

DOCUMENTATION ON GOOGLE DIALOGFLOW

CHATBOT

1. Chatbot Fundamentals

1.1 What is a chatbot?

A chatbot is a software application that mimics conversation with a human in natural languages through various platforms like messaging, websites, mobiles etc. The chatbot responds by identifying the intent of the conversation and then responding accordingly.

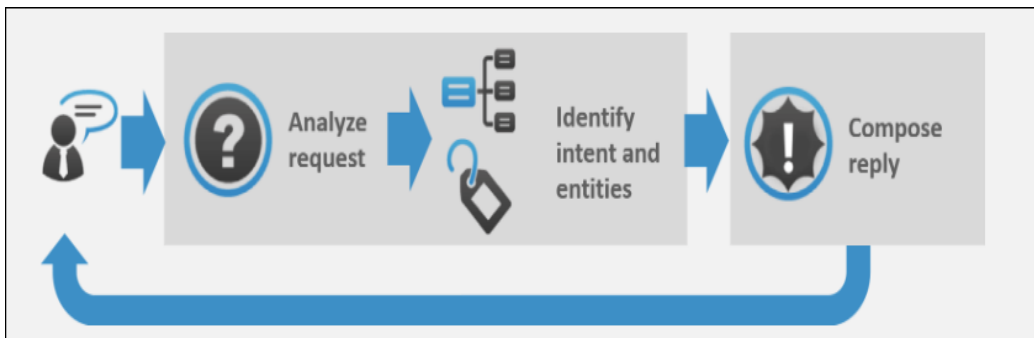


Image Courtesy: <https://expertsystem.com/chatbot/>

Broadly, chatbots can be classified into two categories:

1. Rule-Based Chatbots: This is an extremely fundamental type of chatbot which generally works on simple 'if-else' constructs. It can respond to only simple predefined queries. The performance of this application highly depends on the programming skills of the developer.
2. Chatbots with Natural Language Understanding: At the core, it has a language processing and understanding model with pre-trained instances using Deep Learning. It can communicate through both text and speech.

1.2 Uses of chatbots:

- a) Can be used to answer FAQs.
- b) Can be used for grievance handling.
- c) Internal organizational automation.
- d) To do the flight, hotel, appointment bookings etc.
- e) Can guide customers to buy the correct product by answering their questions
- f) Can be used for Customer Relationship Management.

1.3 Advantages of Chatbots

- a) 24*7 customer support.
- b) Uniform customer experience.
- c) Cost-efficient.
- d) Build once and deploy everywhere.
- e) Integration with various channels and platforms

f) Better monitoring and insight generation.

1.4 Frameworks Present in the Market:

- a) **Google Dialogflow:** Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on.
- b) **Microsoft Bot Builder with LUIS:** Azure Bot Service enables you to build intelligent, enterprise-grade bots with ownership and control of your data.
- c) **Amazon Lex:** Amazon Lex is a service for building conversational interfaces into any application using voice and text. Amazon Lex provides the advanced deep learning functionalities of automatic speech recognition (ASR) for converting speech to text, and natural language understanding (NLU) to recognize the intent of the text, to enable you to build applications with highly engaging user experiences and lifelike conversational interactions.
- d) **RASA:** Rasa provides infrastructure & tools necessary for high-performing, resilient, proprietary contextual assistants that work.
- e) **Wit.ai (Facebook):** Wit.ai makes it easy for developers to build applications and devices that you can talk or text to.

2. Google Dialogflow

2.1 Introduction:

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

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

2.2 Signup For Dialogflow:

You need to have a google account to signup for Dialogflow.

- a) Go to <https://dialogflow.com/> and click on 'Sign Up for Free' button.

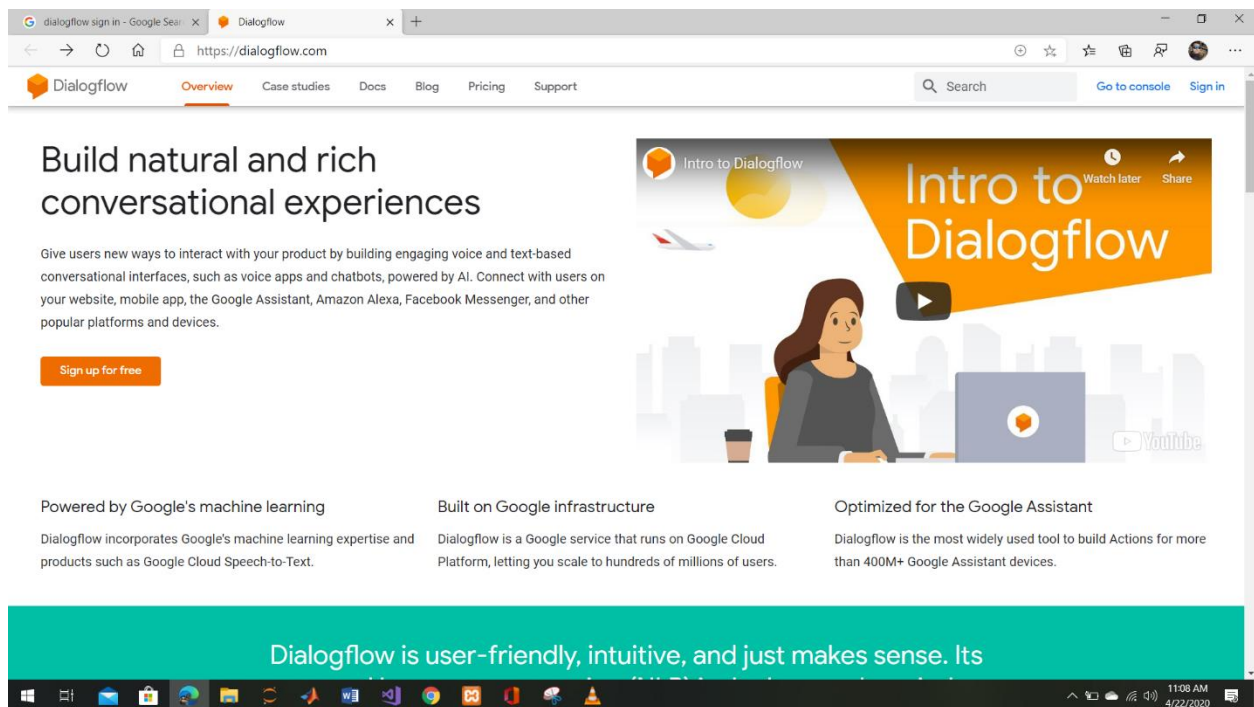


Figure 1: Sign up for Google Dialogflow

- b) Click 'sign-in with Google'.
- c) Select your google account and once you are redirected, click on 'Go To Console' on the upper right corner of the screen.

2.3 Dialogflow Console:

Dialogflow provides a web user interface called the *Dialogflow Console* ([open console](#)). You use this console to create, build, and test agents. The uses are:

- Create Agents
- Create Intents
- Create entities
- Fulfillment to connect to other APIs.
- Integrate the bot with other platforms
- Analyze agent performance
- Test the agent in conversation simulator.

3. Building a chatbot using google Dialogflow

3.1 The problem statement:

To build a chatbot which can answer all the queries of a customer and whenever a customer does an enquiry, it automatically send the customer the course details. Also an email is sent to the support team to assist the customer further with their queries.

3.2 Agent:

A Dialogflow *agent* is a virtual agent that handles conversations with your end-users. It is a natural language understanding module that understands the nuances of human language. A Dialogflow agent is similar to a human call center agent. You train them both to handle expected conversation scenarios, and your training does not need to be overly explicit.

3.2.1 Creating an Agent:

- a) Click on *Create Agent* from the left menu.
- b) Provide the name of the agent and click on the *SAVE* button to create the Agent.

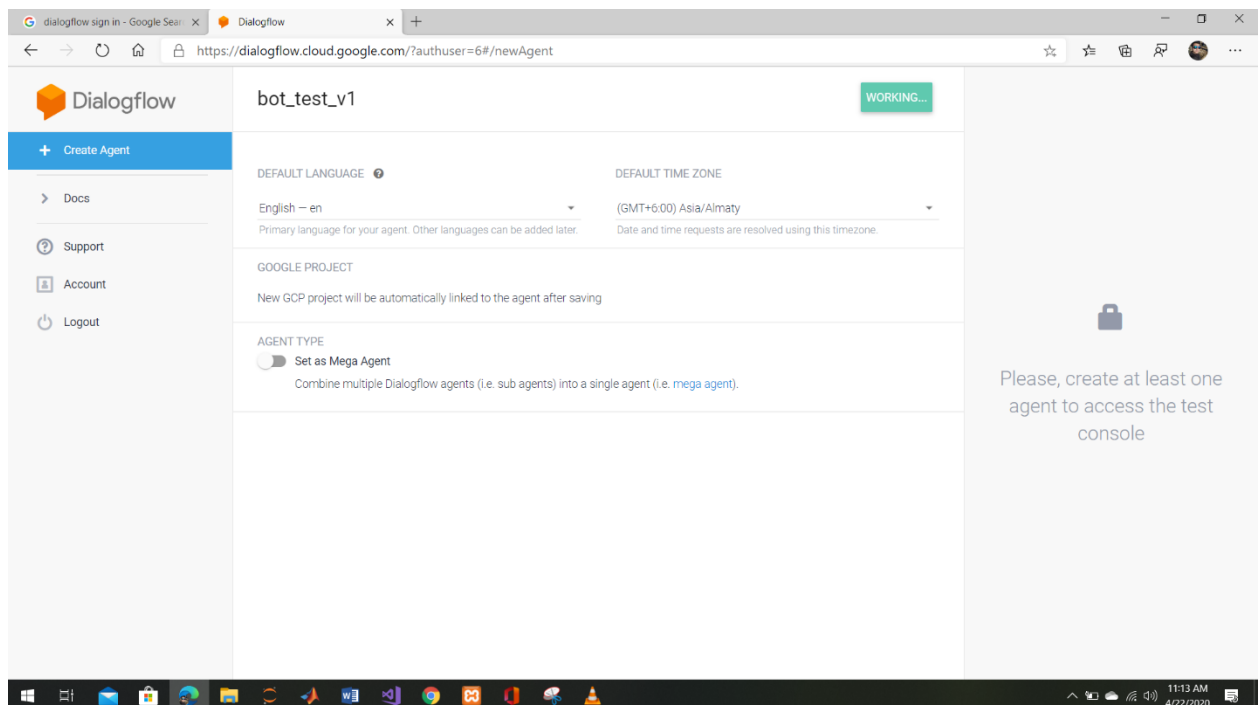


Figure 2: Creating an Agent for our ChatBot.

- c) After saving, the agent is shown in the left hand side of your console. You can click the gear icon to edit the agent settings.

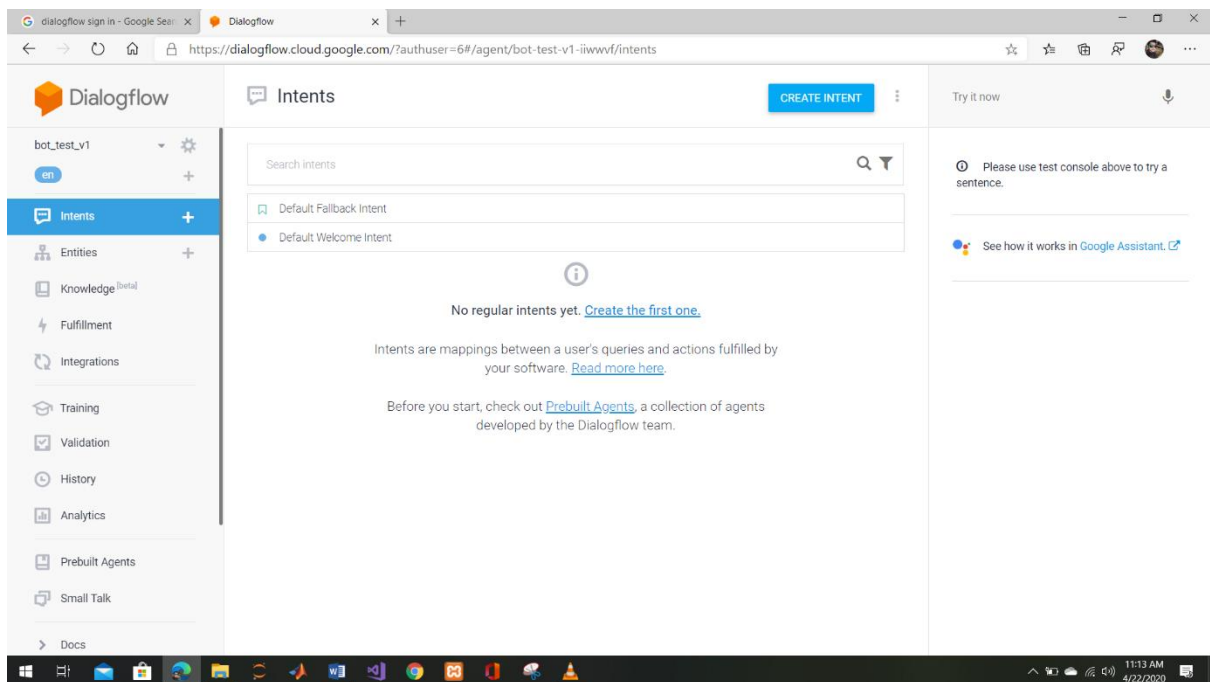


Figure 3: Gear icon to edit the Agent Settings.

3.3 Intent:

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

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

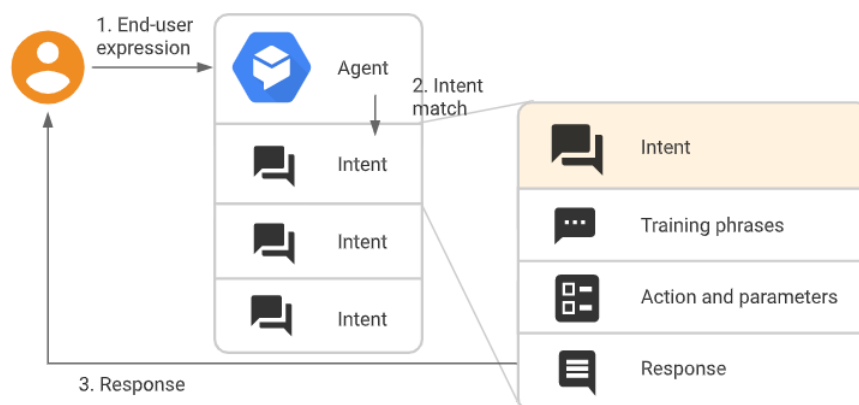


Figure 4: Flow-Chart of Response

3.3.1 Creating an Intent:

- a) Click the + add button next to Intents in the left sidebar menu. Enter a name for your intent. Your intent name should represent the end-user expressions it recognizes and click Save.

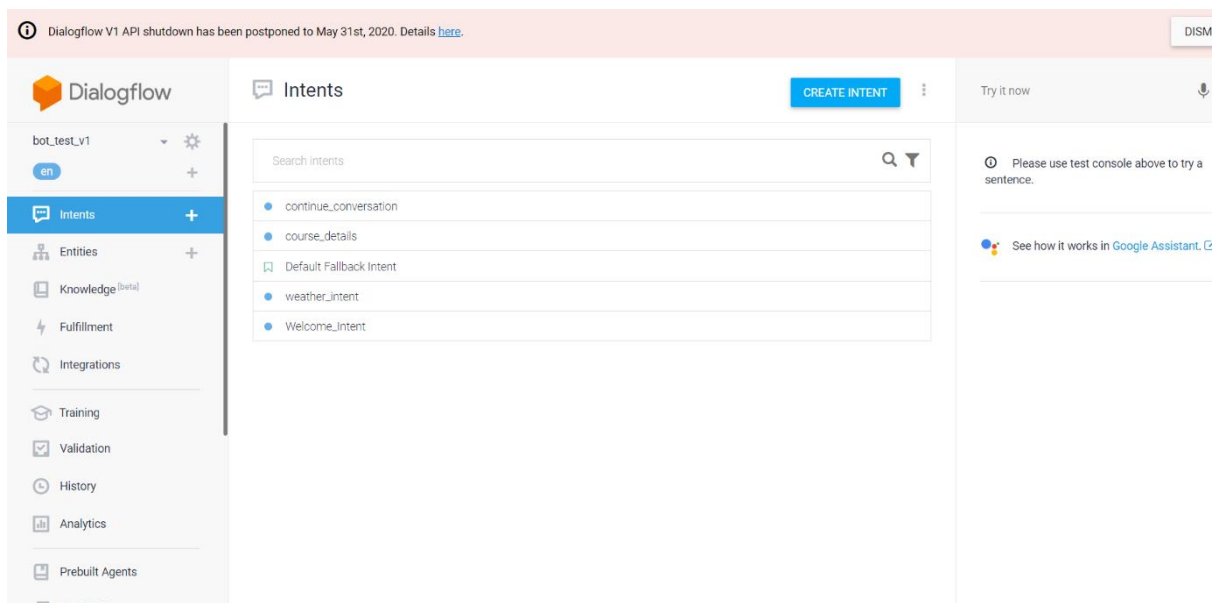


Figure 5: Add button to create an Intent.

3.3.1.1 Training Phrases:

Training phrases are example phrases for what end-users might type or say, referred to as *end-user expressions*. For each intent, you create many training phrases. When an end-user expression resembles one of these phrases, Dialogflow matches the intent.

Adding the training Phrases:

- a) Click the text field that shows "Add user expression".

b) Type your training phrases and press the Enter key after each.

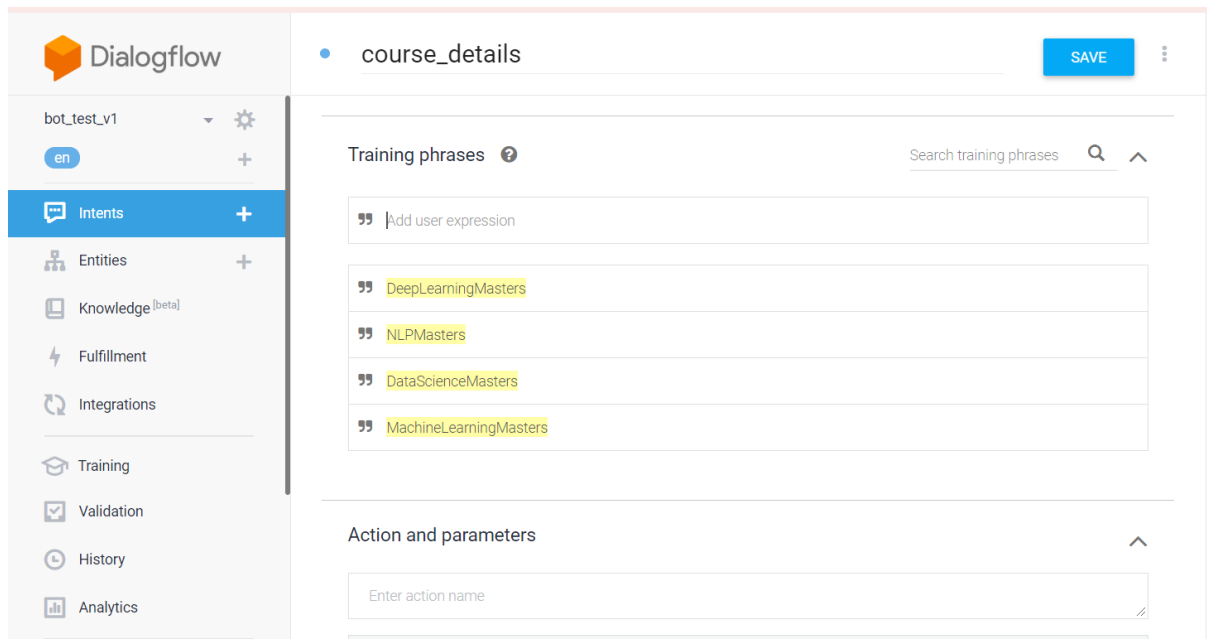


Figure 6: Listing out the training phrases for the model.

3.3.1.2 Extracting the Entities:

When an intent is matched at runtime, Dialogflow provides the extracted values from the end-user expression as *parameters*. Each parameter has a type, called the entity type, which dictates exactly how the data is extracted. Unlike raw end-user input, parameters are structured data that can easily be used to perform some logic or generate responses.

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

Dialogflow provides predefined system entities that can match many common types of data. For example, there are system entities for matching dates, times, colors, email addresses, and so on. You can also create your own custom entities for matching custom data.

REQUIRED	PARAMETER NAME	ENTITY	VALUE	IS LIST	PROMPTS
<input checked="" type="checkbox"/>	course_name	@course_name	\$course_name	<input type="checkbox"/>	Please Enter C o...
<input checked="" type="checkbox"/>	cus_name	@sys.any	\$cus_name	<input type="checkbox"/>	Please enter yo...
<input checked="" type="checkbox"/>	cus_mobile	@sys.phone-num	\$cus_mobile	<input type="checkbox"/>	Please enter yo...
<input checked="" type="checkbox"/>	cus_email	@sys.email	\$cus_email	<input type="checkbox"/>	Please enter yo...
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>	—

Figure 7: Specifying the Action and Parameters in Intents Tab.

3.3.1.3 Specifying custom Responses:

Intents have a built-in response handler that can return responses after the intent is matched. This feature only supports static responses, though you can use parameter references in these responses to make them somewhat dynamic. This is helpful for recapping

information provided by the end-user. For example, your intent response could look like: "Okay, I booked a room for you on date".

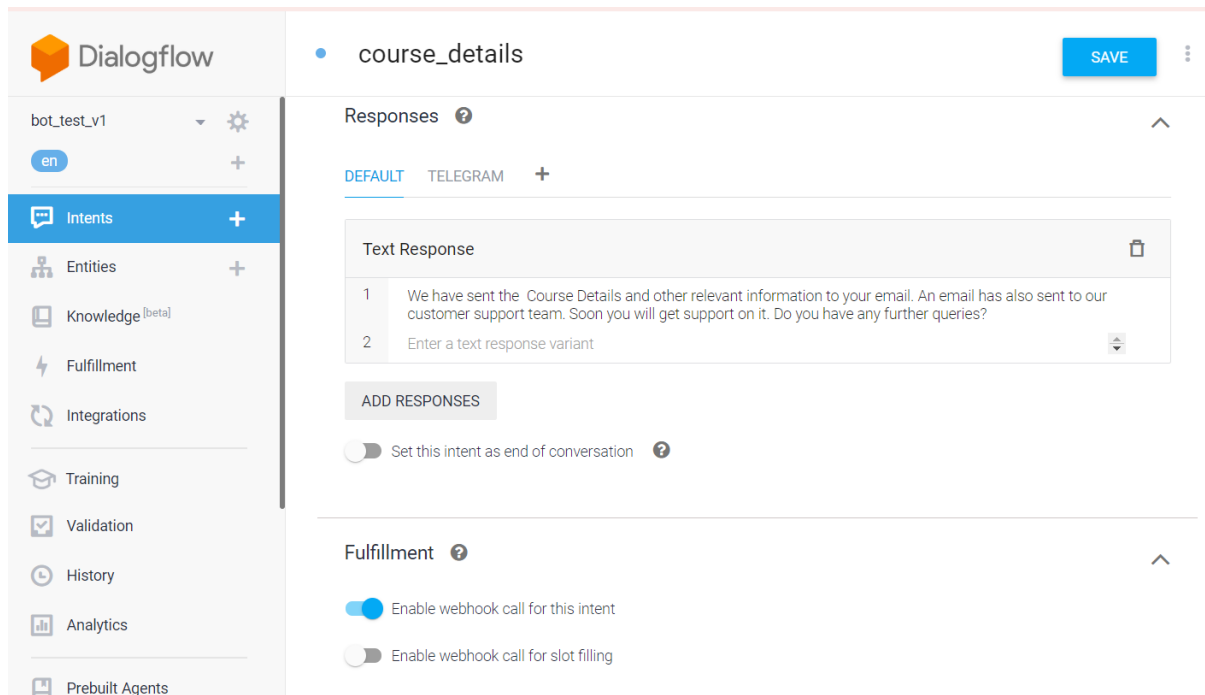


Figure 8: Providing the responses.

3.4 Fulfillment:

By default, your agent responds to a matched intent with a static response. If you're using one of the integration options, you can provide a more dynamic response by using *fulfillment*. When you enable fulfillment for an intent, Dialogflow responds to that intent by calling a service that you define. For example, if an end-user wants to schedule a haircut on Friday, your service can check your database and respond to the end-user with availability information for Friday.

Each intent has a setting to enable fulfillment. If an intent requires some action by your system or a dynamic response, you should enable fulfillment for the intent. If an intent without fulfillment enabled is matched, Dialogflow uses the static response you defined for the intent.

When an intent with fulfillment enabled is matched, Dialogflow sends a request to your *webhook* service with information about the matched intent. Your system can perform any required actions and respond to Dialogflow with information for how to proceed. The following diagram shows the processing flow for fulfillment.

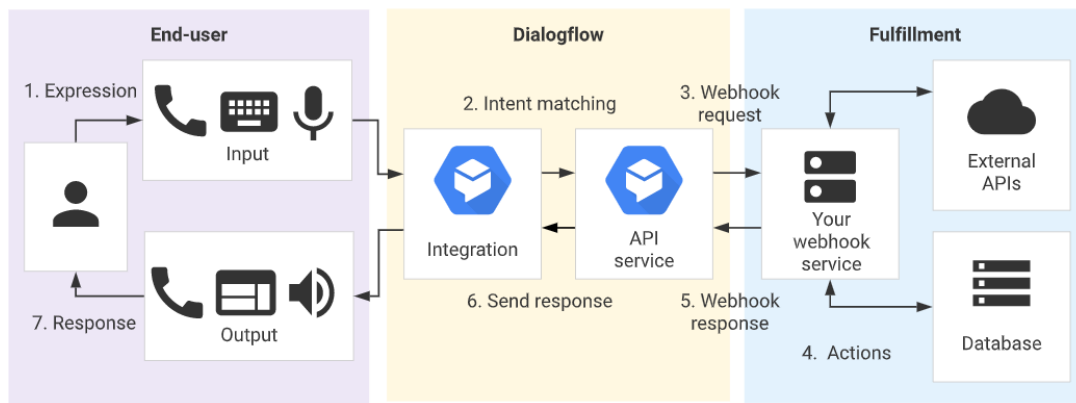


Figure 9: Overall Flow graph for operation in Google DialogFlow.

- a) The end-user types or speaks an expression.
- b) Dialogflow matches the end-user expression to an intent and extracts parameters.
- c) Dialogflow sends a webhook request message to your webhook service. This message contains information about the matched intent, the action, the parameters, and the response defined for the intent.
- d) Your service performs actions as needed, like database queries or external API calls.
- e) Your service sends a webhook response message to Dialogflow. This message contains the response that should be sent to the end-user.
- f) Dialogflow sends the response to the end-user.
- g) The end-user sees or hears the response.

3.4.1 Functionalities achieved in fulfillment:

- a) Sending an email to the customer with the syllabus and all the course details based on the course selected.
- b) Sending an email to the Support team to further contact the customer for further clarification.

3.4.2 Webhook for fulfilment:

- a) Select Fulfillment in the left sidebar menu.
- b) Toggle the Webhook field to Enabled.
- c) Provide the details for your webhook service in the form. If your webhook doesn't require authentication, leave the authentication fields blank.
- d) Click Save at the bottom of the page.

Dialogflow

Fulfillment

bot_test_v1

en

Intents

Entities

Knowledge ^[beta]

Fulfillment

Integrations

Training

Validation

History

Analytics

Drabuilt Agents

Webhook

ENABLED

Your web service will receive a POST request from Dialogflow in the form of the response to a user query matched by intents with webhook enabled. Be sure that your web service meets all the [webhook requirements](#) specific to the API version enabled in this agent.

URL*

BASIC AUTH

Enter username

Enter password

HEADERS

Enter key

Enter value

+ Add header

SMALL TALK

Disable webhook for Smaltalk

Inline Editor

(Powered by Google Cloud Functions)

DISABLED

Build and manage fulfillment directly in Dialogflow via Cloud Functions. [Docs](#)

[index.js](#) [package.json](#)

Figure 10: Provide the details for your webhook service.

3.4.3 Fulfilment Request and Response:

To see the request being sent to the webhook call, click the 'Diagnostic info' button.

Figure 11: Diagnostic Info button is pressed to check Webhook call request.

Try it now

USER SAYS

1234567890

DEFAULT RESPONSE

We have sent the course syllabus and other relevant details to you via email. An email has been sent to the Support Team with your contact information, you'll be contacted soon. Do you have further queries?

INTENT

course_selection

ACTION

Not available

PARAMETER	VALUE
cust_name	vss
cust_email	[REDACTED]
course_name	DataScienceMasters
cust_contact	[REDACTED]

DIAGNOSTIC INFO

After clicking, you'll see the following structure:

```
{  
  
  "responseId": "f1f8df5c-0fdf-43cc-aca9-a41bfd70acfa-ab1309b0",  
  
  "queryResult": {  
  
    "queryText": "1234567890",  
  
    "parameters": {  
  
      "cust_name": "vss",  
  
      "cust_email": "",  
  
      "course_name": "DataScienceMasters",  
  
      "cust_contact": "1234567890"  
  
    },  
  
    "allRequiredParamsPresent": true,  
  
    "fulfillmentText": "We have sent the course syllabus and other relevant details to you via  
      email. An email has been sent to the Support Team with your contact information, you'll  
      be contacted soon. Do you have further queries?",  
  
    "fulfillmentMessages": [  
  
      {  
  
        "text": {  
  
          "text": [  
  
            "We have sent the course syllabus and other relevant details to you via email. An email  
            has been sent to the Support Team with your contact information, you'll be contacted  
            soon. Do you have further queries?"  
  
          ]  
  
        }  
  
      }  
  
    ],  
  
  },  
  
}
```

```
"intent": {  
  "name": "projects/faq-goilum/agent/intents/64b7ea04-23a9-4a5a-bcc5-08e971134d9c",  
  "displayName": "course_selection"  
},  
"intentDetectionConfidence": 1,  
"diagnosticInfo": {  
  "webhook_latency_ms": 4731  
},  
"languageCode": "en"  
},  
"webhookStatus": {  
  "message": "Webhook execution successful"  
}  
}
```

The Request above gives you the idea of how to parse the requests in your webhook service.

We'll discuss the webhook code in detail during the session.

3.5 Knowledge base:

Knowledge connectors complement defined intents. They parse *knowledge documents* (for example, FAQs or articles) to find automated responses. To configure them, you define one or more knowledge bases, which are collections of knowledge documents.

For this project, we have created a knowledge document which contains all the FAQs of the students and we have integrated the same with the bot.

3.5.1 Creating a Knowledge base

- a) To create a knowledge base, click on 'Knowledge' on the left hand side of your console and then click 'CREATE KNOWLEDGE BASE'.

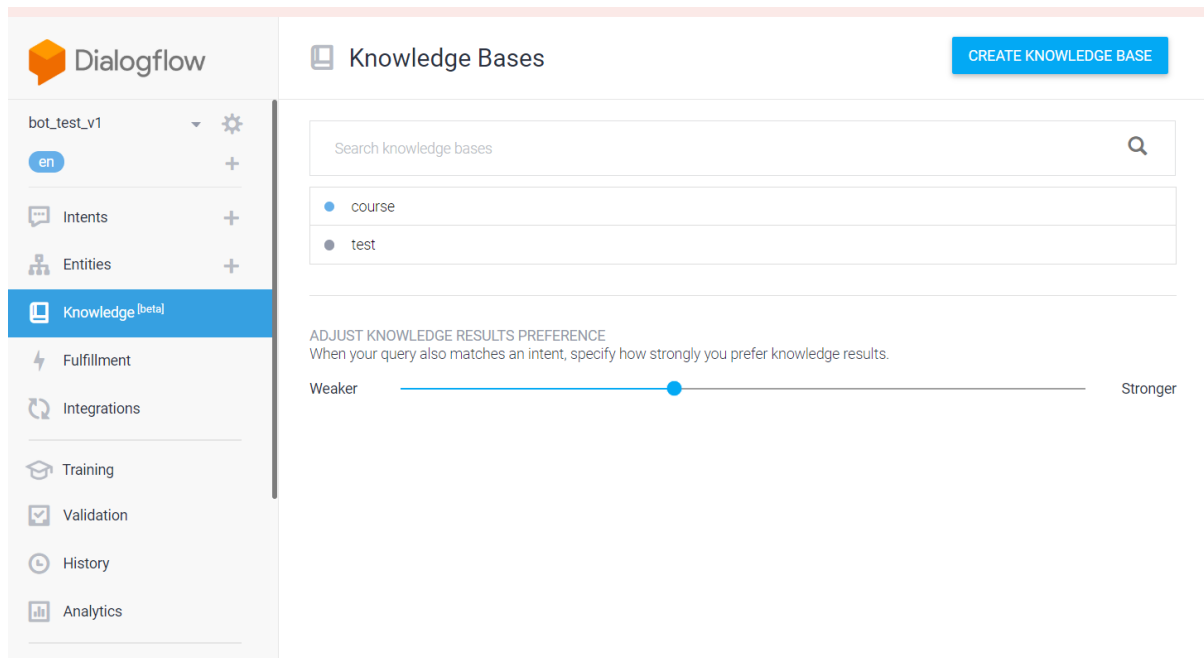


Figure 12 : Knowledge base is created by clicking on Create knowledge base.

- b) Give the name of your knowledge base and click 'Save'.
- c) Once created, you'll see a message stating that no knowledge document has been created yet. Click 'create the first one' to start creating your document.

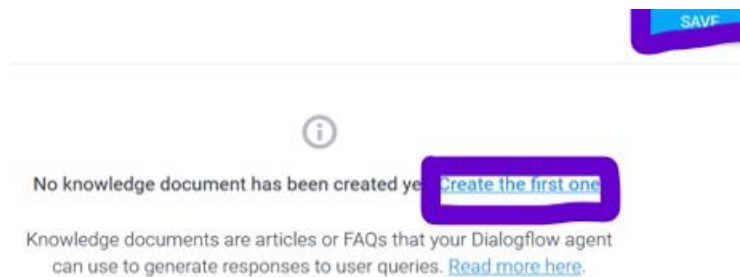
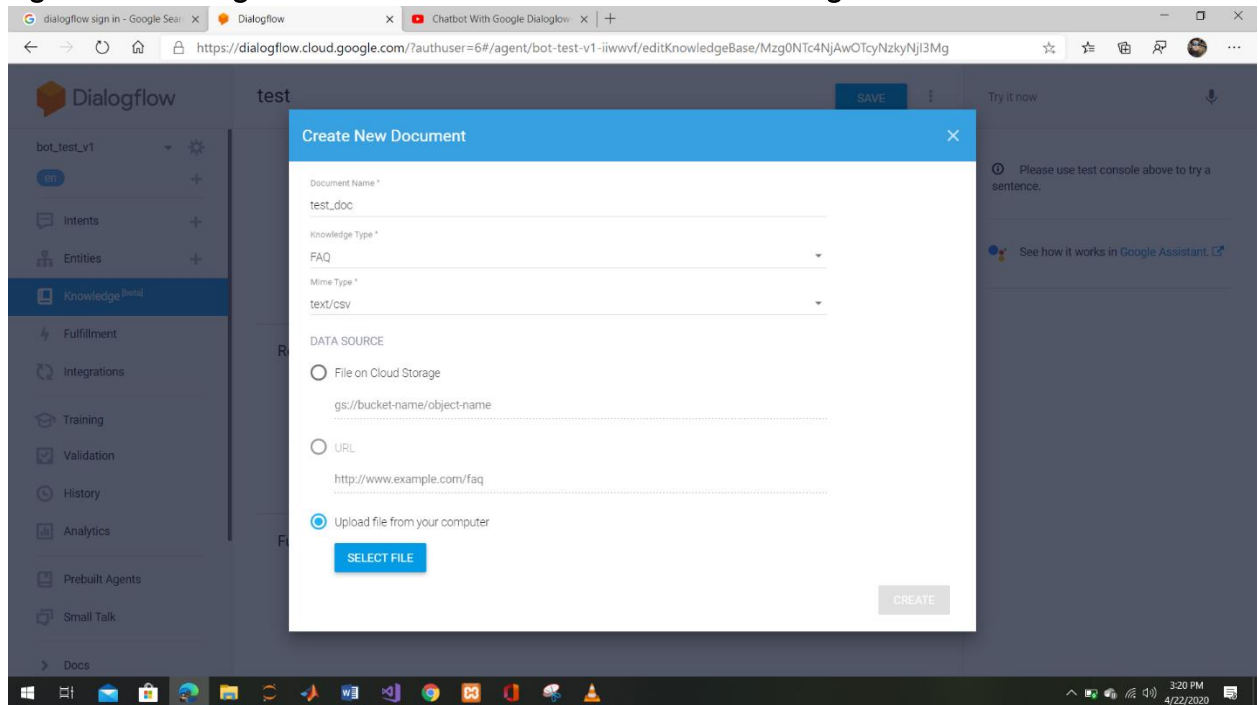


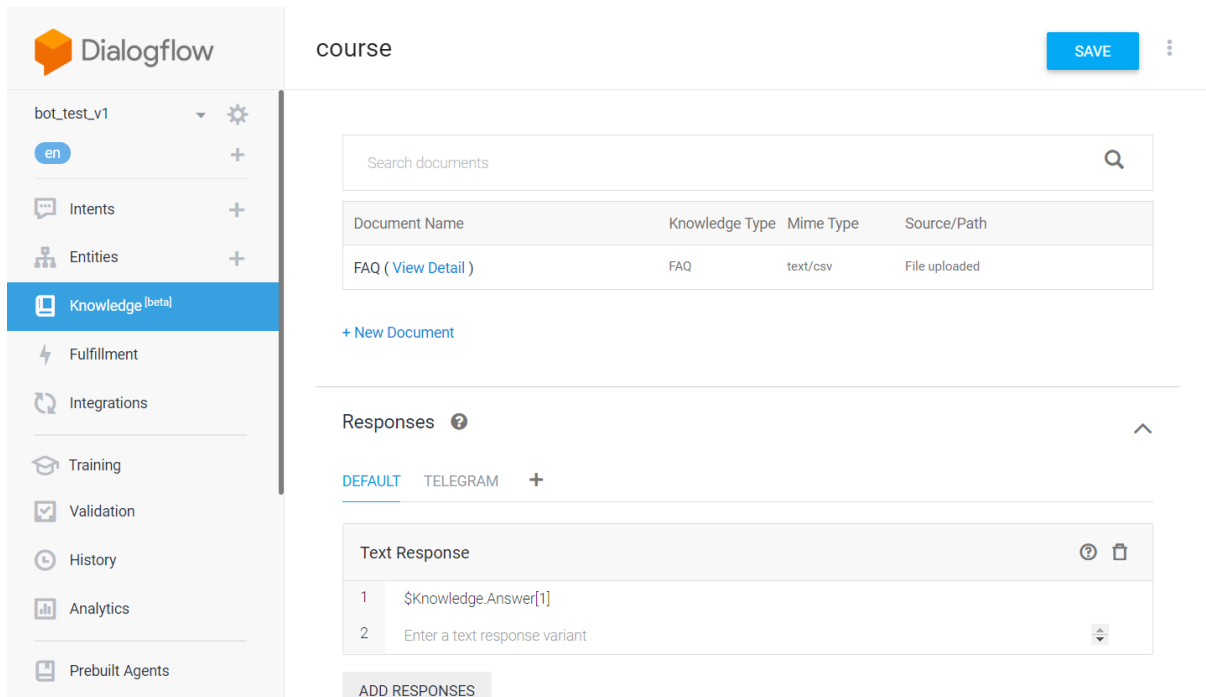
Figure 13: Creating knowledge document and saving it.

- d) Select FAQ and text/CSV and then upload the FAQ CSV file created and then click Create.

Figure 14: Entering the details in the new Document for Knowledge Base.



- e) Scroll down to the Responses section and add responses as desired:
- When defining the first response for the intent, use `$Knowledge.Question[1]` and `Knowledge.Answer[1]` where you want the question and answer to be supplied.
 - The index for `Knowledge.Question` and `Knowledge`.



- Answer starts at 1, so increase this index when adding more responses.

Figure 15: Saving the text response for the Knowledge Base.

f) **Click SAVE once you are done editing.**

3.6 Test the Agent:

You can now test the agent from the 'test it now' section.

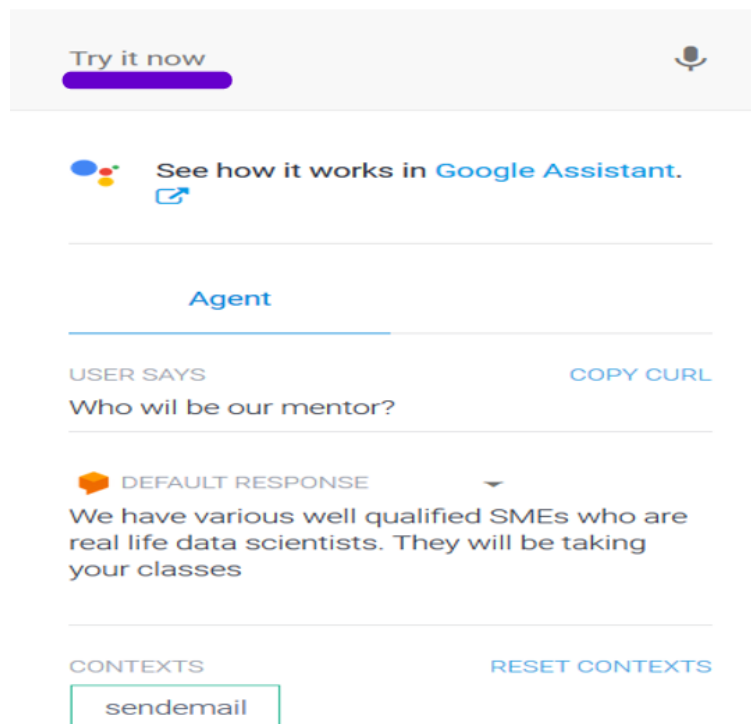


Figure 16: Testing the agent.

3.7 Integration:

Dialogflow integrates with many popular conversation platforms like Google Assistant, Slack, and Facebook Messenger. If you want to build an agent for one of these platforms, you should use one of the many *integrations* options. Direct end-user interactions are handled for you, so you can focus on building your agent.

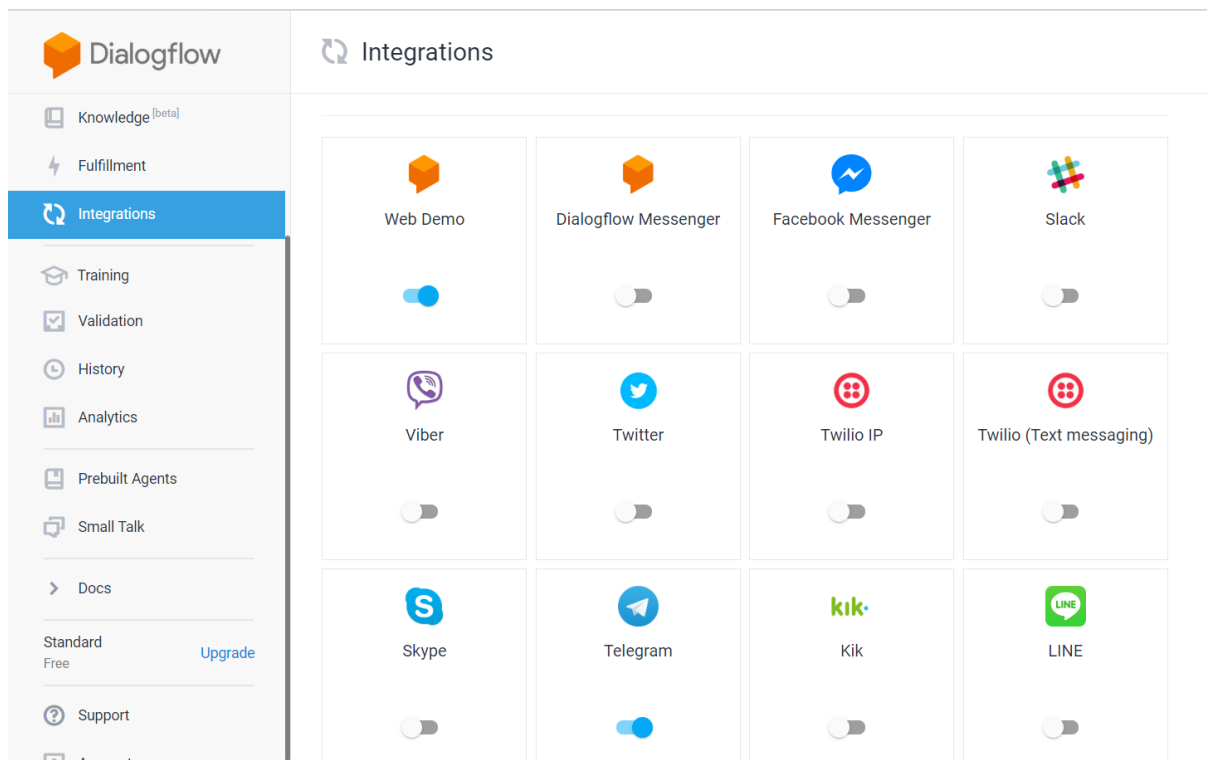


Figure 17: Intergration with other platforms.

3.7.1 Integration with Web Demo:

The following link directs the user to html.

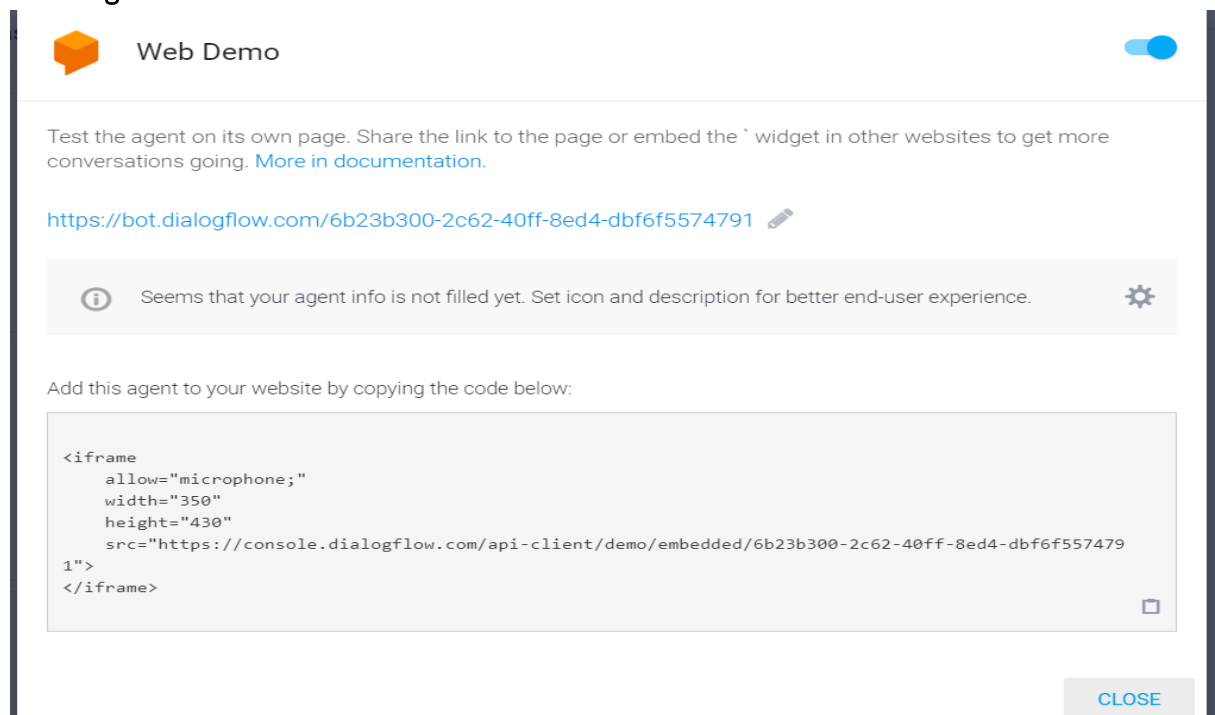


Figure 18: Web Demo intergration.

3.7.2 Integration with Telegram:

Dialogflow Telegram Integration allows you to easily create Telegram bots with natural language understanding based on the Dialogflow technology.

In order to set up the Telegram integration for your agent, you'll need the following:

- a) Telegram account.

3.7.1.2 Creating a Bot in Telegram

- a) Login to Telegram and go to <https://telegram.me/botfather>
- b) Click the Start button in the web interface or type /start
- c) Click on or type /newbot and enter a name
- d) Enter a username for the bot, ending in "bot" (e.g. garthswetherbot)
- e) Copy the generated access token

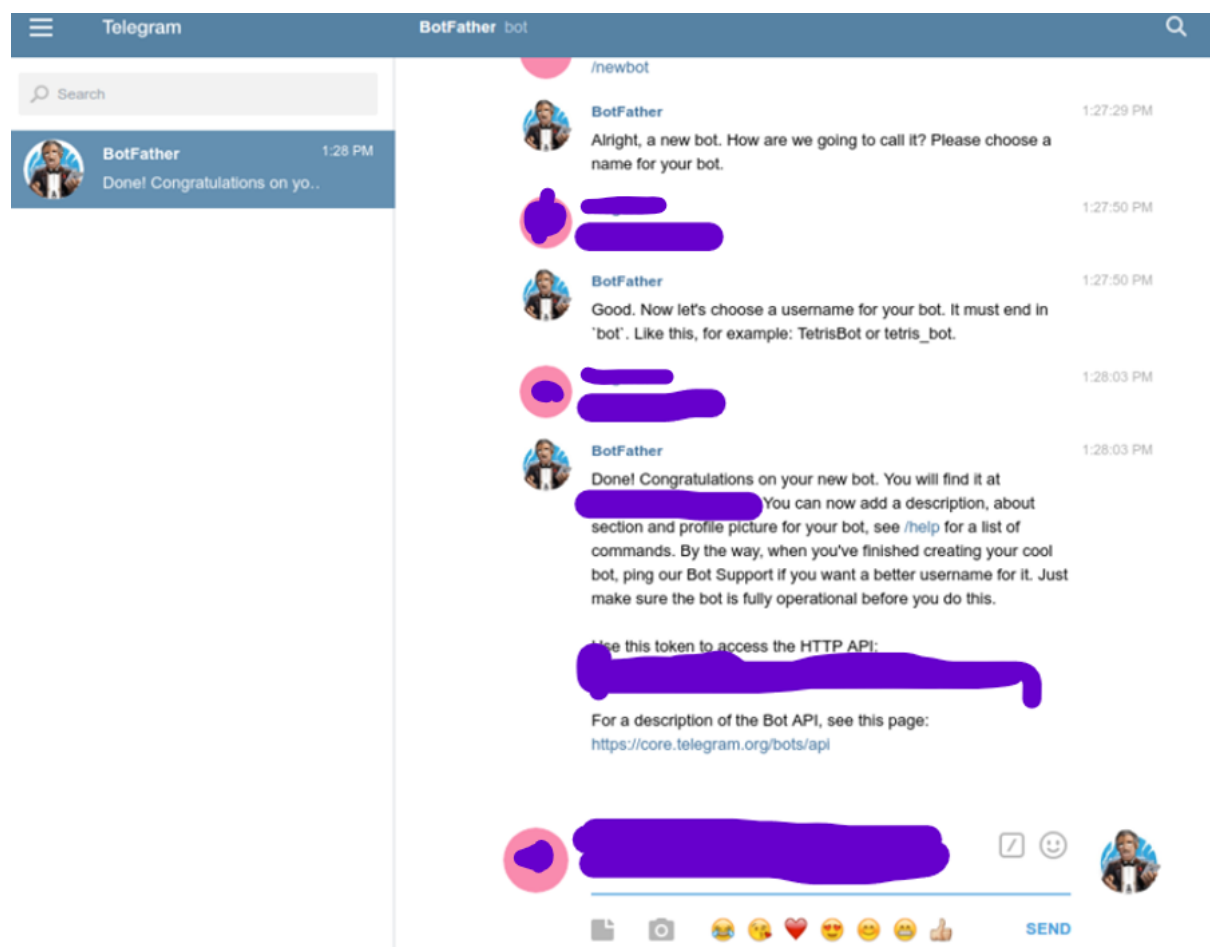
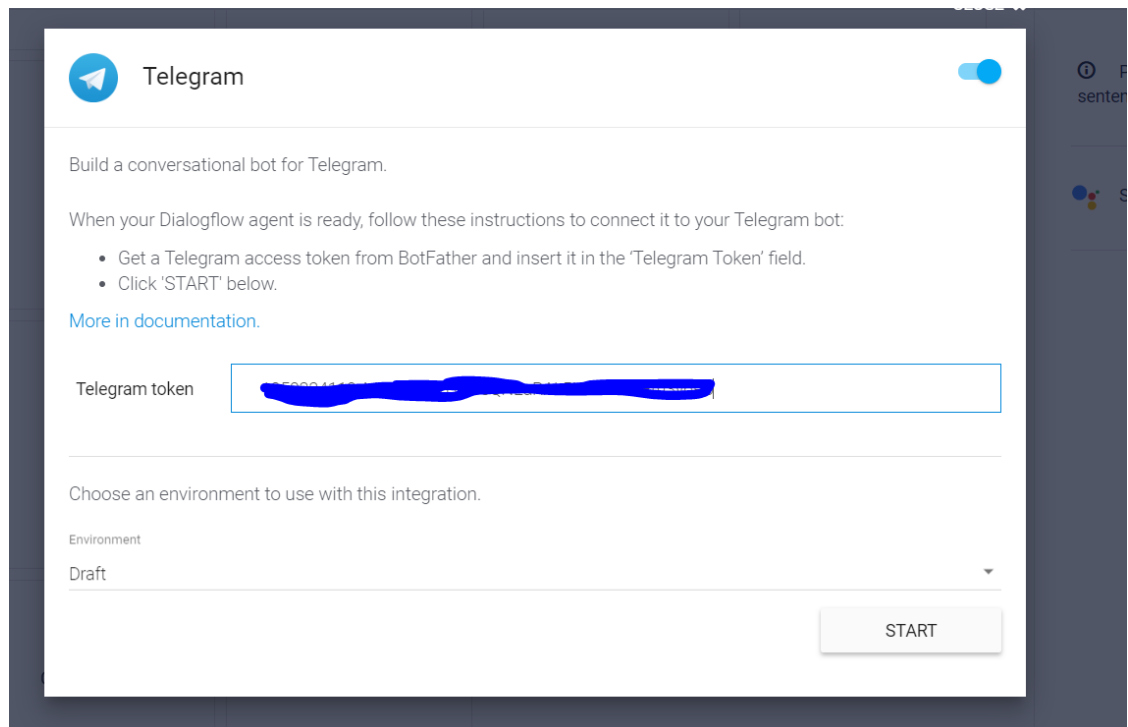


Figure 19: Creation of Bot in Telegram using the generated access token.

3.7.1.3 Setting Up Dialogflow

- a) **In Dialogflow, go to Integrations in the left hand menu**
- b) **Click on the Telegram tile**
- c) **Paste the Access Token into the related field**
- d) **Click the Start button**



The screenshot shows the 'Telegram' integration setup page in Dialogflow. At the top, there's a Telegram logo and a toggle switch. Below this, instructions are provided: 'Build a conversational bot for Telegram.' and 'When your Dialogflow agent is ready, follow these instructions to connect it to your Telegram bot:'. Two bullet points follow: 'Get a Telegram access token from BotFather and insert it in the 'Telegram Token' field.' and 'Click 'START' below.' A link 'More in documentation.' is also present. A text input field labeled 'Telegram token' contains a blue-redacted token. Below this, a section titled 'Choose an environment to use with this integration.' shows a dropdown menu with 'Environment' as the label and 'Draft' as the selected option. A 'START' button is located at the bottom right of the form.

Figure 20: DialogFlow intergration with Telegram.

e) Now you can search your bot in telegram and you can chat with it.

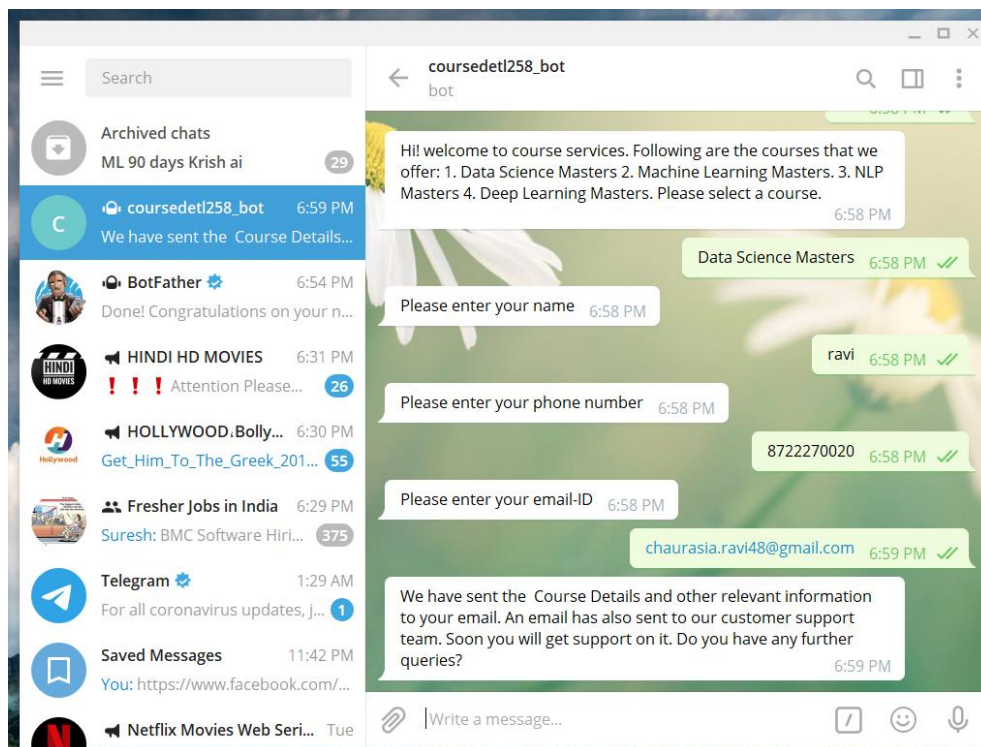


Figure 21: Testing the integrated Telegram ChatBot with DialogFlow .

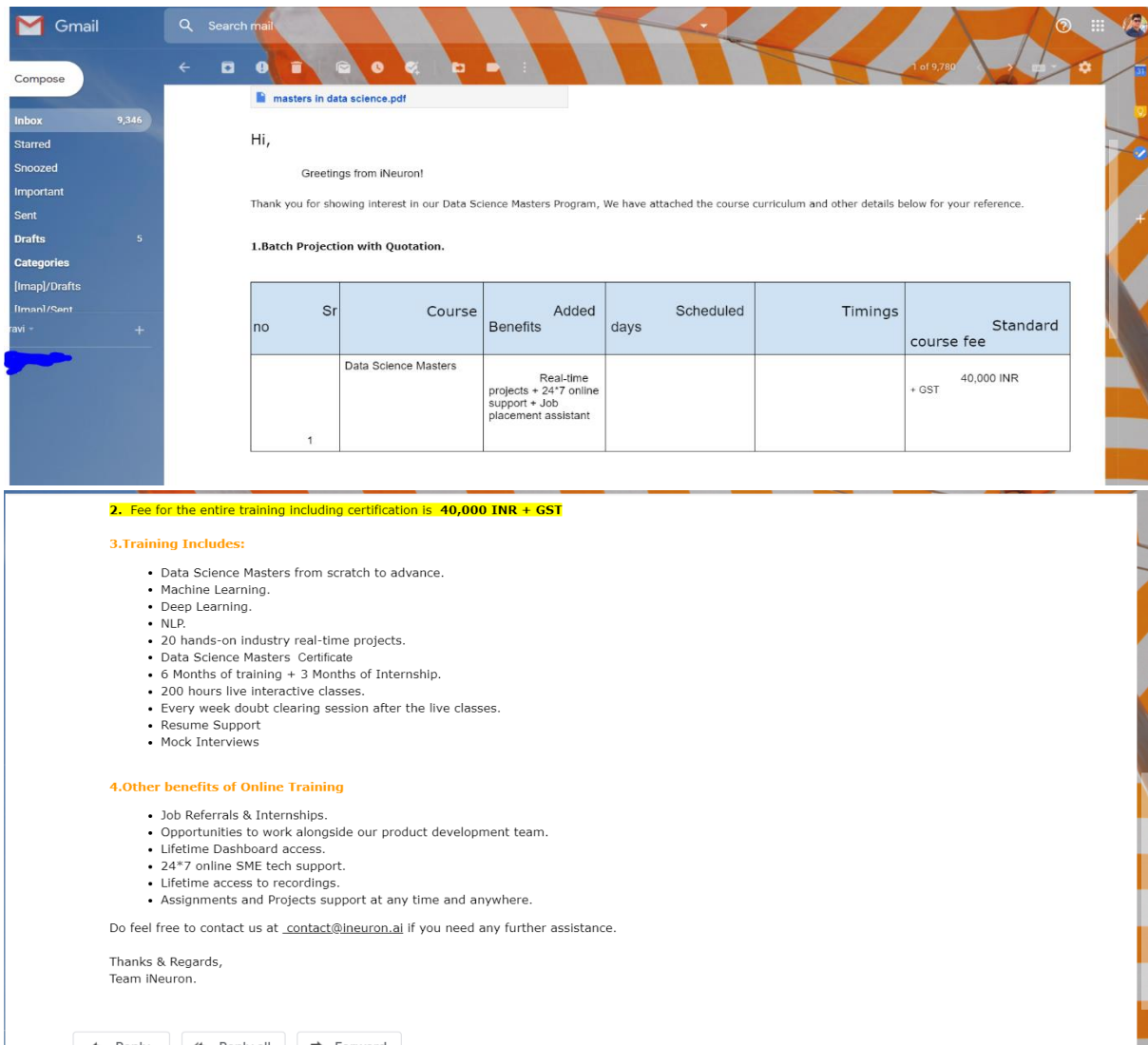


Figure 22,23: E-mail sent to student regarding course details.

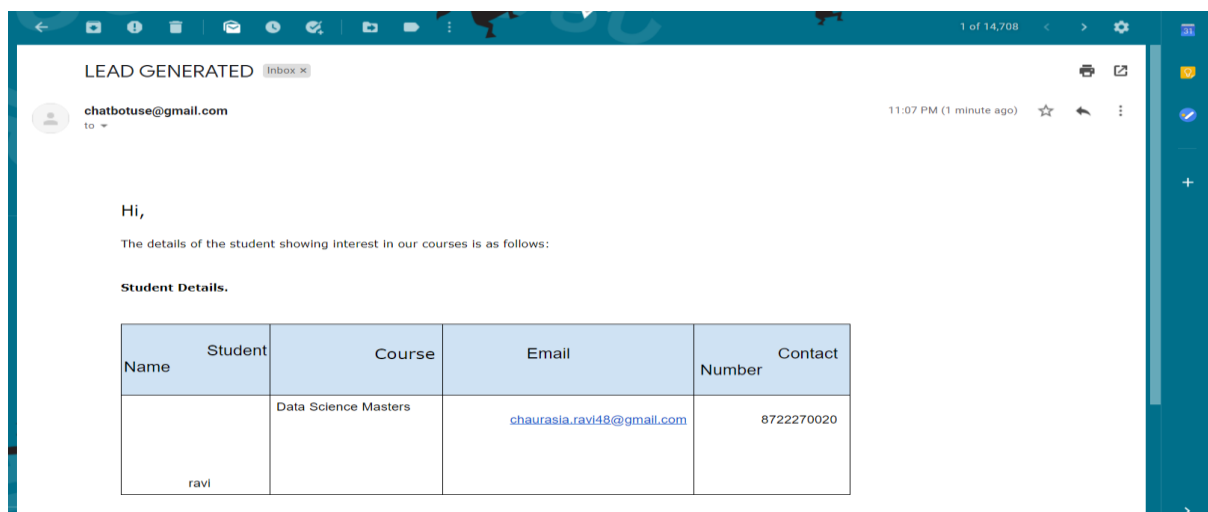


Figure 24: E-mail sent to customer support regarding student details.

Thank for Watch