

## Código escrito en lenguaje python para actualizar el chatbot a partir de la información que hay en un documento de Google sheets

```
!pip install google-auth google-api-python-client #Instalamos las librerías necesarias
!pip install --upgrade google-cloud-dialogflow-cx
```

```
from google.colab import files
print("Por favor suba el archivo de credenciales de google sheets")
uploaded = files.upload()
print("Por favor suba el archivo de credenciales de dialogflow")
uploaded = files.upload()
```

```
#Este código borra todas las rutas que estén en la página "actualizar"
from google.oauth2 import service_account
from google.cloud.dialogflowcx.v3.services.pages import PagesClient
from google.cloud.dialogflowcx.v3.types.page import Page

# Configuración de Dialogflow CX
DIALOGFLOW_CX_CREDENTIALS = "/content/proyecto-final-379916-a58d7bc22b78.json" # Se usa el archivo de credenciales subido anteriormente
DIALOGFLOW_CX_PROJECT_ID = 'proyecto-final-379916' # Nombre del proyecto
DIALOGFLOW_CX_LOCATION = 'us-central1' # Ubicación del proyecto
DIALOGFLOW_CX_AGENT_ID = '2542e633-3551-4551-9862-c1c95acc71b7' # ID del agente
DIALOGFLOW_CX_FLOW_ID = '00000000-0000-0000-0000-000000000000' # ID del flujo que contiene la página
DIALOGFLOW_CX_PAGE_ID = 'f603290a-5278-4775-b40e-9b710db103d6' # ID de la página "actualizar" a la que se quiere acceder

# Autenticación con Dialogflow CX
credentials = service_account.Credentials.from_service_account_file(DIALOGFLOW_CX_CREDENTIALS)
pages_client = PagesClient(credentials=credentials, client_options={"api_endpoint": "us-central1-dialogflow.googleapis.com"})

# Construir el nombre de la página a partir de los valores proporcionados
page_name = f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/flows/{DIALOGFLOW_CX_FLOW_ID}/pages/{DIALOGFLOW_CX_PAGE_ID}"

# Obtener la página actual
page = pages_client.get_page(name=page_name)

# Crear una nueva página con el mismo nombre y displayName pero sin rutas
new_page = Page(name=page.name, display_name=page.display_name)

# Actualizar la página en Dialogflow CX para borrar todas las rutas existentes
updated_page = pages_client.update_page(page=new_page)

# Autenticación de google sheets y obtención de datos del documento
from google.oauth2 import service_account
from google.cloud.dialogflowcx.v3.services.pages import PagesClient
from google.cloud.dialogflowcx.v3.types.page import Page, Form
from google.cloud.dialogflowcx.v3.types.page import TransitionRoute
from google.cloud.dialogflowcx.v3.types import Fulfillment
from googleapiclient.discovery import build

# Configuración de Dialogflow CX
GOOGLE_SHEETS_CREDENTIALS = "/content/proyecto-final-379916-505ec0fealle.json"
DIALOGFLOW_CX_PROJECT_ID = 'proyecto-final-379916'
DIALOGFLOW_CX_LOCATION = 'us-central1'
DIALOGFLOW_CX_AGENT_ID = '2542e633-3551-4551-9862-c1c95acc71b7'
DIALOGFLOW_CX_FLOW_ID = '00000000-0000-0000-0000-000000000000' # ID del flujo que contiene la página
DIALOGFLOW_CX_PAGE_ID = 'f603290a-5278-4775-b40e-9b710db103d6' # ID de la página a la que se quiere acceder
PAGE_TRANSICION = "354634ba-24f9-4879-9227-93e72a9e4269" # ID de la página "Tienes otra duda"

# Configuración de Google Sheets
SPREADSHEET_ID = '1odR1VizvZurolfMY-1j2V2ectXGr8ZDoP_qvs9Jfs1Q'
RANGE_NAME = 'Hoja 1!A2:C'

# Autenticación con Dialogflow CX y Google Sheets
credentials = service_account.Credentials.from_service_account_file(GOOGLE_SHEETS_CREDENTIALS)
pages_client = PagesClient(credentials=credentials, client_options={"api_endpoint": "us-central1-dialogflow.googleapis.com"})
sheets_service = build('sheets', 'v4', credentials=credentials)

# Obtener los valores de la hoja de cálculo
result = sheets_service.spreadsheets().values().get(spreadsheetId=SPREADSHEET_ID,
                                                    range=RANGE_NAME).execute()
values = result.get('values', [])

# Obtener la página actual y crear una copia para modificarla
page = pages_client.get_page(name=f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/flows/{DIALOGFLOW_CX_FLOW_ID}/pages/{DIALOGFLOW_CX_PAGE_ID}")
new_page = Page(display_name=page.display_name, entry_fulfillment=page.entry_fulfillment, form=page.form,
                transition_route_groups=page.transition_route_groups)
```

```

#Eliminar todos los intents y luego creamos un intent por cada fila que exista en el google sheets
#tiempo de ejecución aproximada de 9 minutos para 68 intents
from google.cloud.dialogflowcx_v3.services.intents import IntentsClient
from google.cloud.dialogflowcx_v3.types.intent import Intent
from google.cloud import dialogflowcx_v3 as dialogflowcx
import time

endpoint = 'us-central1-dialogflow.googleapis.com:443'
intents_client = IntentsClient(credentials=credentials, client_options={'api_endpoint': endpoint})

# Eliminar todos los intents excepto dos específicos y el intent de fallback
intents_to_keep = ["Default Welcome Intent", "sin condición para volver a actualizar"]
intents = intents_client.list_intents(parent=f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}")
for intent in intents:
    if not intent.is_fallback and intent.display_name not in intents_to_keep:
        intents_client.delete_intent(name=intent.name)
        time.sleep(1.5) # Agregar una demora entre las llamadas a la API para evitar exceder el límite de llamadas que se pueden hacer a la API

# Crear intents a partir de las filas del documento de Google Sheets
for row in values:
    display_name = row[0]
    training_phrases = row[2].split("-") if len(row) > 2 and row[2].strip() else [] # Verificar contenido de la celda y separar las frases por guiones "-"

    # Crear un nuevo intent con el nombre y las frases de entrenamiento especificadas
    intent = dialogflowcx.Intent(display_name=display_name)
    if training_phrases:
        intent.training_phrases.extend([dialogflowcx.Intent.TrainingPhrase(parts=[dialogflowcx.Intent.TrainingPhrase.Part(text=phrase)], repeat_count=1) for phrase in training_phrases])
    intents_client.create_intent(parent=f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}", intent=intent)

# Agregar una demora entre las llamadas a la API para evitar exceder el límite de llamadas que se pueden hacer a la API
time.sleep(5)

#Por ultimo creamos una route en la pagina "actualizar" para cada intent que se creo anteriormente
from google.cloud.dialogflowcx_v3.services.pages import PagesClient
from google.cloud.dialogflowcx_v3.types import page

WEBHOOK_ID = "44c8f82d-0b0a-489f-8b28-87e92f4e72a4" #ID del webhook para mandar informacion y guardar las preguntas que hacen los usuarios

pages_client = PagesClient(credentials=credentials, client_options={'api_endpoint': endpoint})

page_name = f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/flows/{DIALOGFLOW_CX_FLOW_ID}/pages/{DIALOGFLOW_CX_PAGE_ID}"

# Obtener la página
my_page = pages_client.get_page(name=page_name)

# Obtener una lista de todos los intents en el agente
intents = intents_client.list_intents(parent=f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}")

# Crear un diccionario para mapear los nombres de los intents a sus IDs
intent_ids = {intent.display_name: intent.name.split('/')[1] for intent in intents}

# Crear una ruta para cada fila del documento de Google Sheets
for row in values:
    display_name = row[0]
    fulfillment_message = row[1]

    # Obtener el ID y el nombre del recurso del intent existente
    intent_id = intent_ids[display_name]
    intent_name = f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/intents/{intent_id}"

    # Crear una nueva ruta con el nombre y el mensaje de cumplimiento especificados
    route = page.TransitionRoute(intent=intent_name)
    route.trigger_fulfillment.messages.append(dialogflowcx.ResponseMessage(text=dialogflowcx.ResponseMessage.Text(text=[fulfillment_message])))

    # Seleccionar el webhook que se utilizará para esta ruta
    route.trigger_fulfillment.webhook = f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/webhooks/{WEBHOOK_ID}"

    # Especificar la etiqueta de cumplimiento para el webhook
    route.trigger_fulfillment.tag = "Enviar pregunta del usuario"

    # Agregar la transición de página a otra página específica
    route.target_page = f"projects/{DIALOGFLOW_CX_PROJECT_ID}/locations/{DIALOGFLOW_CX_LOCATION}/agents/{DIALOGFLOW_CX_AGENT_ID}/flows/{DIALOGFLOW_CX_FLOW_ID}/pages/{PAGE_TRANSICION}"

    # Agregar la ruta a la página
    my_page.transition_routes.append(route)

# Actualizar la página con las nuevas rutas
request = page.UpdatePageRequest(page=my_page)
pages_client.update_page(request)

```