Creating chatbots with Google Dialogflow

1. **Chatbot Fundamentals**
   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. **Uses of chatbots:**
3. **Can be used to answer FAQs.**
4. **Can be used for grievance handling.**
5. **Internal organizational automation.**
6. **To do the flight, hotel, appoint bookings etc.**
7. **Can guide customers to buy the correct product by answering their questions**
8. **Can be used for Customer Relationship Management.**
   1. **Advantages of Chatbots**
9. **24\*7 customer support.**
10. **Uniform customer experience.**
11. **Cost-efficient.**
12. **Build once and deploy everywhere.**
13. **Integration with various channels and platforms**
14. **Better monitoring and insight generation.**
    1. **Frameworks Present in the Market:**
15. **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.**
16. **Microsoft Bot Builder with LUIS: Azure Bot Service enables you to build intelligent, enterprise-grade bots with ownership and control of your data.**
17. **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.**
18. **RASA: Rasa provides infrastructure & tools necessary for high-performing, resilient, proprietary contextual assistants that work.**
19. **Wit.ai (Facebook):** **Wit.ai makes it easy for developers to build applications and devices that you can talk or text to.**
20. **Google Dialogflow**
    1. **Introduction:**

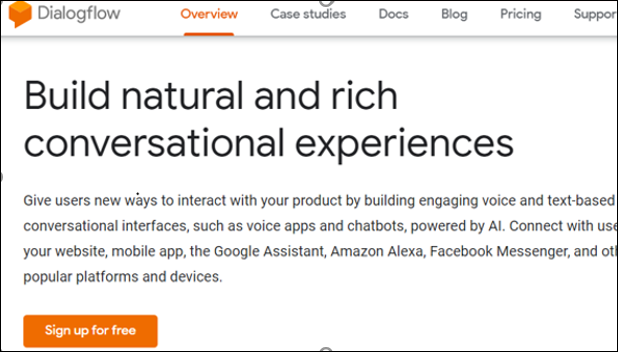
**Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on. Using Dialogflow, you can provide new and engaging ways for users to interact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

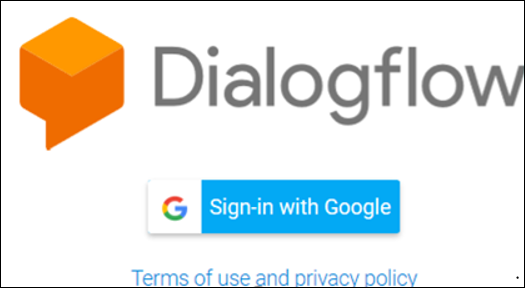
* 1. **Signup For Dialogflow:**

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

1. **Go to** <https://dialogflow.com/> **and click on ‘Sign Up for Free’ button.**



1. **Click ‘sign-in with Google’.**



1. **Select your google account and once you are redirected, click on ‘Go To Console’ on the upper right corner of the screen.**
   1. **Dialogflow Console:**

**Dialogflow provides a web user interface called the *Dialogflow Console*** ([open console](https://dialogflow.cloud.google.com/)). **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.**

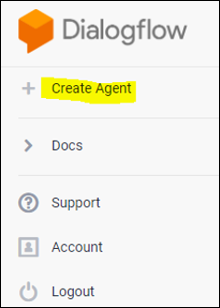
1. **Building a chatbot using google Dialogflow**
   1. **The problem statement:**

**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 futher with their queries.**

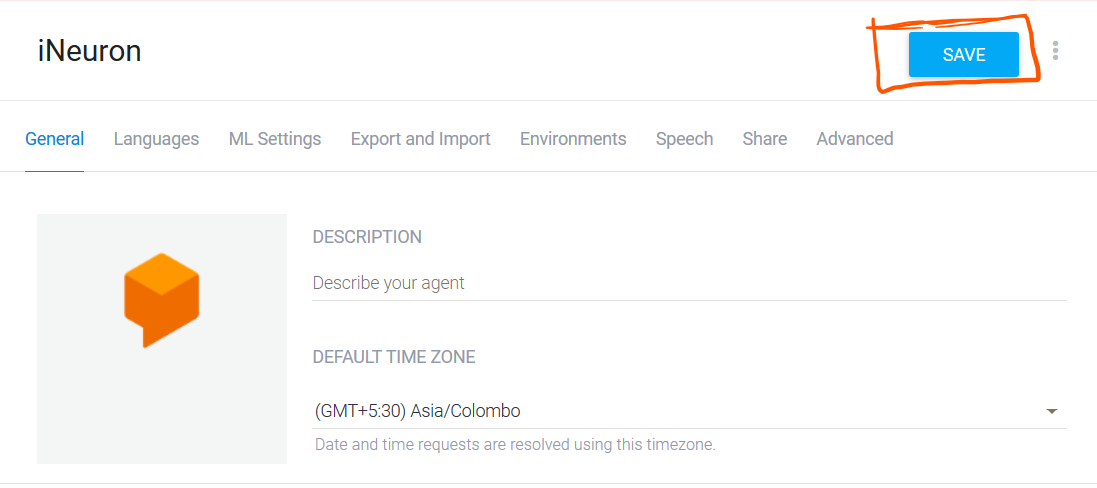
* 1. **Agent:**

**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.**

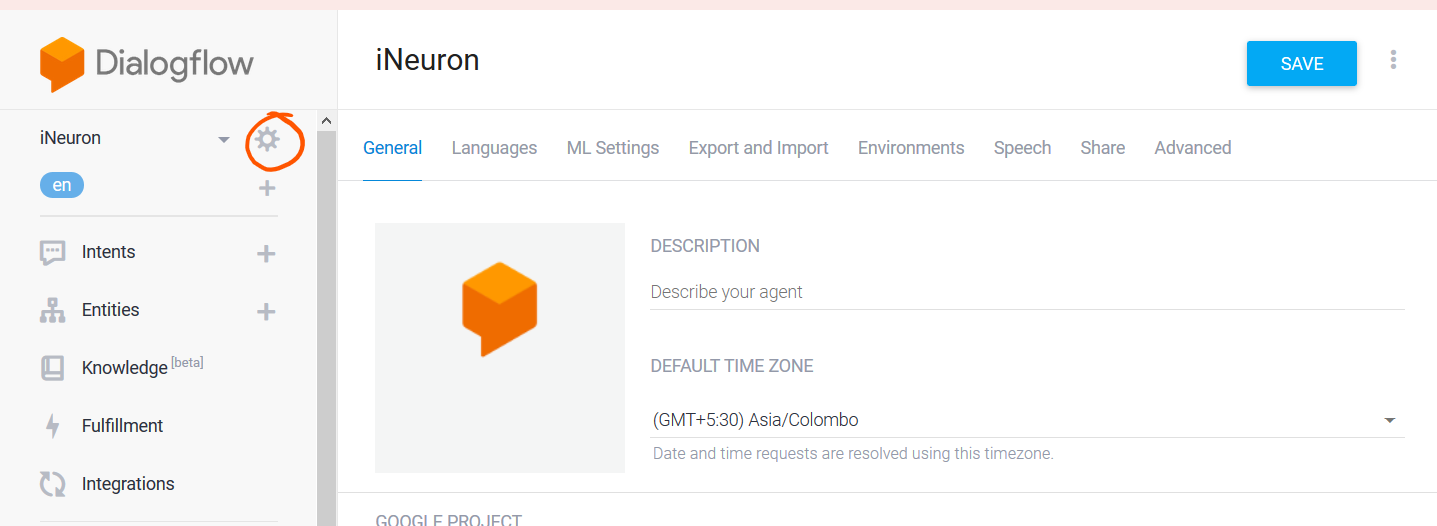
* + 1. **Creating an Agent:**
  1. **Click on *Create Agent* from the left menu.**



* 1. **Provide the name of the agent and click on the *SAVE* button to create the Agent.**

****

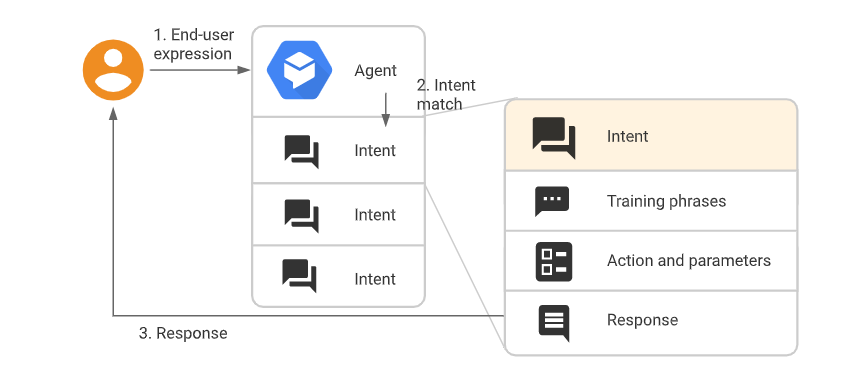
* 1. **After saving, the agent is shown in the left hand side of your console. You can click the gear icon (highilighted in orange) to edit the agent settings.**

****

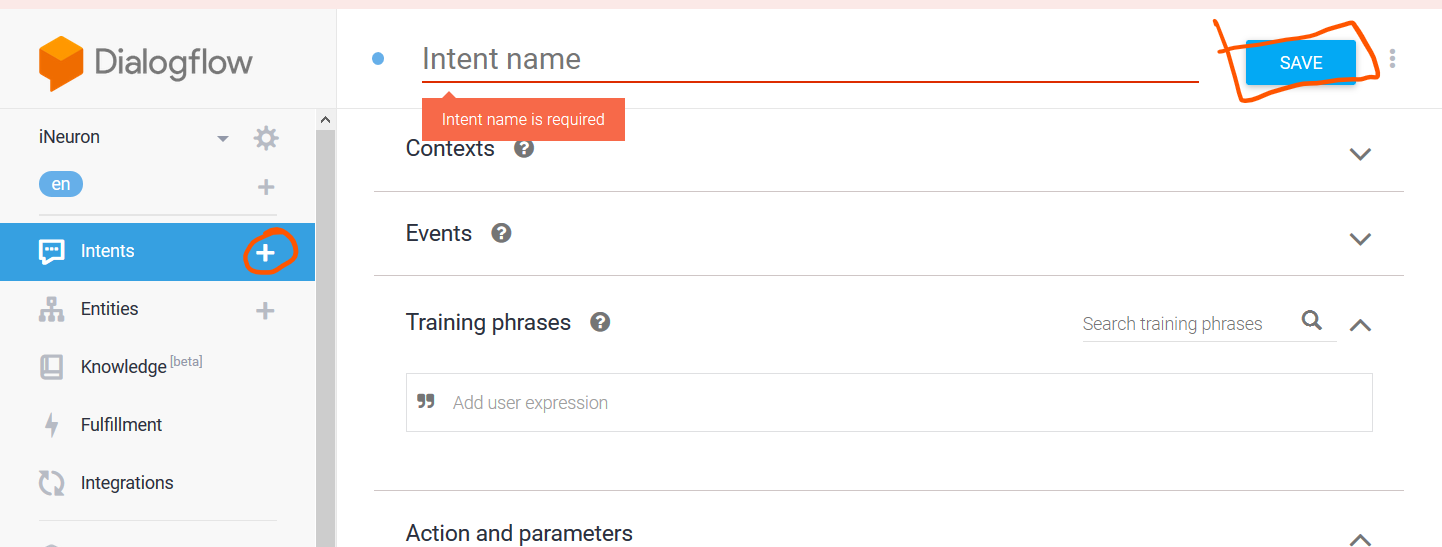
* 1. **Intent:**

**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:**

****

* + 1. **Creating an Intent:**
    2. **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**.**

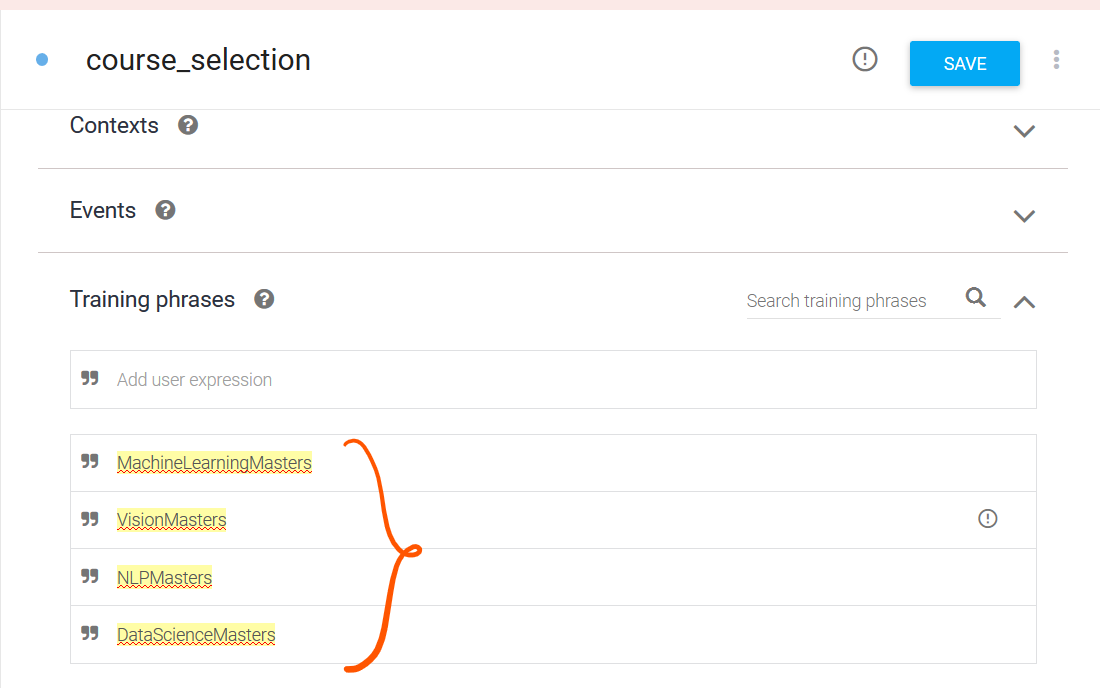
****

* + - 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:**

1. **Click the text field that shows "Add user expression".**
2. **Type your training phrases and press the Enter key after each.**

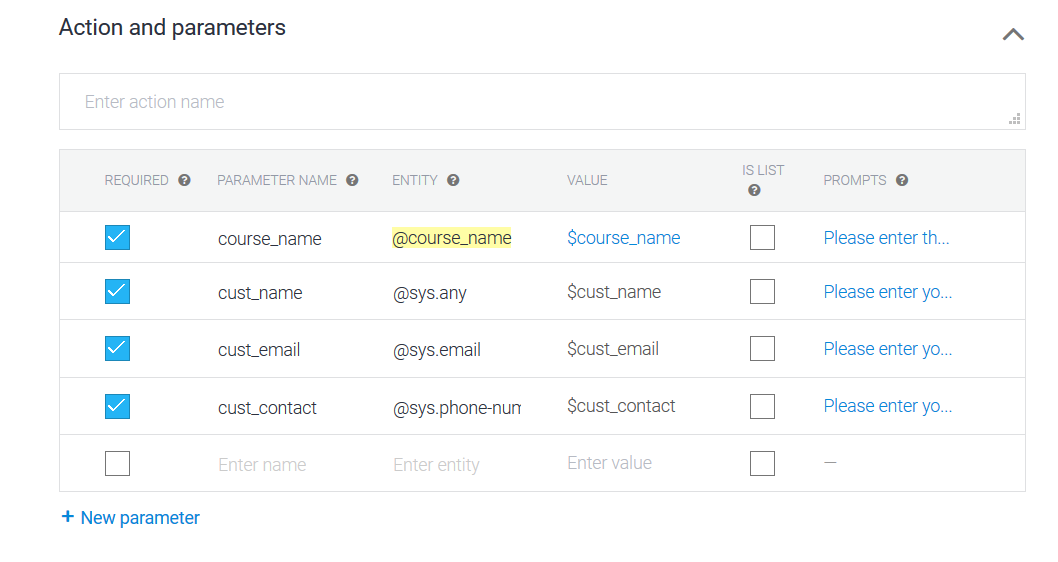
****

* + - 1. **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**](https://cloud.google.com/dialogflow/docs/entities-overview)**, 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.**

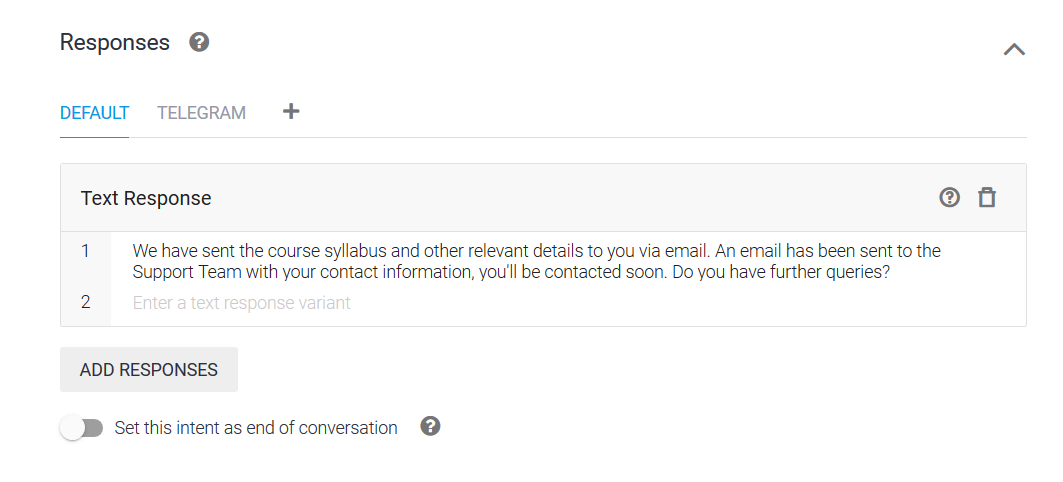
**Each intent**[**parameter**](https://cloud.google.com/dialogflow/docs/intents-actions-parameters#params)**has a type, called the *entity type*, which dictates exactly how data from an end-user expression is extracted.**

**Dialogflow provides predefined**[**system entities**](https://cloud.google.com/dialogflow/docs/entities-system)**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**](https://cloud.google.com/dialogflow/docs/entities-custom)**for matching custom data.**



* + - 1. **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**](https://cloud.google.com/dialogflow/docs/intents-actions-parameters#values)**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".**

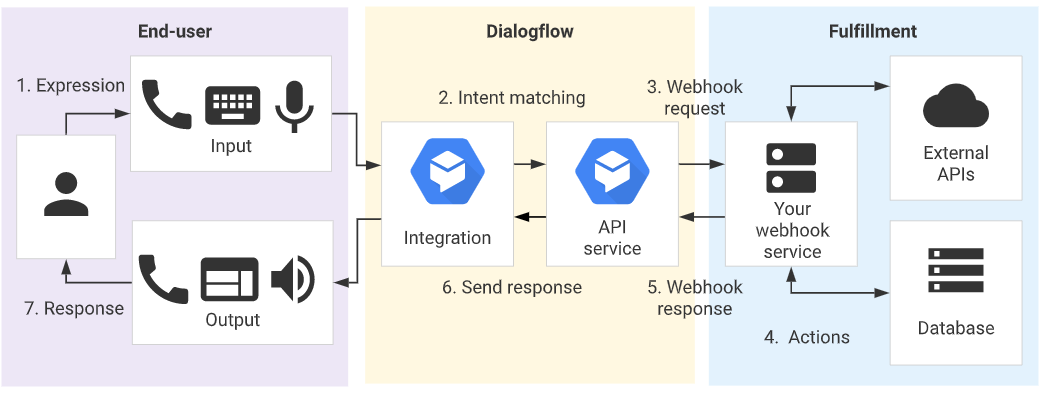
****

* 1. **Fulfillment:**

**By default, your agent responds to a matched intent with a static response. If you're using one of the**[**integration**](https://cloud.google.com/dialogflow/docs/integrations)**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.**

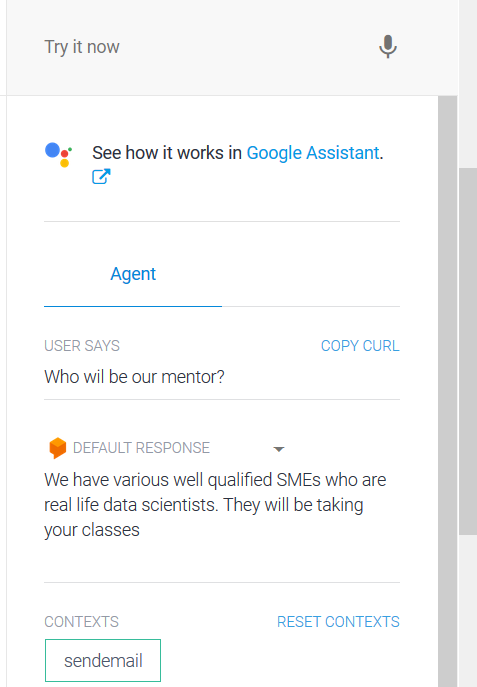
**Each**[**intent**](https://cloud.google.com/dialogflow/docs/intents-overview)**has a setting to enable fulfillment. If an intent requires some action by your system or a dynamic response, you should enable fulfillment for the intent. If an intent without fulfillment enabled is matched, Dialogflow uses 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.**

****

1. **The end-user types or speaks an expression.**
2. **Dialogflow matches the end-user expression to an intent and extracts parameters.**
3. **Dialogflow sends a**[**webhook request**](https://cloud.google.com/dialogflow/docs/fulfillment-webhook#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.**
4. **Your service performs actions as needed, like database queries or external API calls.**
5. **Your service sends a**[**webhook response**](https://cloud.google.com/dialogflow/docs/fulfillment-webhook#webhook_response)**message to Dialogflow. This message contains the response that should be sent to the end-user.**
6. **Dialogflow sends the response to the end-user.**
7. **The end-user sees or hears the response.**
   * 1. **Test the Agent:**

**You can now test the agent from the ‘test it now’ section.**

****