a parent intent related to the user's query about a specific department, allowing the chatbot to provide more information and continue the conversation in a natural and intuitive way.

By using follow-up intents, the chatbot can help prospective students and parents learn more about the college's different departments and programs, and provide personalized responses to their queries. Overall, the image demonstrates how follow-up intents can be used to create a seamless and engaging conversation between a chatbot and its users, and help them find the information they need quickly and easily.

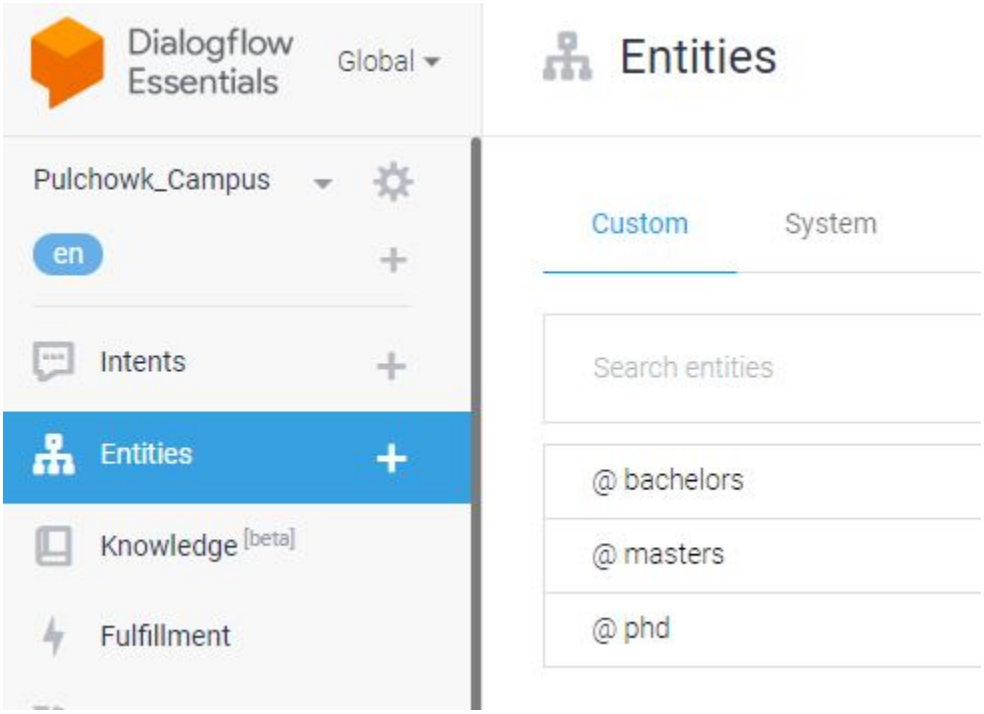

Figure 5.3: Entities

The image displays various entities related to different academic programs. The chatbot is designed to provide information about bachelor's, master's, and PhD programs offered by the college. The entities shown in the image include program names, program descriptions, program requirements, and program duration. These entities are used by the chatbot to understand user queries related to academic programs and provide appropriate responses. The use of entities in the chatbot allows for a more personalized and relevant experience for users seeking information about academic programs, by providing accurate and specific information related to each program.

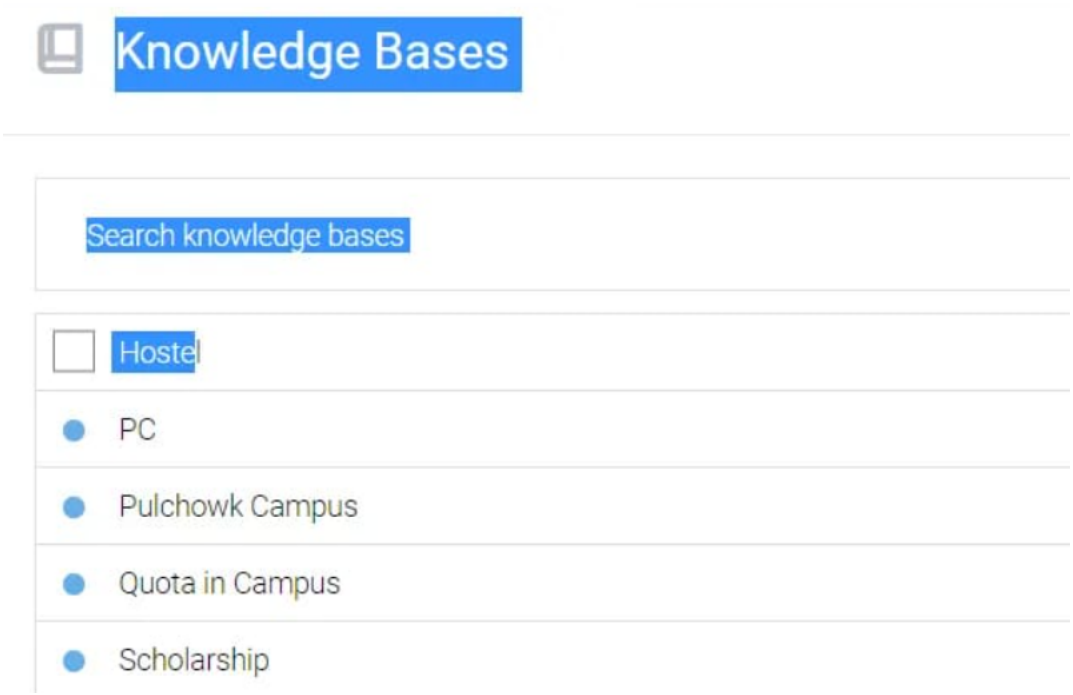

Figure 5.4: Knowledge bases

The above figure shows knowledge bases. Knowledge bases are typically used in situations where the chatbot needs to provide more specific or detailed information about a particular topic. The knowledge base could include information such as program descriptions, course schedules, admission requirements, faculty profiles, and campus facilities.

By using a knowledge base in the chatbot, students and other website visitors can get quick and accurate answers to their questions about the college and its programs. For example, a student might ask the chatbot about the requirements for the computer science program. The chatbot could access the knowledge base to provide a detailed response that includes the necessary coursework, GPA requirements, and other relevant information.

The knowledge base could also include answers to frequently asked questions about the college, such as tuition costs, financial aid options, and housing information. This would allow the chatbot to quickly provide helpful information to students and other website visitors, improving the user experience and reducing the workload on human staff.

Overall, a knowledge base in Dialogflow can be a valuable tool for improving the effectiveness of a chatbot for an engineering college website. By providing easy access to structured data and information, the chatbot can provide more personalized and relevant responses to user queries, improving the user experience and helping students make informed decisions about their education.

```
const functions = require('firebase-functions');
const {WebhookClient} = require('dialogflow-fulfillment');
const {Card, Suggestion} = require('dialogflow-fulfillment');

process.env.DEBUG = 'dialogflow:debug'; // enables lib debugging statements

exports.dialogflowFirebaseFulfillment = functions.https.onRequest(
(request, response) => {
  const agent = new WebhookClient({ request, response });
  console.log('Dialogflow Request headers: ' + JSON.stringify(request.headers));
  console.log('Dialogflow Request body: ' + JSON.stringify(request.body));

  function welcome(agent) {
    agent.add(`Welcome to my agent!`);
  }

  function fallback(agent) {
    agent.add(`I didn't understand`);
    agent.add(`I'm sorry, can you try again?`);
  }
```

Figure 5.5: Nodejs code

Dialogflow offers a Node.js client library that allows developers to easily integrate Dialogflow functionality into their Node.js applications. The Node.js client library provides a simple and intuitive API for accessing Dialogflow's natural language processing capabilities and building conversational interfaces.

One important part of the Node.js code section of Dialogflow is the initialization of the Dialogflow client object, which allows the application to interact with the Dialogflow API. This is typically done using the dialogflow module and the SessionsClient class, which allows the application to create and manage sessions with the Dialogflow API.

Once the Dialogflow client object has been initialized, the application can use it to send user queries to the Dialogflow API and receive responses. This is typically done using the **detectIntent** method, which takes a **DetectIntentRequest** object as input and returns a **DetectIntentResponse** object. The **DetectIntentRequest** object contains the user's query, along with other information such as the session ID and the language code.

In addition to sending user queries and receiving responses, the Node.js client library also provides a range of other functionality, such as managing contexts, handling events, and working with fulfillment webhooks. Developers can use these features to build complex conversational interfaces that can handle a wide range of user queries and actions.

Overall, the Node.js code section of Dialogflow provides a powerful and flexible framework for building chatbots and other conversational interfaces using Node.js. By leveraging

Dialogflow's natural language processing capabilities and the Node.js client library's simple and intuitive API, developers can create chatbots that can provide helpful and engaging experiences for users.

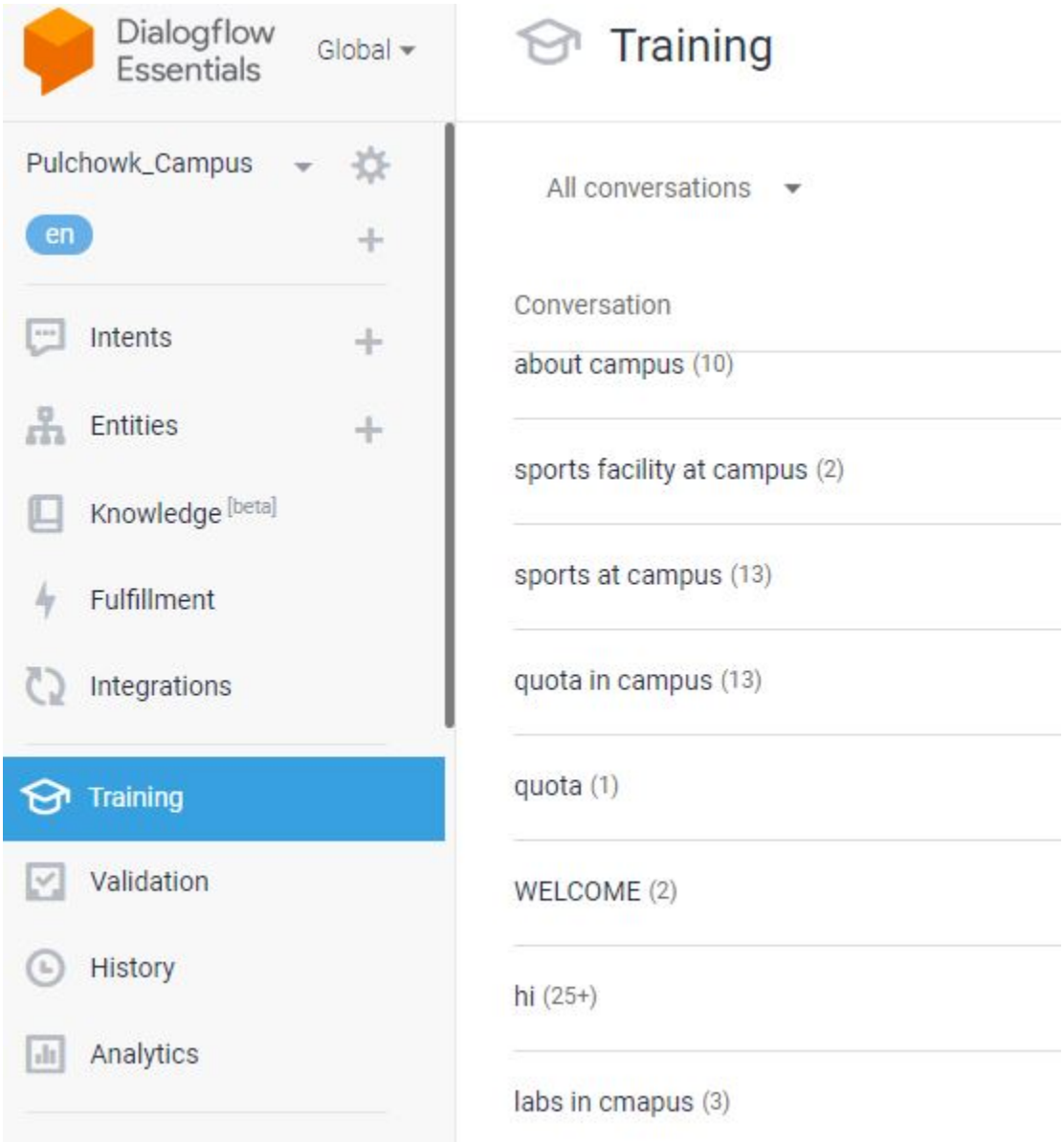

Figure 5.6: Training

The figure above shows the training section. The training section of Dialogflow is where users can teach their chatbot how to understand and respond to user queries. This process is also known as "training the model".

- To begin the training process, the user creates a set of intents that correspond to the types of questions or requests that users might make when interacting with the chatbot. For instance, if the chatbot is designed to assist prospective students, the user might create an intent for questions about program descriptions, another intent for admission requirements, and so on.
- After defining the intents, the user can then add training phrases to each intent. These training phrases represent the different ways that a user might express the same intent. For instance, for the "program descriptions" intent, the training phrases might include "What are the available programs?" or "Can you tell me about the engineering program?"
- Next, the user can define entities that correspond to specific pieces of information within the user's query. For instance, if a user is inquiring about admission requirements, the user might define entities for high school GPA, standardized test scores, and so on.
- Finally, the user can define fulfillment logic for each intent. This might involve generating a response that includes information from a database, triggering an action such as scheduling a campus visit, or transferring the conversation to a human agent.

Once the chatbot is trained, it can begin to understand and respond to user queries in a way that is tailored to the user's intent and context. Over time, as users interact with the chatbot, the model can be refined and improved by analyzing user feedback and making adjustments to the intents, entities, and fulfillment logic.

5.4.2 Results Snapshots

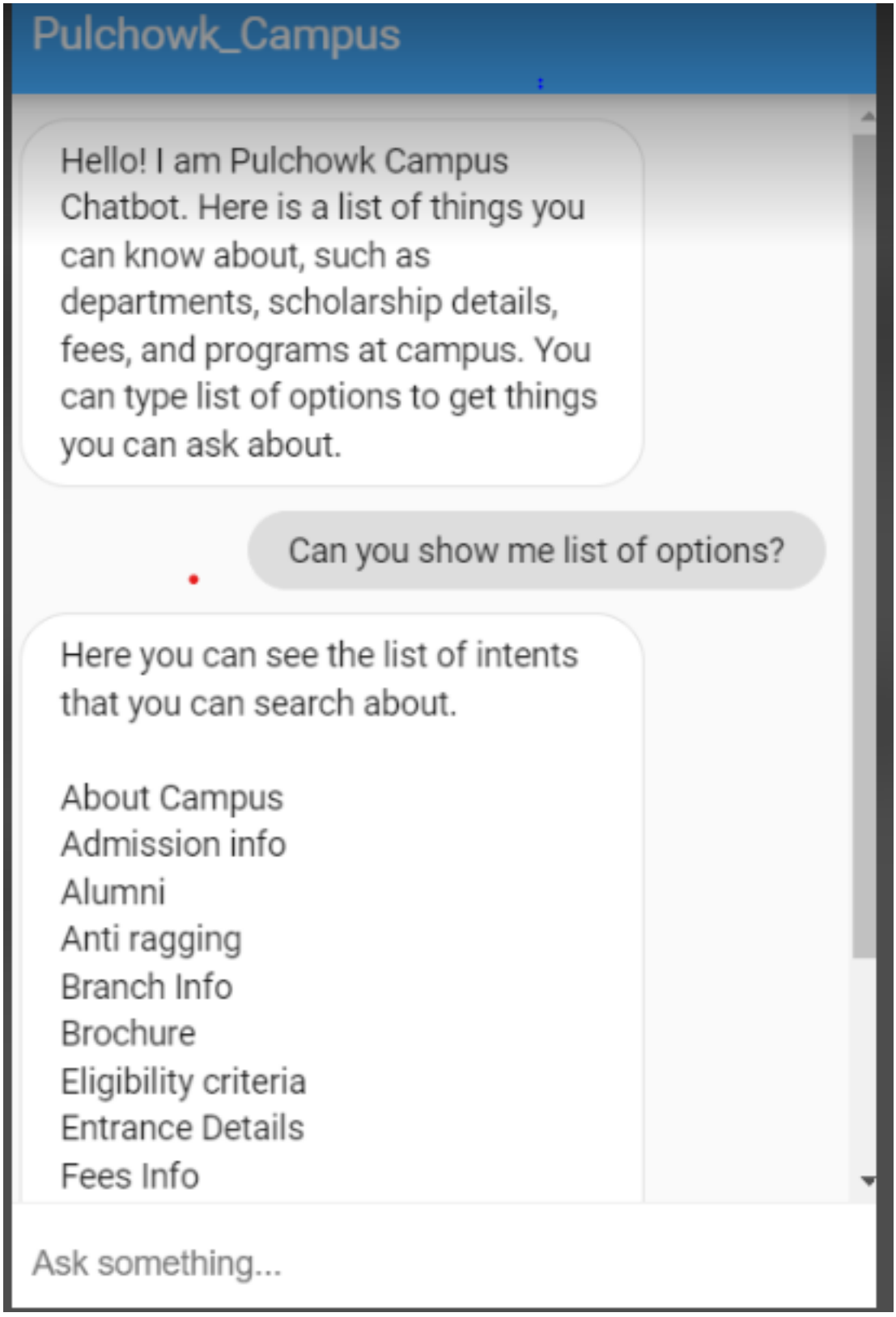

Figure 5.7: Starting of conversation

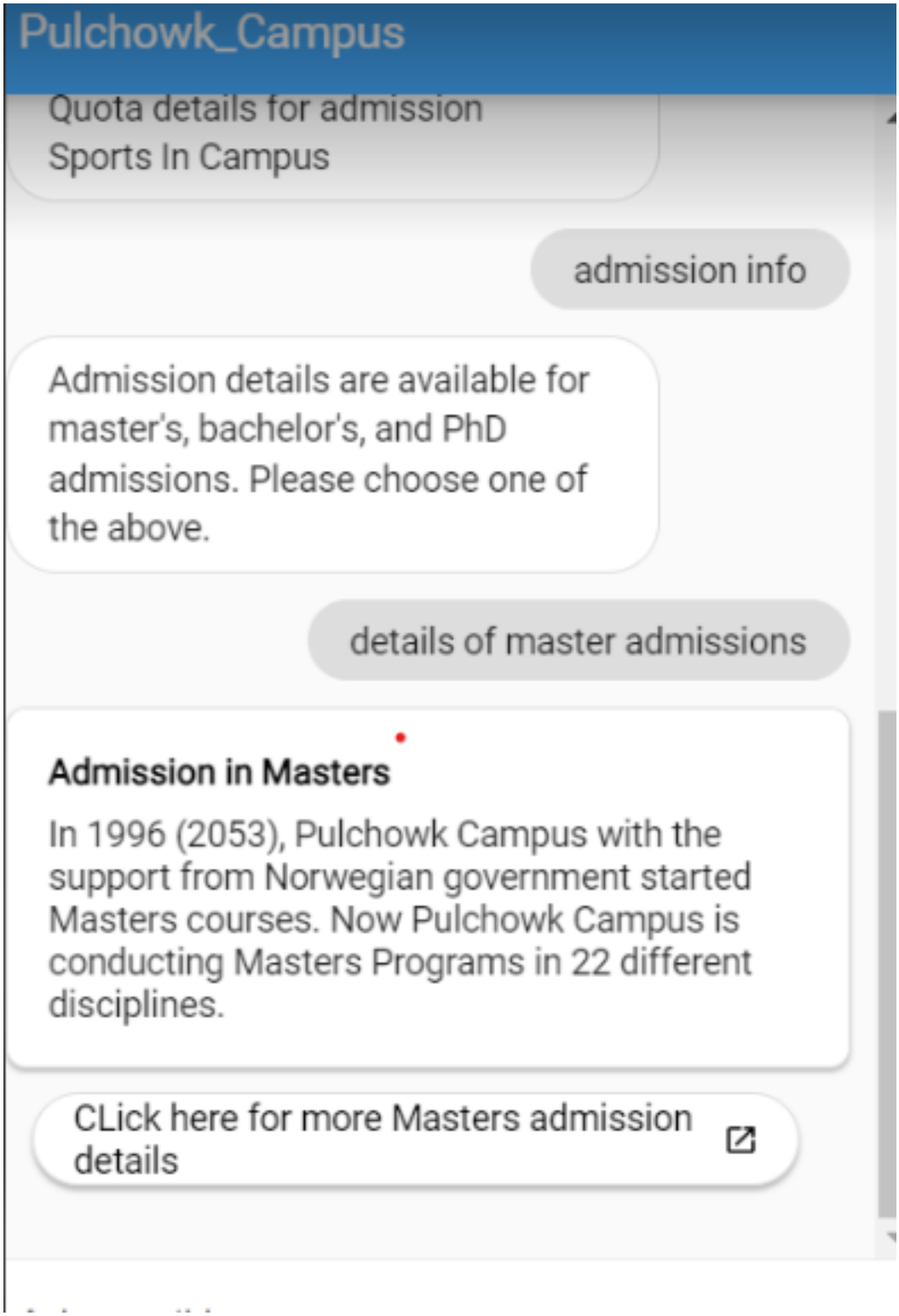

Figure 5.8: Obtaining Information

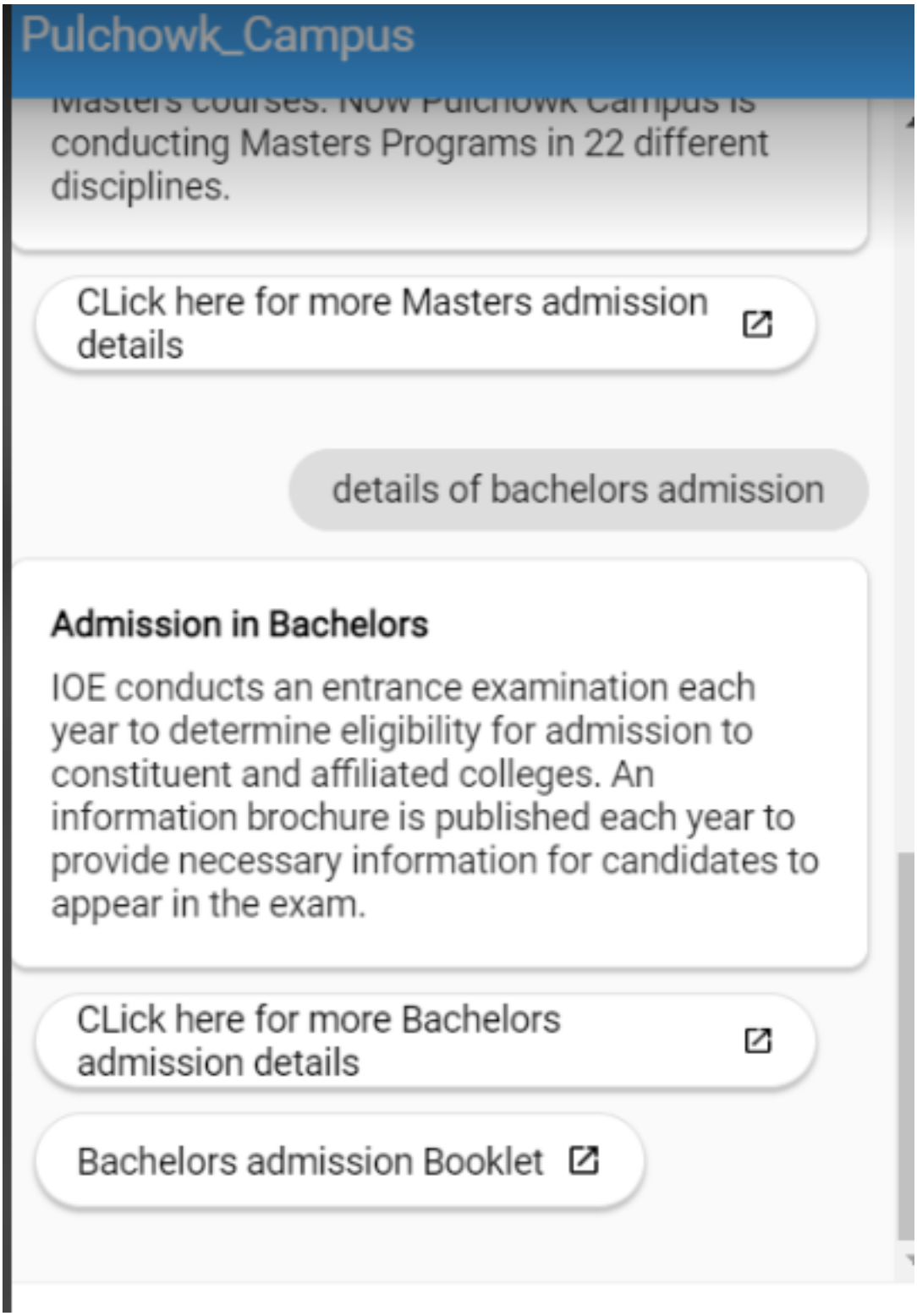

Figure 5.9: Recommendations

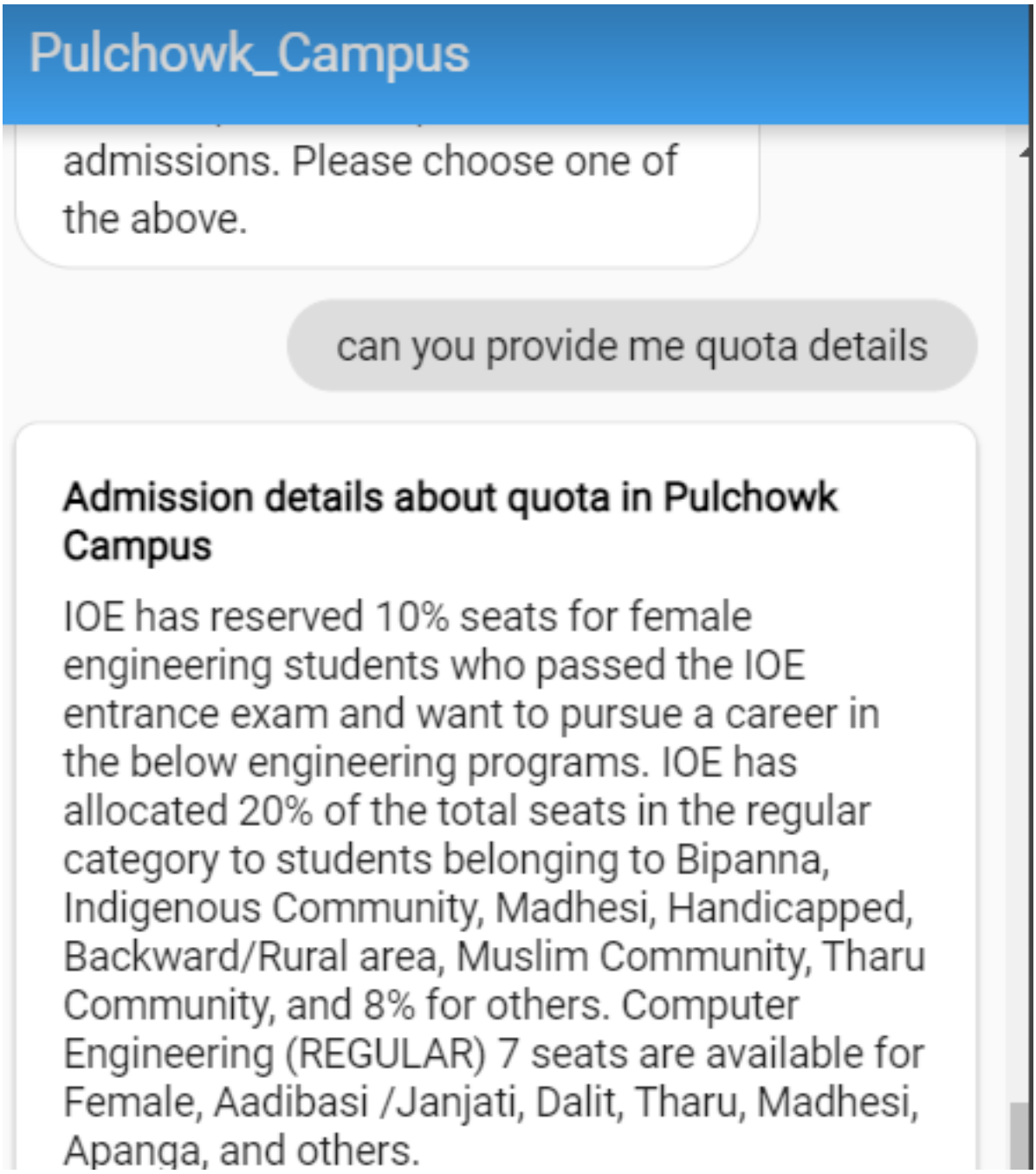

Figure 5.10: Quota information

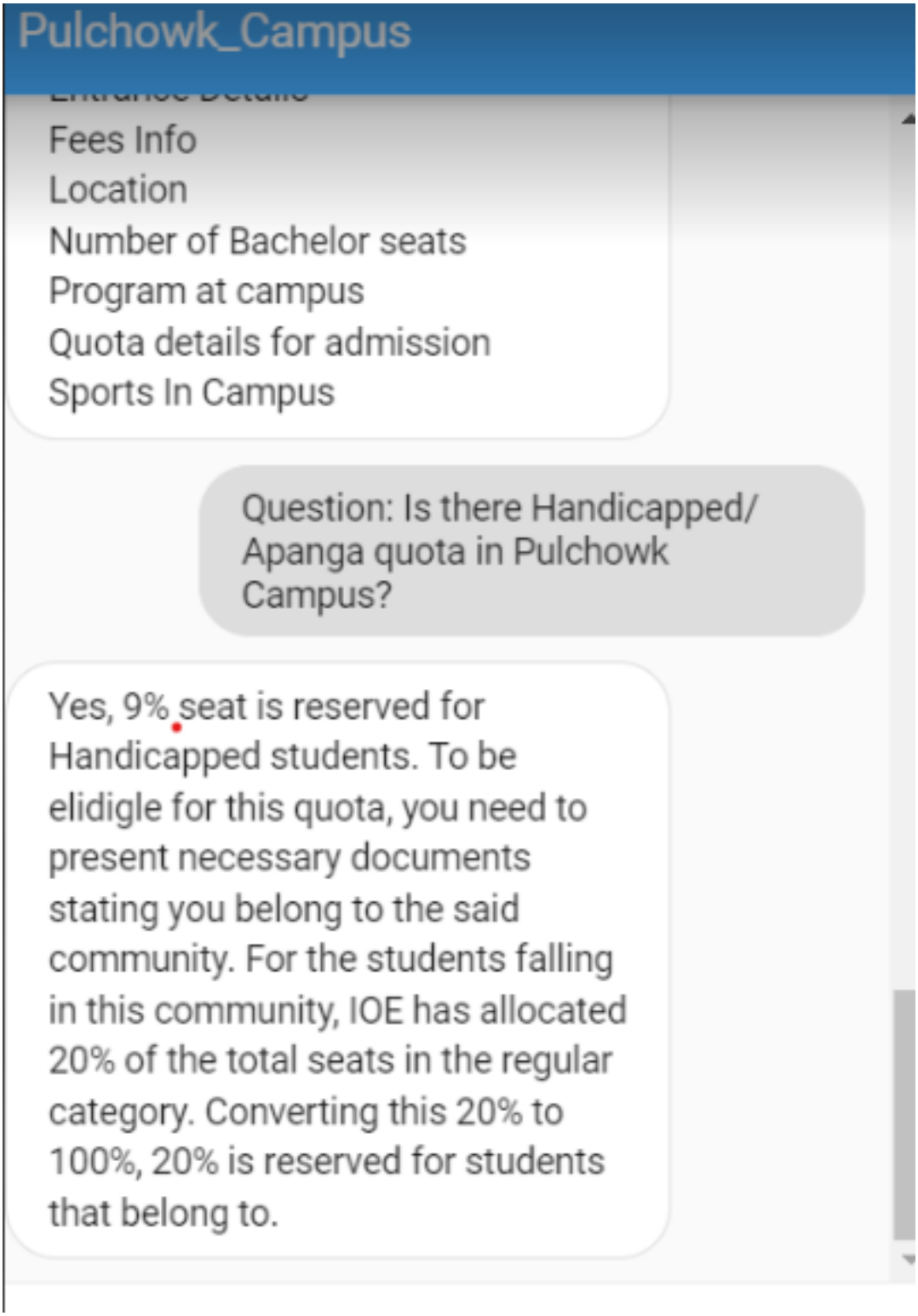

Figure 5.11: Sub-Quota information

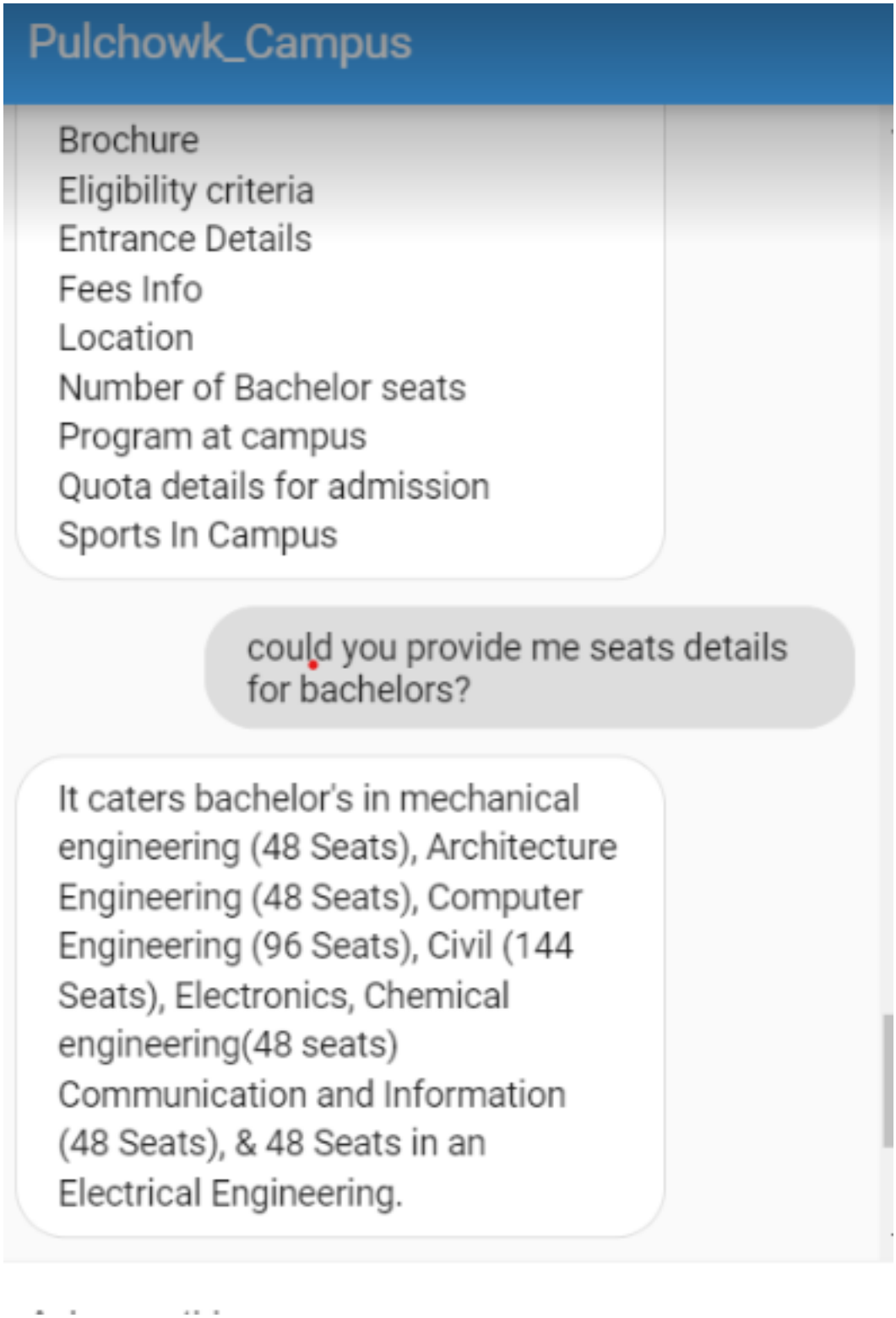

Figure 5.12: Seat details

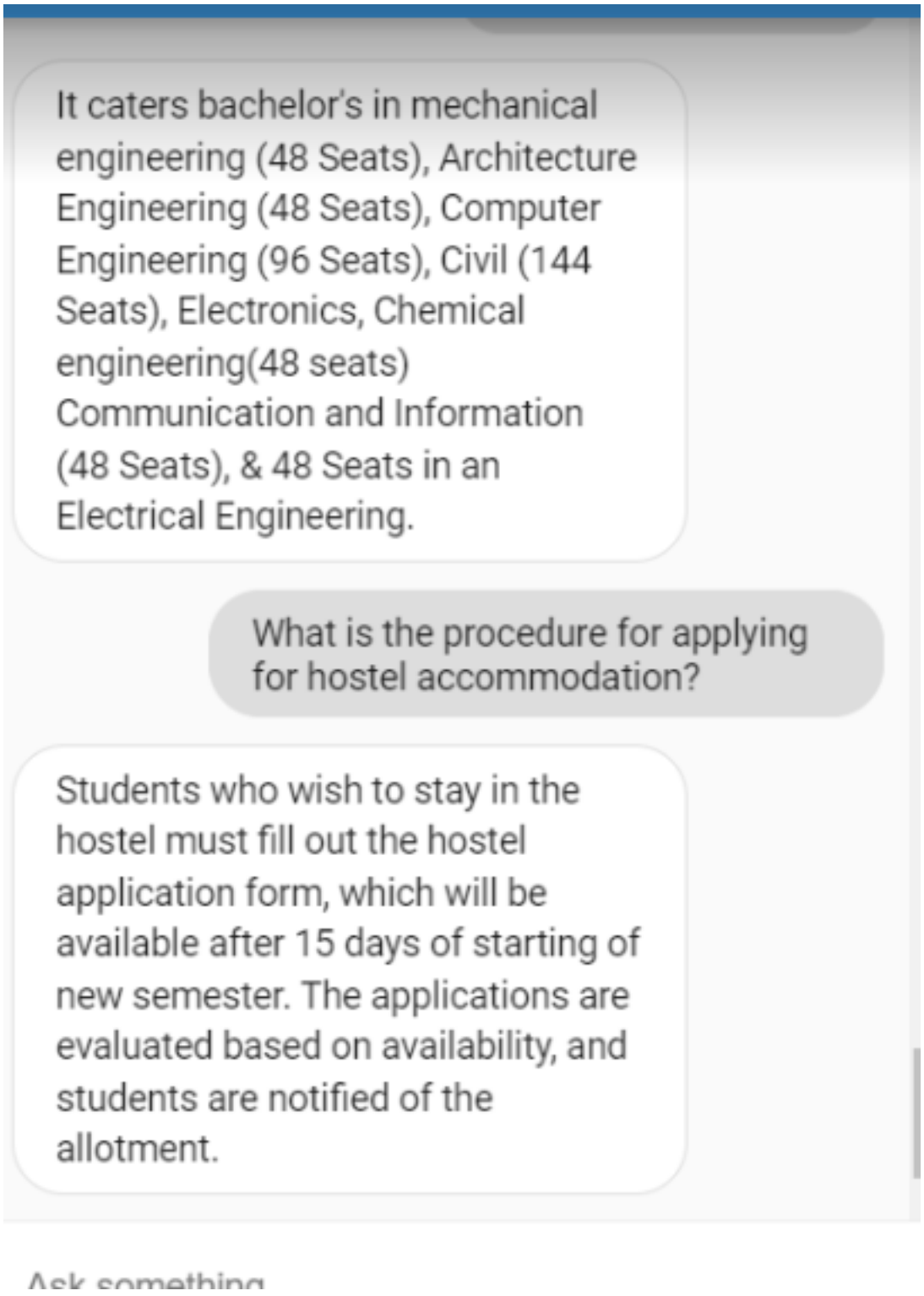

Figure 5.13: Hostel accommodation details

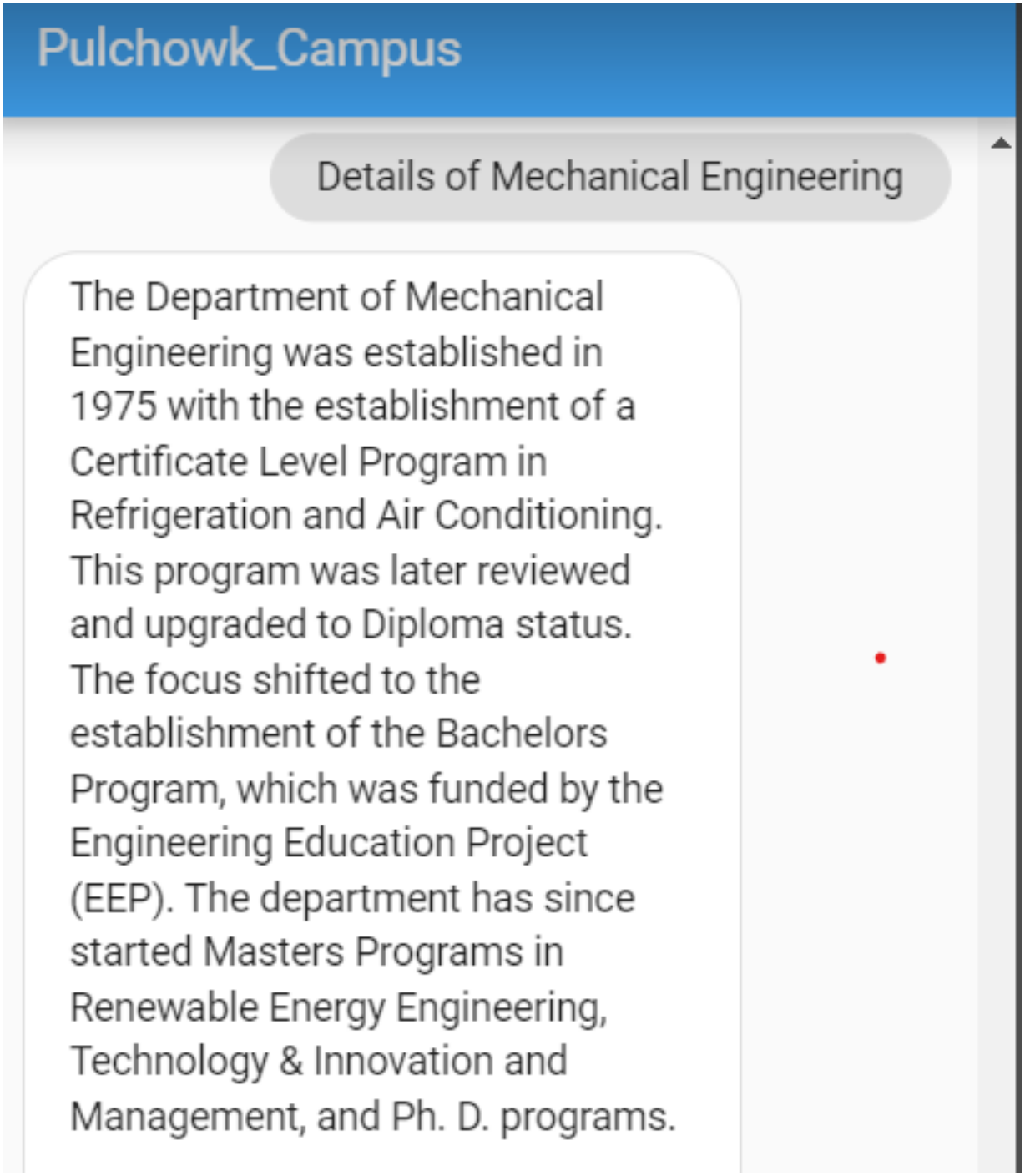

Figure 5.14: Department details

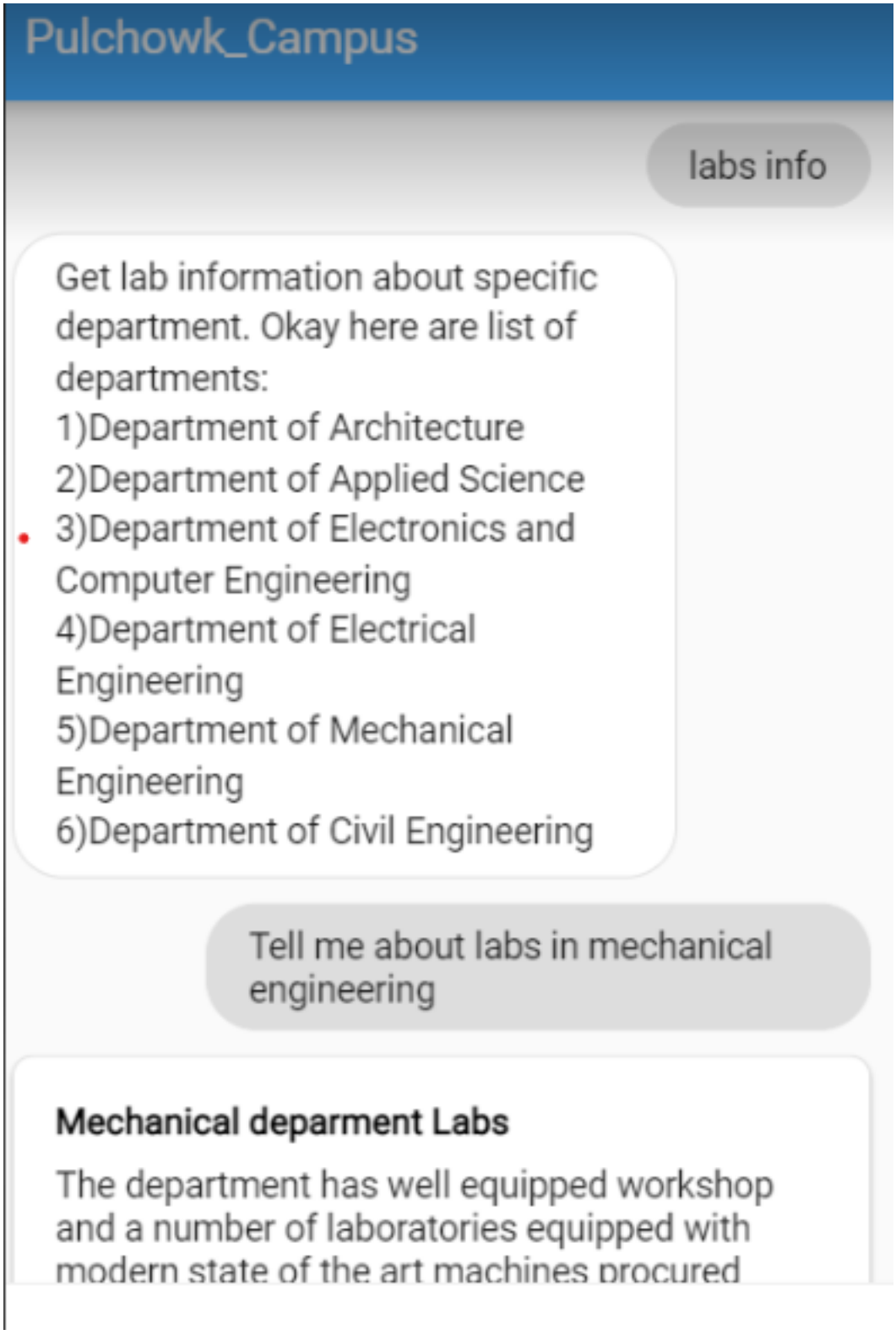

Figure 5.15: Lab information

6. Epilogue

6.1 Use Cases

The AI-powered Chatbot - Gabby can help us in many ways. The results can be seen in benefits like-

- It can provide the most convenient way to efficiently search for the most relevant queries from students.
- It can provide support that can help the user with the most basic queries 24/7.
- The ease of working to find different features on our university website.
- You can create a fun situation for parents and students with additional technical support.
- This application saves time for both the student and the teaching and non-teaching staff.
- Helps to get rid of manual efforts.

6.2 Conclusion

Chatbots are becoming a standard part of the digital world. Undergraduates and workers can freely exchange questions.The chatbot provides quick and effective responses to questions and obtains important connections to their query. The system's purpose is to keep students informed about their college.The project's major goal is to lessen the workload of the college's office personnel and the time it takes to respond to a user's enquiry

6.3 Limitations

Limited Control Over the Conversation Flow: While Dialogflow provides a visual interface to design conversation flows, it can be challenging to control the conversation flow precisely. This can lead to unexpected user interactions and may require additional effort to refine the conversation flow.

Lack of Flexibility: Dialogflow is primarily designed for building text-based chatbots and may not be suitable for building voice-based or multi-modal chatbots. If you need to build

a chatbot that supports voice-based interactions or integrates with other modalities such as video or images, you may need to consider other platforms.

Integration Complexity: While Dialogflow provides integrations with popular messaging platforms such as Facebook Messenger and Slack, integrating with other platforms may require custom code and additional development effort. This can make it challenging for teams without a strong technical background to build and deploy chatbots on multiple platforms.

6.4 Future Scope

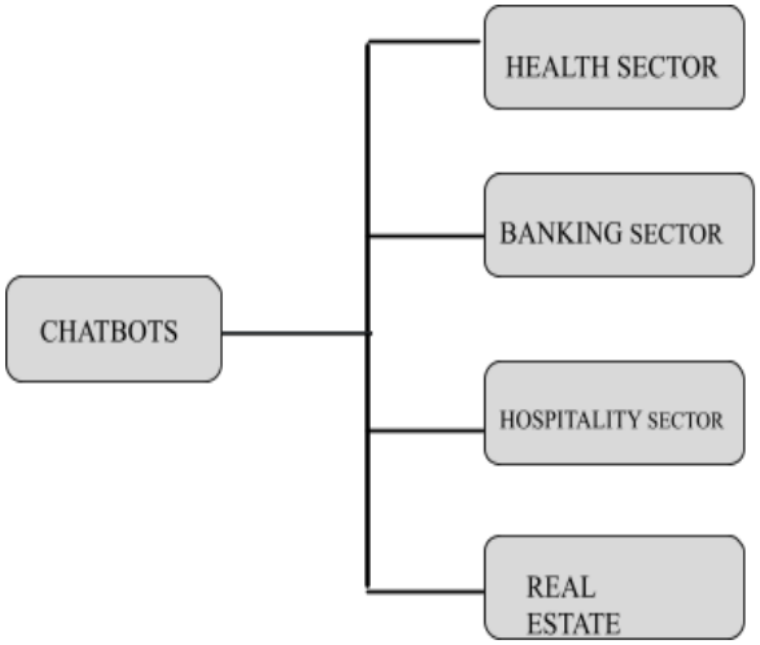

Figure 6.1: Future Scope

The future scope of chatbots could include many benefits for businesses, but according to experts, chatbots need to be gently pushed in the right direction to reap the benefits for businesses.

The growth rate of chatbots is 24represent a 1.25 billion market by 2025, according to this report by The Grand View Research Inc. Due to their human conversational flexibility and 24/7 service, these chatbots are becoming essential to maintain their longevity within business parameters.

There are many areas where chatbots dominate and they will become the future of many of these areas.

Future bots can accurately understand the full syntax and intent of a user's input to deliver information that is specific and relevant to the complex problem.

References

- Bocklisch, T., Faulkner, J., Pawlowski, N., and Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. arXiv preprint arXiv:1712.05181.
- https://github.com/RasaHQ/rasa
- Lacerda, A. R. T. D. (2019). Rasa-ptbr-boilerplate: FLOSS project that enables Brazilian Portuguese chatbot development by non-experts.
- Jiao, A. (2020). An Intelligent Chatbot System Based on Entity Extraction Using NLU and Neural Network. JPhCS, 1487(1), 012014.
- Singh, A., Ramasubramanian, K., andShivam, S. (2019). Introduction to Microsoft Bot, RASA, and Google Dialogflow. In Building an Enterprise Chatbot (pp. 281-302). Apress, Berkeley, CA.
- Frommert, C., Häfner, A., Friedrich, J., and Zinke, C. (2018, September). Using chatbots to assist communication in collaborative networks. In Working Conference on Virtual Enterprises (pp. 257-265). Springer, Cham.
- Segura, C., Palau, A., Luque, J., Costa-Jussà, M. R., and Banchs, R. E. International Workshop on Spoken Dialogue System Technology (pp. 319-330). Springer, Singapore.
- Tembhekar S and Kanojiya M 2017 A Survey Paper on Approaches of Natural Language Processing (NLP) J. International Journal of Advance Research, Ideas and Innovations in Technology 3(3) p 1496-98.42
- Kamper H and Ruder 2018, October 1 A Review of the Neural History of Natural Language Processing Aylien Retrieved March 18, 2019, from recent-history-of-natural-language-processing
- Jiao, A. (2020, March). An intelligent chatbot system based on entity extraction using NLU and neural network. In Journal of Physics: Conference Series (Vol. 1487, No. 1, p.012014). IOP Publishing.
- https://spotleai.sgp1.digitaloceanspaces.com/course/pdf/Chatbot$_{ag}$*upta.pdf*