



**Proyecto integrador (Gpo 10)**

**Dra. Grettel Barceló Alonso**

**Dr. Luis Eduardo Falcón Morales**

**Mtra. Verónica Sandra Guzmán de Valle**

**Proyecto:** Como la GenIA está redefiniendo el diseño: Partir de lo abstracto para llegar a lo concreto

**Alumnos:**

Ian André Ortega Suárez-A01021190

Enrique Ricardo García Hernández-A01315428

Proyecto en colaboración con el Dr. Ricardo A. Ramirez-Mendoza, Profesor del área de Mecatrónica e Ingeniería Mecánica, con base a su reporte técnico “Engineering the Present: How Generative AI Models Are Redefining Design”

Avance 4 Modelos alternativos

**Domingo 26 de Mayo de 2024**

<b>Introducción.</b>	<b>1</b>
<b>Objetivo del proyecto.</b>	<b>2</b>
1. Implementación y entrenamiento de modelos generativos en Vertex AI	2
2. Creación y configuración del agente conversacional en Dialogflow:	2
3. Integración y despliegue del chatbot en Google Chat	2
4. Evaluación y mejora continua utilizando Google Cloud Pub/Sub y Cloud Storage	2
<b>Propuesta de la arquitectura</b>	<b>3</b>
1. Componentes Principales	3
2. Descripción de los Componentes	3
<b>Metodología</b>	<b>4</b>
<b>Procesamiento en Python para entrenamiento</b>	<b>6</b>
<b>Chatbot en producción</b>	<b>8</b>
<b>Resultados Esperados</b>	<b>8</b>
<b>Conclusión</b>	<b>8</b>
<b>Referencias bibliográficas</b>	<b>9</b>

## Introducción.

Esta entrega se utilizó para el avance 3, pero dado que el avance 4 requería pruebas de diferentes modelos, cosa que fue realizada para la entrega anterior junto con la tokenización de datos para mejorar los resultados, concluimos en utilizar igualmente esta entrega para el avance 4 que ya se ha visto junto con el Doctor Ricardo. La entrega final contará con la versión final del modelo lograda mediante diversas pruebas y mejoras.

En el ámbito de la inteligencia artificial (IA), los chatbots han emergido como una de las aplicaciones más prometedoras y versátiles. Estas interfaces conversacionales están diseñadas para interactuar con los usuarios de manera natural, emulando la conversación humana. En este contexto, los avances en modelos de lenguaje generativos han revolucionado la forma en que los chatbots pueden comprender y generar texto, permitiendo una interacción más fluida y eficiente.

El presente entregable se enfoca en el desarrollo de un chatbot para atender requerimientos / dudas referentes a la política de gestión de créditos de una empresa financiera, utilizando herramientas de inteligencia artificial generativa. El objetivo principal es crear un sistema que pueda mantener conversaciones coherentes y contextualmente relevantes con los usuarios, en este caso agentes promotores de crédito, proporcionando respuestas precisas y útiles. Para ello, se emplearán técnicas avanzadas de procesamiento de lenguaje natural (NLP) y modelos de aprendizaje profundo.

# Objetivo del proyecto.

## 1. Implementación y entrenamiento de modelos generativos en Vertex AI

- Utilizar Vertex AI para entrenar y desplegar modelos de lenguaje generativos avanzados como GPT-4.
- Ajustar y personalizar los modelos utilizando datos específicos del dominio para mejorar la relevancia y precisión de las respuestas.

## 2. Creación y configuración del agente conversacional en Dialogflow:

- Desarrollar el agente conversacional en Dialogflow para gestionar la lógica de las conversaciones y la interacción con los usuarios.
- Configurar las intenciones y entidades necesarias para comprender y responder adecuadamente a las consultas de los usuarios.

## 3. Integración y despliegue del chatbot en Google Chat

- Implementar Google Chat API para conectar el chatbot con Google Chat, permitiendo la interacción en tiempo real con los usuarios.
- Desarrollar Cloud Functions para manejar la lógica de negocio y las tareas específicas necesarias para las interacciones del chatbot.

## 4. Evaluación y mejora continua utilizando Google Cloud Pub/Sub y Cloud Storage

- Establecer un sistema de comunicación asíncrona mediante Google Cloud Pub/Sub para integrar de manera eficiente todos los componentes del sistema.
- Utilizar Google Cloud Storage para almacenar datos relevantes y modelos entrenados, asegurando la persistencia y accesibilidad de la información.
- Implementar métricas de rendimiento y análisis de datos para evaluar la efectividad del chatbot y aplicar mejoras iterativas basadas en el feedback de los usuarios.

# Propuesta de la arquitectura

Para construir un chatbot utilizando las herramientas de Google Cloud Platform (GCP), Vertex AI, Dialogflow y Google Chat API, la arquitectura debe diseñarse para maximizar la eficiencia, escalabilidad y facilidad de mantenimiento. A continuación, se detallan sus componentes.

## 1. Componentes Principales

**Google Cloud Platform (GCP):** La plataforma principal para alojar y gestionar todos los recursos.

**Vertex AI:** Utilizado para el entrenamiento, despliegue y administración de los modelos de inteligencia artificial.

**Dialogflow:** Servicio de creación de agentes conversacionales que permite gestionar la lógica de conversación.

**Google Chat API:** Para integrar el chatbot con Google Chat y permitir la interacción con los usuarios.

**Google Cloud Functions:** Para manejar las funciones serverless que ejecutan lógica específica según las interacciones del usuario.

**Google Cloud Storage:** Para almacenar datos y modelos necesarios para el funcionamiento del chatbot.

**Google Cloud Pub/Sub:** Para gestionar la comunicación asíncrona entre los distintos componentes del sistema.

## 2. Descripción de los Componentes

**Google Chat:** Punto de interacción donde los usuarios se comunican con el chatbot.

**Google Chat API:** Recibe los mensajes de los usuarios desde Google Chat y los envía a Dialogflow.

**Dialogflow:** Procesa las entradas de los usuarios, gestiona la lógica de las conversaciones y envía las consultas adecuadas a los modelos de Vertex AI.

**Vertex AI NLP:** Utiliza modelos de lenguaje generativos para interpretar las consultas de los usuarios y generar respuestas adecuadas.

**Cloud Functions:** Ejecuta funciones serverless para realizar tareas específicas como integraciones adicionales, almacenamiento de datos o lógica de negocio personalizada.

**Cloud Pub/Sub:** Facilita la comunicación asíncrona y la integración entre los distintos servicios y componentes del sistema.

**Cloud Storage:** Almacena datos y modelos entrenados necesarios para el funcionamiento del chatbot, asegurando la persistencia y accesibilidad de la información.

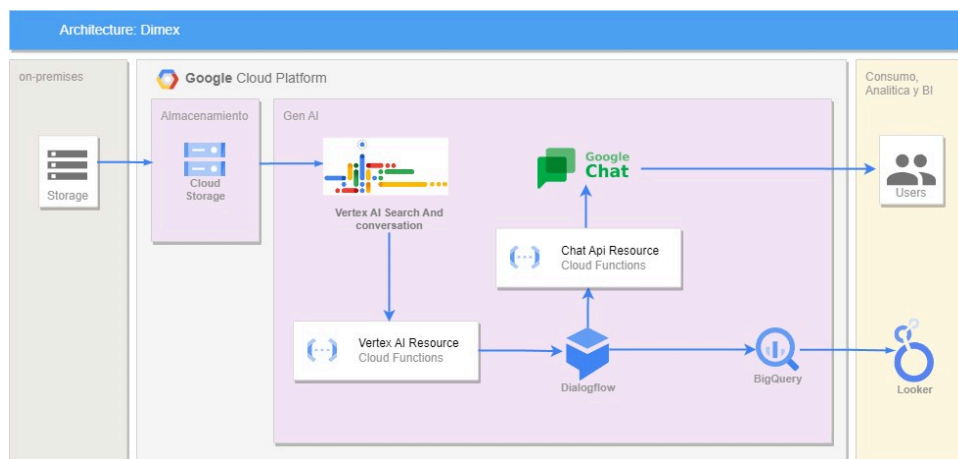


Figura 1. Arquitectura propuesta

## Metodología

La metodología para este proyecto se divide en las siguientes etapas:

### 1. Recolección de Datos:

- Compilación de un corpus de datos de entrenamiento relevante, incluyendo conversaciones reales y preguntas frecuentes de usuarios en el dominio de interés.
- Garantizar que los datos sean de alta calidad y representativos de los casos de uso previstos para el chatbot.

### 2. Preprocesamiento de Datos:

- Limpieza y preparación de los datos para asegurar su calidad y relevancia, aplicando técnicas de tokenización, eliminación de stopwords y lematización.

- Dividir los datos en conjuntos de entrenamiento, validación y prueba para evaluar el rendimiento del modelo.
3. Entrenamiento del Modelo en Vertex AI:
    - Utilización de Vertex AI para entrenar modelos de lenguaje generativos avanzados.
    - Fine-tuning de los modelos con datos específicos del dominio para mejorar su desempeño en contextos particulares.
  4. Desarrollo del Agente Conversacional en Dialogflow:
    - Configuración del agente en Dialogflow, definiendo intenciones, entidades y flujos de conversación.
    - Integración con Vertex AI para aprovechar las capacidades avanzadas de procesamiento de lenguaje natural.
  5. Despliegue e Integración:
    - Implementación de Google Chat API para conectar el chatbot con Google Chat.
    - Desarrollo de Cloud Functions para manejar la lógica de negocio y las integraciones necesarias.
    - Configuración de Google Cloud Pub/Sub para la comunicación asíncrona entre los componentes del sistema.
  6. Evaluación y Mejora Continua:
    - Establecimiento de métricas de rendimiento para evaluar la precisión, relevancia y coherencia de las respuestas del chatbot.
    - Recopilación y análisis de feedback de los usuarios para identificar áreas de mejora.
    - Iteración sobre el diseño y funcionalidad del chatbot para optimizar su desempeño.

# Procesamiento en Python para entrenamiento

```
## Se hace la conexión con google Drive
from google.colab import drive
drive.mount('/content/drive', force_remount=True)

#%cd "/content/drive/SharedDrives/Ciencia de datos - Riesgos/Proyectos/2023/Chatbot/"
Mounted at /content/drive

[ ] ## IMPORTACIONES A UTILIZAR
!pip install numpy_financial
## Procesamiento del lenguaje natural
import nltk
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('vader_lexicon')
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.sentiment import SentimentIntensityAnalyzer

#from sentiment_analysis_spanish import sentiment_analysis ## Análisis de sentimientos en español
#sentiment = sentiment_analysis.SentimentAnalysisSpanish()

import json
import pickle

import re # Se utiliza para manejar texto
from unicodedata import normalize

## Manipulación de datos
import numpy as np
import numpy_financial as npf
```

Figura 2. Pre-procesamiento de datos. Chatbot

+ Código + Texto

▼ **Procesamiento del lenguaje natural**

Se realiza el tratamiento requerido para llevar todos los inputs del usuario a su lema.

```
[ ] ## Importando el archivo que contiene el corpus de las posibles contestaciones
file_route = "/content/drive/MyDrive/corpus.json"
corpus = json.loads(open(file_route, encoding='utf-8').read())
print(corpus)

{'intents': [{'tag': 'Quiere_Credito_True', 'patterns': ['Si', 'Siu', 'Siuu', 'Sí', 'Síp', 'Claro', 'Clarin', 'De acuerdo', 'Me interesa', 'Simón', 'Afirmativo', 'Pues sí', 'Concuendo', 'Sipi',
```

Figura 3. Procesamiento de datos. Chatbot

## ▼ Preguntas del chatbot

Se crea una función por cada posible pregunta que puede realizar el chatbot.

```
## Función que responde a la pregunta sobre si el usuario está interesado en un crédito
def obtenerInteresaCredito():
    keys = ['Quiere_Credito_True', 'Quiere_Credito_False']
    response = 'Lo siento, intenta brindar una respuesta más sencilla.'

    while (response == 'Lo siento, intenta brindar una respuesta más sencilla.'):
        print("Bot: " + 'Hola, ¿estás buscando un crédito?')
        user_input = input("Tú: ")
        response = generarRespuesta(user_input, stop_words, keys)
        print("Bot: " + response)

    for key in keys:
        if response in clases[key][1]:
```

Figura 4. Preguntas iniciales.

## Flujo del Chatbot

Se implementa el flujo que seguirá el chatbot, así como la recopilación de la información necesaria para dar una respuesta de crédito

```
def conversacionChatbot():  
    ## Flujo de la conversación  
    limite_superior_edad = 76  
    limite_inferior_edad = 25  
    estatus = 0  
    interes_aceptar = 'alternativa'  
  
    while estatus == 0:  
        interes_credito = obtenerInteresaCredito() ## Se pregunta si el usuario quiere un crédito  
        if interes_credito == True:  
            edad_cliente = obtenerEdadCliente() ## Se pregunta la edad del cliente  
            if ((edad_cliente >= limite_inferior_edad) & (edad_cliente <= limite_superior_edad)):  
                estatus_cliente = obtenerEstatusCliente() ## Se obtiene si el cliente es jubilado, pensionado o activo  
                if estatus_cliente == "NINGUNA":  
                    agradecimientos()  
                    estatus = 1  
                else:  
                    bool_institucion = obtenerInstitucionBool(estatus_cliente) ## Se obtiene si el cliente el jubilado esta afiliado a una institución de convenio  
                    if bool_institucion == False:  
                        agradecimientos()  
                        estatus = 1  
                    else:  
                        cliente_institucion = obtenerInstitucionCliente(estatus_cliente) ## Se obtiene la institución a la que está afiliado el cliente  
                        if cliente_institucion == "NINGUNA":  
                            agradecimientos()  
                            estatus = 1  
                        else:  
                            cliente_banco = obtenerBancoCliente(estatus_cliente, cliente_institucion)
```

Figura 5. Flujo de chatbot

```
Bot: Hola, ¿estás buscando un crédito?  
Tú: sí  
Bot: ¡Perfecto! Por favor respuesta a la siguiente serie de preguntas para saber qué es lo que le podemos ofrecer  
Bot: ¿Qué edad tiene usted?  
Tú: 66  
Bot: ¿Me podría decir si es jubilado, pensionado o si aún se mantiene activo en el ámbito laboral?  
Tú: jubilado  
Bot: ¿Es jubilado de alguna de estas instituciones? IMSS, ISSSTE, CFE, PEMEX  
Tú: sí  
Bot: ¿En qué institución es usted jubilado? (IMSS, ISSSTE, PEMEX, CFE)  
Tú: IMSS  
Bot: ¿En qué banco le depositan su jubilación IMSS?  
Tú: banorte  
Bot: ¿Cuánto le depositan en BANORTE de su jubilación IMSS?  
Tú: 10 mil  
Bot: ¿Podría ingresar el monto que se le deposita?  
Bot: ¿Cuánto le depositan en BANORTE de su jubilación IMSS?  
Tú: 10,000  
Bot: ¿Podría ingresar el monto que se le deposita?  
Bot: ¿Cuánto le depositan en BANORTE de su jubilación IMSS?  
Tú: 10,000  
Bot: ¿Podría ingresar el monto que se le deposita?  
Bot: ¿Cuánto le depositan en BANORTE de su jubilación IMSS?  
Tú: 10000  
Bot: Buenas noticias, con la información que nos ha proporcionado le podemos ofrecer hasta $25500.0 pagando $2500.0 mensuales durante 36 meses  
Bot: La oferta realizada está sujeta a revisión de capacidad de pago, presentando la documentación requerida.  
Bot: ¿Le interesa la oferta que le acabo de mostrar?  
Tú: sí  
Bot: ¡Perfecto!  
Bot: ¿Por qué medio le gustaría que lo siguiéramos atendiendo? (Kiosko más cercano, Sucursal más cercana, Descargar la app en su dispositivo móvil, Call Center)  
Tú: sucursal  
Bot: Muy bien  
  
Bot: Las sucursales más cercanas a ti son:  
1. Alameda  
2. San Pedro  
3. Garza Sada
```

Figura 6. Ejemplo de una conversación real



# Chatbot en producción

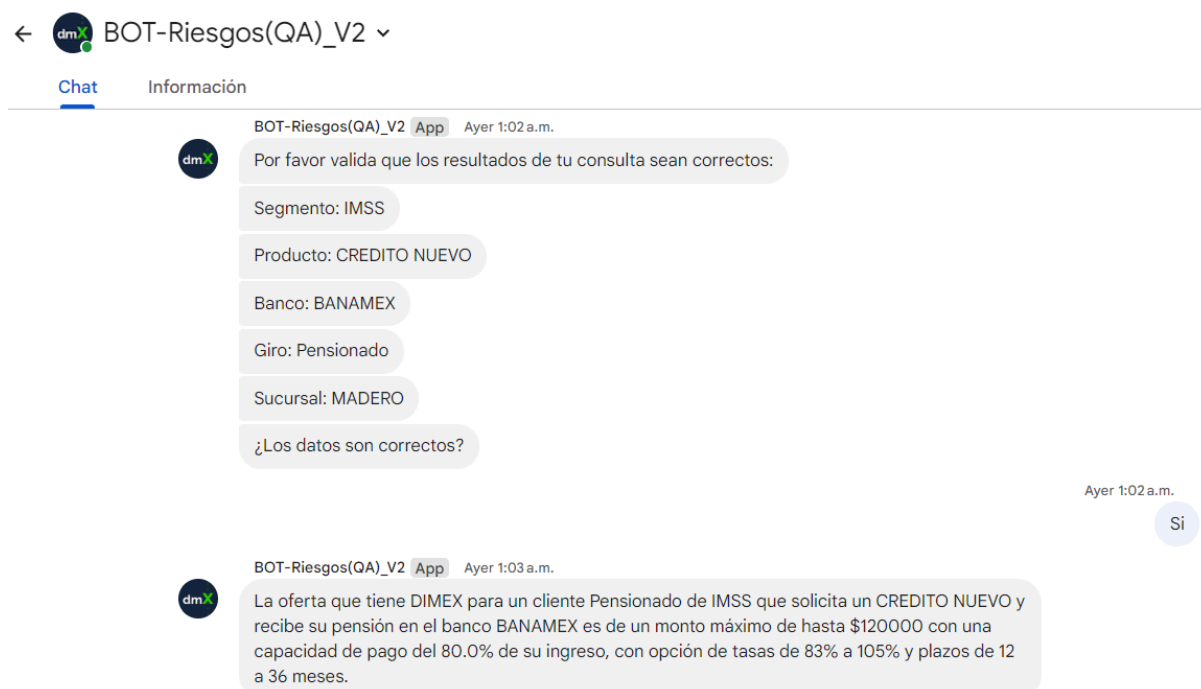


Figura 7. Extracto de chatbot en ambiente de pre-producción

## Resultados Esperados

Se espera que el chatbot en desarrollo sea capaz de manejar una amplia gama de consultas y conversaciones con un alto grado de precisión y naturalidad. Además, se anticipa que el sistema sea adaptable y escalable, permitiendo futuras expansiones y mejoras en función de las necesidades de los usuarios y de la compañía, ampliando la(s) capa(s) de entrenamiento así como los avances tecnológicos en el campo de la IA.

## Conclusión

El desarrollo de un chatbot utilizando herramientas de IA generativa representa un paso significativo hacia la creación de interfaces más intuitivas y efectivas. Este proyecto no solo tiene el potencial de mejorar la interacción usuario-máquina, sino también de contribuir al campo de la inteligencia artificial aplicada, demostrando el poder y la versatilidad de los modelos de lenguaje generativos en aplicaciones prácticas. La integración de Google Cloud Platform, Vertex AI, Dialogflow y Google Chat API asegura una arquitectura robusta y escalable, capaz de ofrecer una experiencia de usuario de alta calidad y adaptable a futuros desafíos y necesidades.

# Referencias bibliográficas

Google Cloud. (2024). Google Cloud Platform documentation. Recuperado de:  
<https://cloud.google.com/docs>

Google Cloud. (2024). Vertex AI documentation. Recuperado de:  
<https://cloud.google.com/vertex-ai/docs>

Google Cloud. (2024). Dialogflow documentation. Recuperado de:  
<https://cloud.google.com/dialogflow/docs>

Google Cloud. (2024). Google Chat API documentation. Recuperado de:  
<https://developers.google.com/chat>

Google Cloud. (2024). Google Cloud Functions documentation. Recuperado de:  
<https://cloud.google.com/functions/docs>

Google Cloud. (2024). Google Cloud Storage documentation. Recuperado de:  
<https://cloud.google.com/storage/docs>

Google Cloud. (2024). Google Cloud Pub/Sub documentation. Recuperado de:  
<https://cloud.google.com/pubsub/docs>