

Chatbot PoC Walk-thru

Buffet Recommender Chatbot

Neo Group Ltd

AUG 2020



Agenda

Agenda

1. Buffet Recommender Chatbot demo
2. Basic Concepts in Dialogflow
3. Architecture
4. Setup Test/Dev Environment – Dialogflow, Kommunicate.io and Firebase
5. Code Walk-thru – Dialogflow Fulfilment Webhook Client
6. Run it on your own environment

Buffet Recommender Bot Demo



Dialogflow Basics

What is Dialogflow?

- An AI-powered cloud service for NLP - Natural Language Processing
- Owned by Google (thus the tight integration with Google Cloud Platform)
- <https://www.youtube.com/watch?v=yT58gTXdQb8>

Dialogflow - Basic Concepts

Agent

Fulfilment

Integrations

Webhook

Intents

Entities

Context

Agent

Virtual

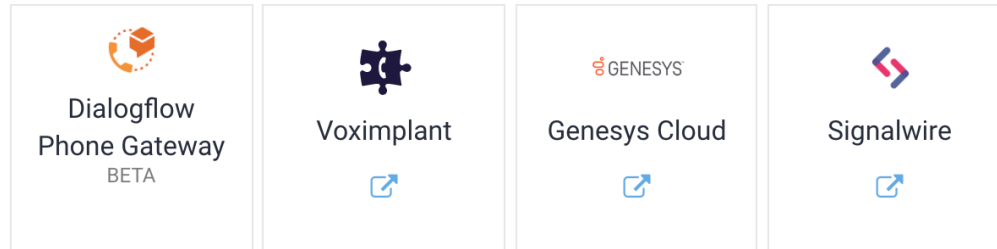
Your AI-powered Bot

Handles all your end-users' conversations

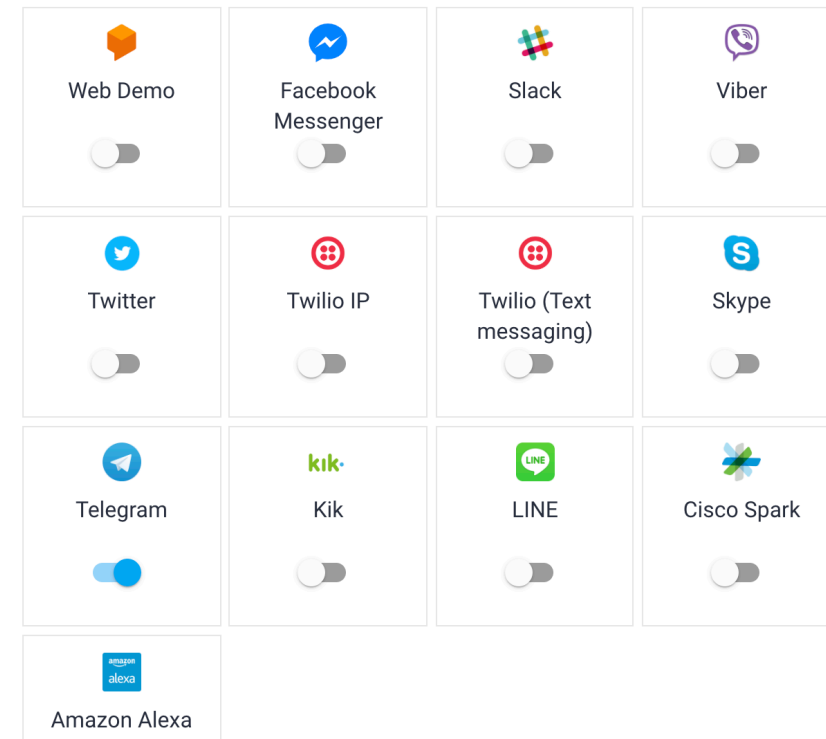
Your Dialogflow project

Dialogflow – Frontend Integrations with Client-Facing Apps

Telephony



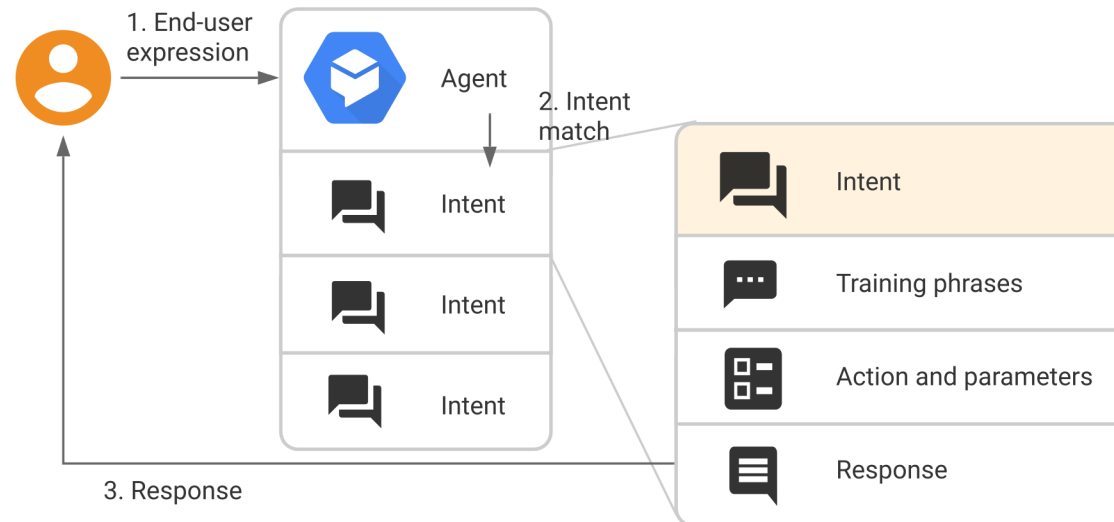
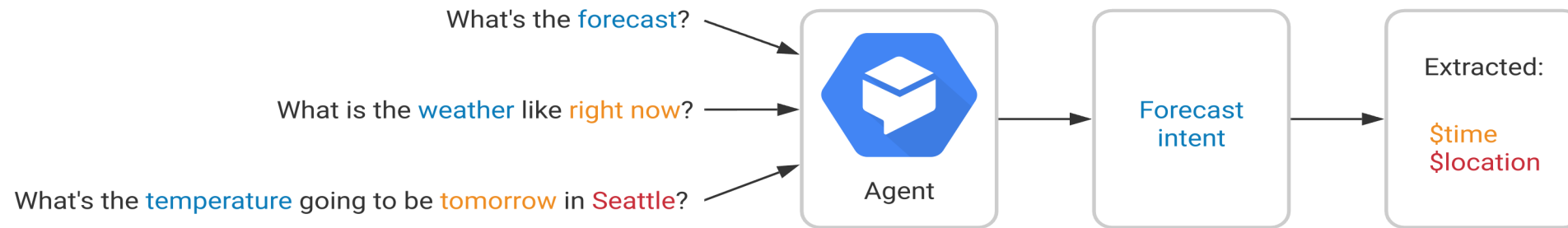
Text based



- For the PoC, the frontend integration is using *Kommunicate.io*
- More info: <https://www.kommunicate.io/product/dialogflow-integration>

Intents

Categorize/define an end-users' intention – for one conversational turn



Entities

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

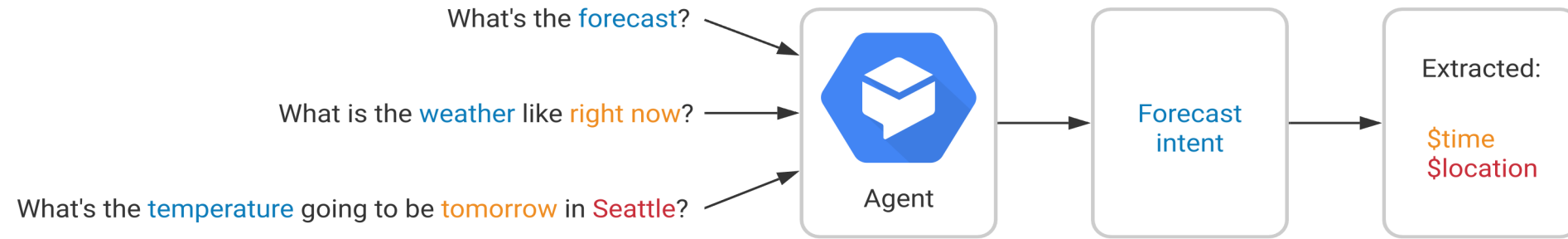
- System Entities – ready-made by Dialogflow
- Custom Entity – You create your own eg. MenuType, which can refer to “Bento”, “Mini Buffet”
- Session Entities – Short-lived entities created programmatically during runtime

Example:

” I want to order 6 pancakes to be delivered tomorrow

PARAMETER NAME	ENTITY	RESOLVED VALUE
number	@sys.number	6
date-time	@sys.date-time	tomorrow

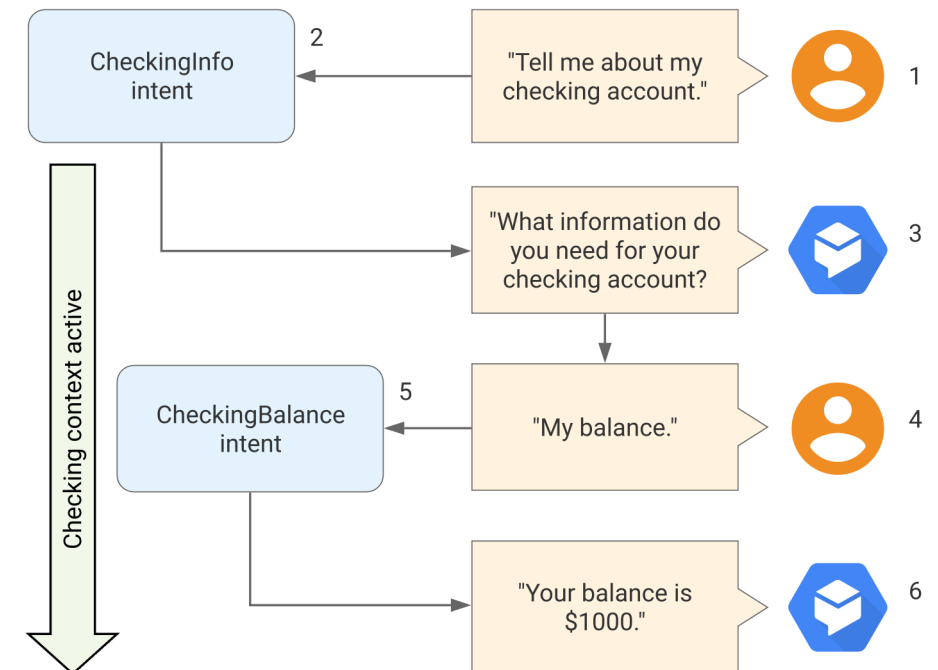
Context



NLP AI engine that can **Detect Entities** such as names, dates, duration, countries, addresses, locations, even sentiments etc. from a user's **utterance or expression**

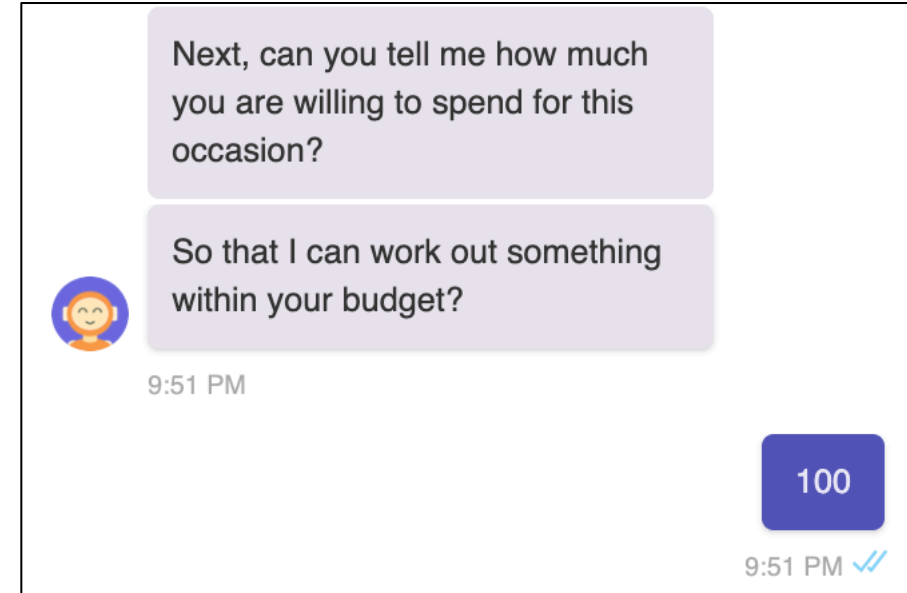
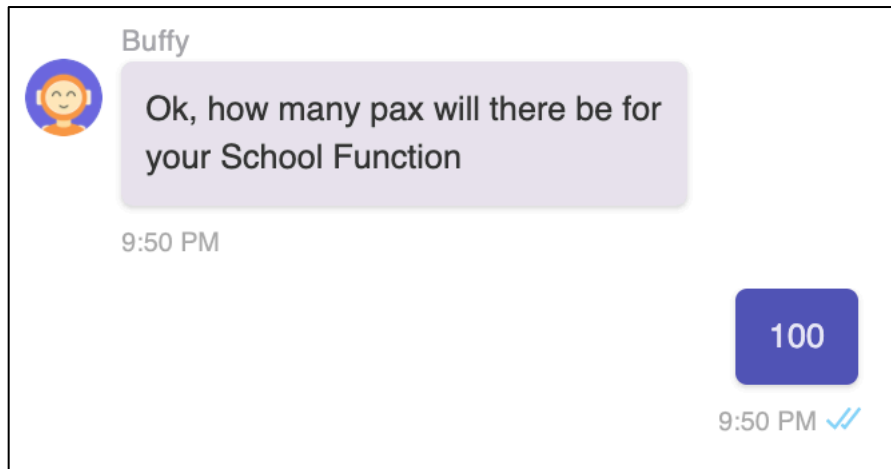
Detected Entities are matched to your **Intents**

A matched Intent sets the current **Context**. You can think of Context as a means of “remembering **Parameters**”



Context

The end-user entered “100” as response for these 2 Intents.
Both contexts are different.



Fulfilment

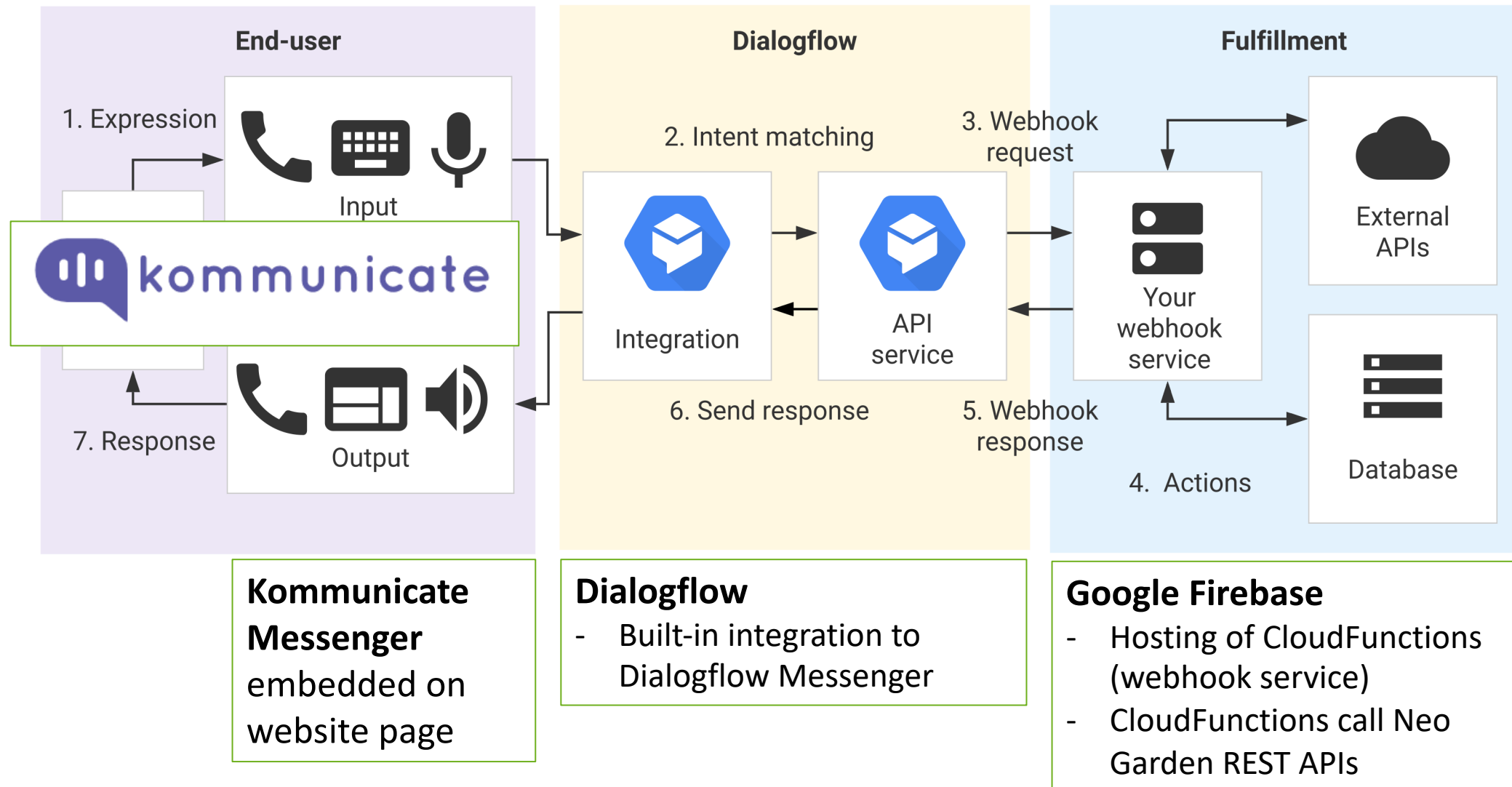
Once an Intent is matched, how should your agent respond?

1. Static Response
2. Webhook Call
 - POST to a web service
 - Inline Editor deployed to Cloud Functions

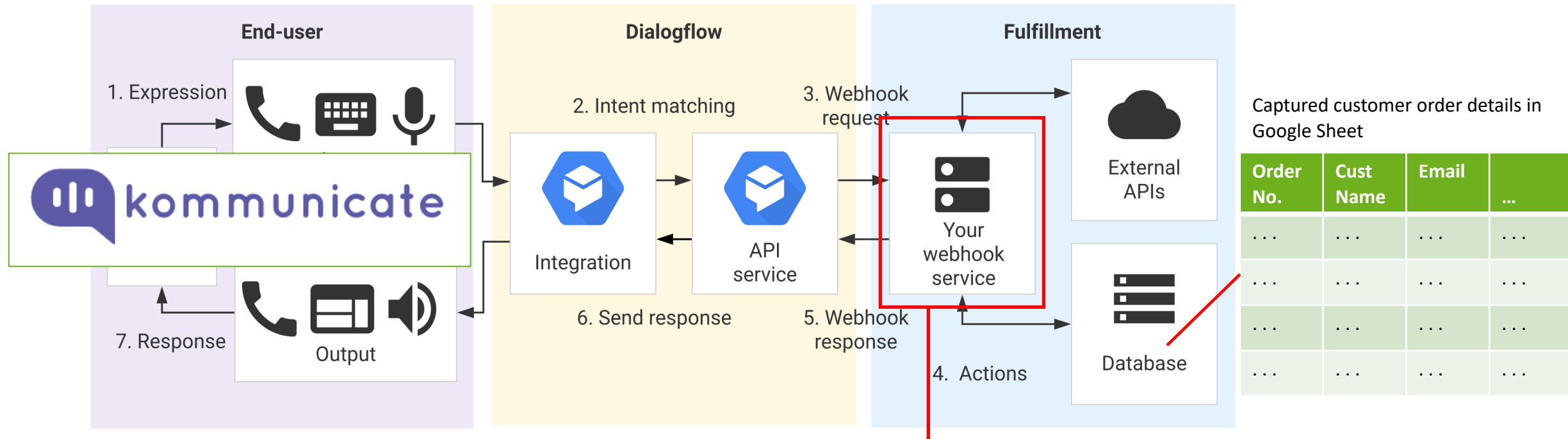


Architecture

Architecture – Cloud Services



Architecture – Development/Custom Work



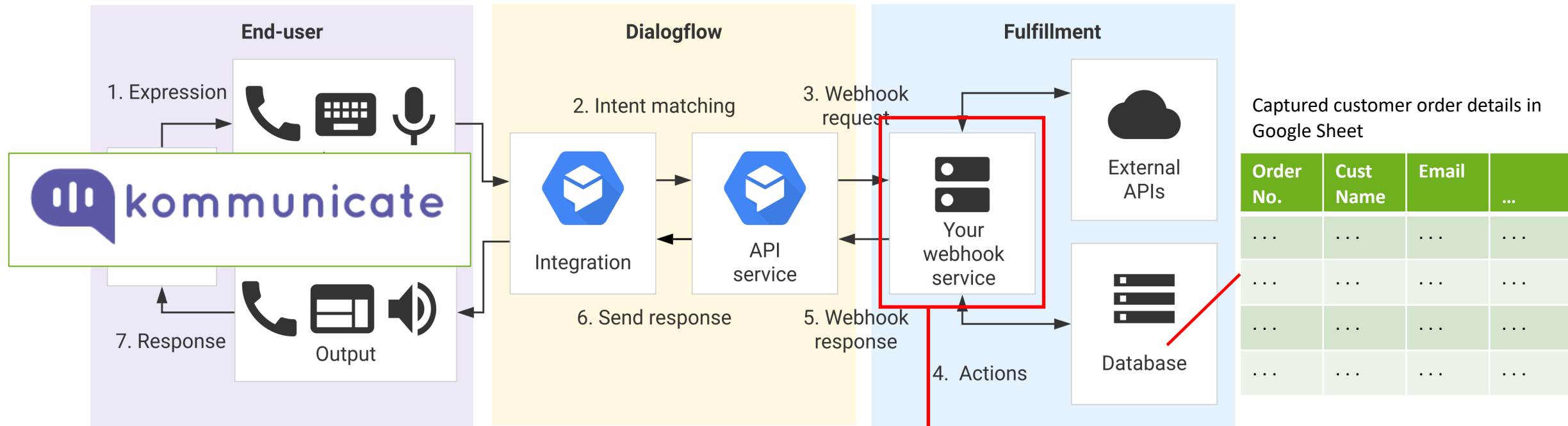
Dialogflow Console:

- Intents:
 - Training phrases
 - Responses
 - Parameters
- Entities:
 - Menus, dishes etc.

Node JS deployed as Google CloudFunction

- Write the webhook code
 - Map Intents to functions
 - Extract parameters from Intent
 - Calls Neo Garden BCMS APIs
 - Responds to the user (via Dialogflow Fulfilment payload)
- **Deploy** webhook via Inline Editor in Dialogflow console

Architecture – Development/Custom Work



Dialogflow Console:

- Intents:
 - Training phrases
 - Responses
 - Parameters
- Entities:
 - Menus, dishes etc.

Node JS deployed as Google CloudFunction

- Install node.js on workstation *
- Install Firebase CLI tools *
- **Code** the Cloudfunctions:
 - Map Intents to functions
 - Extract parameters from Intent
 - Calls Neo Garden BCMS APIs
 - Responds to the user (via Dialogflow Fulfilment payload)
- **Deploy** the cloudfunctions to Firebase using the CLI *



Setup the PoC

Environment Setup – Dialogflow

1. Go to: <https://cloud.google.com/dialogflow/docs/quick/setup> , click on the

Go to the project selector page

1. In the project selector page, click “CREATE PROJECT”, give it a name

2. Enable API

Enable the API

3. Go back to “Quick Setup” page, click

Go to the Create Service Account Key page

4. A JSON file containing your Service Account private key will be downloaded, keep it safe.

Github Repository

- All the source code and resources for the PoC can be found here:
 - <https://github.com/khehhin/buffie>

Setup Dialogflow – PoC Intents and Entities

1. Download Buffie.zip from <https://github.com/khehhin/buffie/blob/master/Bufie.zip>
2. Go to: <https://cloud.google.com/dialogflow> , sign-in with your Gmail account
3. Go to your Dialogflow console: <https://console.dialogflow.com/>
4. Go to <https://cloud.google.com/dialogflow/docs/quick/build-agent> , follow the guide instructions to:
 - Create an Agent (give a personality name)
 - Restore from zip – Buffie.zip

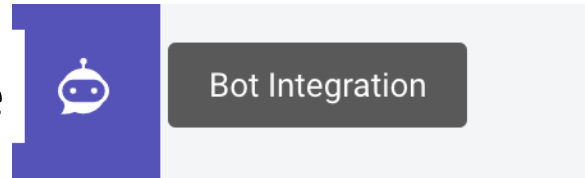
Setup Dialogflow – PoC Fulfilment Webhook

1. Go to Fulfilment, disable Webhook, enable Inline Editor (you may have to enable Google Billing)
2. Go to <https://github.com/khehhin/buffie/blob/master/functions/index.js> , copy/paste code into Inline Editor
3. Click Deploy

Setup Dialogflow – PoC Integration to Kommunicate.io

1. Go to <https://www.kommunicate.io/> , login or sign-up

2. On the dashboard, click the




3. On the Dialogflow card, click “INTEGRATE BOT”,

4. Follow the instructions on-screen to create a service account JSON key file.

- NOTE: In the Google Cloud Console, choose the “Dialogflow Integrations” service account.
- Leave “Human Handoff” disabled

5. Try out the bot from the dashboard

Kommunicate.io – Embed HTML Code

1. You can find the HTML code for the web chat interface by:
 - On the Kommunicate dashboard, go to  Settings -> Install -> Web



Buffet Recommender Bot

Intents–Webhook Functions Mapping

●	Budget
●	Cuisine
🔖	Default Fallback Intent
●	Default Welcome Intent
●	Dietary Restrictions
●	EventDateTime
●	Number of Pax
●	Occasion
●	Recommendation

```
593 let intentMap = new Map();
594 intentMap.set('Default Welcome Intent', welcome);
595 intentMap.set('Default Fallback Intent', fallback);
596 intentMap.set('Recommendation', occasion);
597 intentMap.set('Occasion', pax);
598 intentMap.set('Number of Pax', dietary);
599 intentMap.set('Dietary Restrictions', budget);
600 intentMap.set('Budget', whenDate);
601 intentMap.set('EventDateTime', cuisine);
602 intentMap.set('Cuisine', getMenu);
```

Entities

dietaryType	
<input checked="" type="checkbox"/> Define synonyms ?	<input type="checkbox"/>
vegetarian	
vegetarian only	
halal	
vegetarian and halal	
no restrictions	

occassionType	
<input checked="" type="checkbox"/> Define synonyms ?	<input type="checkbox"/> R
House Warming	
Baby Full Month	
Charity	
Birthday	
Corporate Function	
Praying	
Wedding	
100th Day Celebration	
Festival Celebration	
BBQ	
Church Function	
Gathering	
Funeral	
School Function	

menuType	
<input checked="" type="checkbox"/> Define synonyms ?	<input type="checkbox"/>
BBQ SIZZLE	
BENTO SET	
HEALTHIER CHOICE	
MINI BUFFET	
OFFICE SHARING COMBO	
PERANAKAN HERITAGE	
REGULAR BUFFET	
SEMINAR PACKAGE	
TEA RECEPTION	
VEGAN DELIGHT	

Retrieving Parameters

```
function budget(agent) {  
    diet = agent.parameters.dietaryType;  
  
    agent.add("Ok, you have specified " + diet + " dietary preference for you  
    agent.add(`Next, can you tell me how much you are willing to spend for th  
    agent.add(`So that I can work out something within your budget?`);  
}
```

Retrieving Form Data

```
function cuisine(agent){  
  let formData = request.body.originalDetectIntentRequest.payload.formData;  
  eventDate = formData["Event Date"];  
  eventTime = formData["Event Time"];
```

Additional Resources

Dialogflow documentation: <https://cloud.google.com/dialogflow/docs>

Dialogflow Fulfillment for Node JS: <https://github.com/dialogflow/dialogflow-fulfillment-nodejs>

Kommunicate.io general docs: <https://docs.kommunicate.io/>

Kommunicate.io for Dialogflow integration: <https://docs.kommunicate.io/docs/bot-dialogflow-integration>

Kommunicate Rich Messages: <https://docs.kommunicate.io/docs/message-types>

Dialogflow Github: <https://github.com/dialogflow>

Kommunicate.io Github: <https://github.com/Kommunicate-io>

Gerald's Github: <https://github.com/khehhin/buffie>

Order Capture

1. Allow user to browser dishes
 - a. User selects Menu
 - b. User selects MenuCategory
 - c. User browse dishes in MenuCategory in a Template List
2. Allow user to select dishes
 - a. User click “I’m ready to order”
 - b. User selects a MenuCategory
 - c. User select a dish in Dishes Category List
3. Submit User’s selected dishes via CreateOrder_V2 API

Key Challenges/Concepts to Prove

1. Recommendation of buffet Menus
2. Invoking BCMS APIs
3. Dynamically detect Menu names
4. Capture User's dish choices
 - a. Presenting MenuCategories and Dishes
 - b. Tracking the number of User's dish choices
5. Populate CreateOrder JSON with selected Menu info and selected dishes info
6. Submit CreateOrder JSON via CreateOrder_V2 API



REPUBLIC
POLYTECHNIC
CENTRE

Thank You



<https://www.rp.edu.sg/coi-scm/>