Introducción Práctica a la API de Open AI

1950LABS

Material adicional - Workshop Campus Party 2024

Santiago Vespa
Diego Merentiel
Gonzalo Rodríguez

INDICE

Introducción al Ejercicio

El ejercicio que te presentamos es una herramienta desarrollada en Angular 18. La misma ha sido diseñada para recibir un fragmento de código (snippet) a través de un campo de entrada de texto, permitiendo al usuario seleccionar una acción específica a ejecutar sobre ese fragmento de código.

En principio nosotros proponemos las acciones:

- Corregir bugs.
- Explicar el código.
- Corregir nomenclatura.
- Escribir documentación.
- Generar casos de prueba.

Pero el código es flexible, permitiendo que se puedan agregar nuevas acciones según sea necesario.

Cada una de estas acciones puede ser ejecutada fácilmente mediante una solicitud (request) a la API de OpenAI. Solo necesitas ingresar un prompt simple con los parámetros requeridos. Aunque ofrecemos esta simplicidad para las pruebas rápidas, **te sugerimos explorar los parámetros opcionales disponibles para obtener resultados más precisos y atractivos según cada caso específico.**

Más abajo en este documento, encontrarás ejemplos detallados de cómo utilizar los parámetros opcionales más importantes y qué tipo de respuestas puedes esperar en diferentes escenarios.

Estos ejemplos te ayudarán a entender mejor las capacidades y el alcance.

API Key Open AI

La API Key de Open AI tiene un costo. Para el taller, proporcionamos una API Key. Si deseas usar el prototipo después, deberás registrarte, agregar fondos y actualizar la API Key en el entorno.

Instalación de Angular y Node

Pasos para descargar e instalar Node.Js

El prototipo funciona con Node Js - v18.9.1 o superior.

1. **Descargar Node Js**
   a. Ir a la página nodejs.org.
   b. Descargar el instalador adecuado según tu sistema operativo (Windows, macOS, Linux).
   c. Recomendación bajar la versión LTS disponible, actualmente es la v20.16.0.

2. **Instalar Node Js**
   a. Ejecutar el instalador descargado.
   b. Seguir las instrucciones del instalador, aceptando los términos y condiciones y eligiendo la configuración por defecto.
   c. Una vez completada la instalación abrir una terminal o consola de comandos.

3. **Verificar instalación**
   a. Escribir en la terminal el comando *node -v*
   b. Verificar la instalación de npm, ingresando el comando *npm -v* en ambos casos debería mostrar la versión de cada uno.

En caso de necesitar actualizar la versión de NPM se puede hacer con el comando *npm install -g npm*

Pasos para instalar Angular

Una vez instalado Node Js, solo resta ejecutar el siguiente comando en una terminal.

npm install -g @angular/cli

ng version para comprobar que se instaló correctamente.

Descargar Código

Opción con Git

Si tu equipo cuenta con Git instalado puedes descargar el proyecto del siguiente repositorio:

https://github.com/1950Labs/workshop-chatgpt-api.git

Opción sin Git

Si tu equipo no cuenta con Git instalado podés descargar el proyecto desde una carpeta compartida en Google Drive.

https://shorturl.at/B2HSU

Instalación de Librerías

Una vez descargado el código deben ir por terminal a la raiz del proyecto y ejecutar el comando.

npm install

Una vez finalizado eso que demora unos 5 minutos.

Iniciar sistema

ng serve -o

Detener sistema

Ctrl + C en la terminal para detener.

Ejemplo de Request a la API desde el Proyecto

En este caso vamos a enfocarnos solo en la comunicación con la API de Open AI por lo cual para implementar cada una de estas features, vamos a trabajar en la clase **ChatGptService** que pueden encontrar en **src/app/chat-gpt.service.ts.**

Endpoint y Header

Para todos los request vamos a precisar incluir en el header el API Key y consumir el mismo endpoint por lo tanto en la clase ChatGPTService, hemos incluido las siguientes variables.

```
private readonly _apiUrl = 'https://api.openai.com/v1/chat/completions';

headers = new HttpHeaders({
  'Authorization': `Bearer ${environment.apiKey}`,
  'Content-Type': 'application/json'
});
```

Ejemplo Request desde ChatGptService

```
refactorCode(input: string) {
  return this._http.post(this._apiUrl,
    {
      model: 'gpt-4o-mini',
      messages: [
        {
          role: "system",
          content: `Se te proporcionará un fragmento de código. Refactoriza el
          código para mejorar su legibilidad y eficiencia.`
        },
        {
          role: "user",
          content: `Refactoriza el siguiente fragmento de código para mejorar su
          legibilidad y eficiencia:\n### Fragmento de Código\n\`${input}`
        }
      ],
      temperature: 0.4, // priorizar respuestas precisas y coherentes
      top_p: 0.8, // se utiliza junto a una temperatura baja para generar respuestas precisas y coherentes
      frequency_penalty: 0.2, // no penaliza la repetición de palabras o frases
      presence_penalty: 0.2 // no penaliza la introducción de nuevos conceptos
    },
    { headers: this.headers }
  );
}
```

Pasos para obtener un API Key

Paso 1: Crear una Cuenta en OpenAI

1. **Visita el Sitio Web de OpenAI**: Abre tu navegador y ve a OpenAI.
2. **Regístrate o Inicia Sesión**: Si ya tienes una cuenta, haz clic en "Log In" e ingresa tus credenciales. Si no tienes una cuenta, haz clic en "Sign Up" para crear una nueva.

Paso 2: Acceder al Panel de Control

1. **Panel de Control**: Una vez que hayas iniciado sesión, serás redirigido al panel de control de OpenAI.
2. **Navegar a la Sección de API Keys**: Busca una opción en el menú que diga "API Keys" o "API" (generalmente se encuentra en la configuración o en el menú de tu perfil).

Paso 3: Generar una Nueva API Key

1. **Crear una Nueva API Key**:
   - **Botón de Creación**: En la sección de API Keys, deberías ver un botón que dice "Create New Key" o algo similar. Haz clic en ese botón.
   - **Nombre de la API Key**: Puede que te pidan nombrar tu API Key para identificar la fácilmente (esto es opcional y depende de la interfaz actual de OpenAI).
2. **Guardar la API Key**:
   - **Visualización de la API Key**: Una vez generada, se te mostrará la API Key en la pantalla.
   - **Guardar la API Key**: Copia y guarda la API Key en un lugar seguro. Esta es la clave que utilizarás para autenticar tus solicitudes a la API de OpenAI. **Nota importante**: No compartas esta clave públicamente ni la incluyas en repositorios de código públicos.

Paso 4: Configuración de la API Key en Tu Aplicación

1. **Integración en Código**:

```
curl https://api.openai.com/v1/models \
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -H "OpenAI-Organization: org-O6AO1CljTZDtnGTJ144uHWjE" \
```

Parámetros Requeridos - model

Model

En model les sugerimos elegir entre alguna de las siguientes opciones:
*Todos los precios están expresados costo por millón de tokens.

- gpt-4o (Input 5 USD / Output: 15 USD)*
- gpt-4o-mini (Input 0.15 USD / Output: 0.6 USD)*
- gpt-4 (Input 10 USD / Output: 60 USD)*
- gpt-4-turbo (Input 10 USD / Output: 30 USD)*

Todos estos modelos son capaces de comprender y generar texto coherente y relevante en varios idiomas.

GPT-4 y GPT-4 Turbo
La principal diferencia entre ellos es el precio, la versión Turbo es más barata, también es más rápida y está optimizado para aplicaciones que requieren respuestas rápidas y eficientes.

GPT-4o
Esta versión es optimizada y más avanzada que GPT-4. Fue lanzado con la intención de superar las limitaciones del modelo anterior, ofreciendo mejores capacidades y rendimiento superior en diversas áreas.

GPT-4o incluye capacidades multimodales, que le permiten procesar diferentes tipos de datos (texto, audio e imágen) al mismo tiempo.

El costo de gpt-4o es significativamente inferior al de gpt-4 o gpt-4-turbo, lo que lo hace una buena opción.

GPT-4o-mini
Es una versión económica del modelo GPT-4o, que es más ligera y más rápida pero menos potente en cuanto a la comprensión y resultados generados.

Admite entradas y salidas multimodales, incluyendo texto y visión, y en un futuro soportará formatos de audio y video.

Parámetros Requeridos - messages

Messages es un array de objetos con dos propiedades requeridas, Role y Content.

Los roles disponibles son:

- **System**: Proporciona instrucciones iniciales al modelo sobre cómo debe comportarse a lo largo de la conversación.
- **User**: Representa el prompt ingresado por el usuario. Estos son mensajes de entradas directas que el usuario proporciona al modelo.
- **Assistant:** Representa respuestas del modelo a request realizados previamente. Se pueden ir incluyendo en nuevos request para brindarle contexto mejorado al modelo.

Content es el prompt que se ingresa para cada rol.

```
messages: [
  {
    role: "system",
    content: `Se te proporcionará un fragmento de código. Refactoriza el
    código para mejorar su legibilidad y eficiencia.`
  },
  {
    role: "user",
    content: `Refactoriza el siguiente fragmento de código para mejorar su
    legibilidad y eficiencia:\n### Fragmento de Código\n\`${input}`
  }
```

Parámetros Opcionales - temperature

Controla la aleatoriedad de las respuestas generadas. Valores más bajos cercanos a 0 hacen que el modelo genere texto más conservador y repetitivo.

Tipo de Dato: Float
Rango: [0, 2]

A continuación vemos dos ejemplos extremos.
Temperature = 0

```
{
  "model": "gpt-4o-mini",
  "messages": [
    {
      "role": "user",
      "content": "Explain the democracy in one sentence"
    }
  ],
  "temperature": 0
```

Response 1:
Democracy is a system of government in which power is vested in the people, who exercise that power directly or through elected representatives.

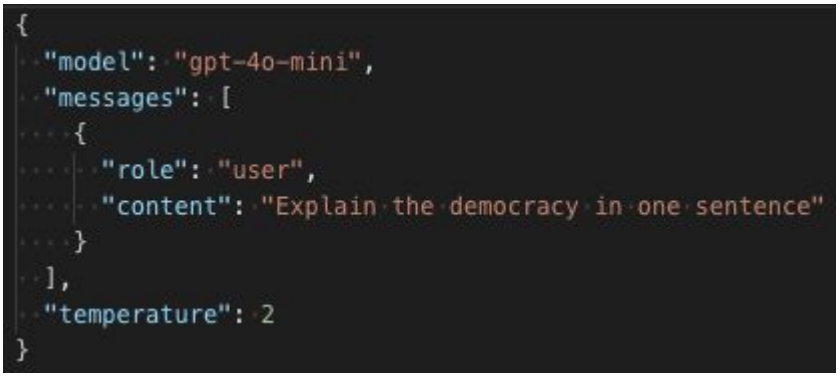
Response 2:
Democracy is a system of government in which power is vested in the people, who exercise that power directly or through elected representatives.

Response 3:
Democracy is a system of government in which power is vested in the people, who exercise that power directly or through elected representatives.

Como podemos ver en todos los casos la respuesta se mantiene, lo cual es muy útil si necesitamos tener control sobre las respuestas generadas para determinados prompts.

Parámetros Opcionales - temperature

Temperature = 2



```
{
  "model": "gpt-4o-mini",
  "messages": [
    {
      "role": "user",
      "content": "Explain the democracy in one sentence"
    }
  ],
  "temperature": 2
}
```

Response 1:
Democracy is a system of government in which power is vested in the populace, typically through elected representatives, enabling citizens to participate collectively in decision-making and exercise their rights.

Response 2:
Democracy is a system of government in which power is vested in the people, who elect their representatives and influence decisions through voting and public participation.

Response 3:
Democracy is a system of government in which power arises from the collaboration of the governed, characterized by regular free and fair elections, the protection of individual rights, and the rule of law.

Como podemos ver en todos los casos la respuesta es diferente y más creativa que en el ejemplo uno.

Al hacer esto le estamos dando más libertad al modelo, pero también tenemos el riesgo de que tiene más posibilidades de darnos una respuesta errónea.

Parámetros Opcionales - top_p

Es un parámetro utilizado en modelos de lenguaje para controlar la diversidad y creatividad de las respuestas generadas.

Se utiliza en un contexto de “muestreo de núcleo” (nucleus sampling) y funciona de la siguiente manera:

`top_p` establece el umbral de probabilidad acumulada. Luego el algoritmo en cada paso de la generación ordena los posibles tokens por su probabilidad. Luego selecciona el conjunto más pequeño de tokens cuya probabilidad acumulada es al menos `top_p`.

Luego del conjunto reducido el modelo elige un token al azar.

En otras palabras, mientras más bajo es el **`top_p`**, menos palabras tiene disponibles para generar una respuesta por lo tanto el resultado será más consistente y repetitivo.

Por otro lado mientras más alto y cercano a 1 es, dispondrá de mayor cantidad de palabras por lo que el resultado será diverso y creativo.

Ejemplo Práctico

- Si **top_p** es 0.9, el modelo considerará los tokens más probables cuya suma de probabilidades alcanza el 90%.
- Si **top_p** es 1.0, el modelo considerará todos los posibles tokens (equivalente a no usar muestreo de núcleo y depender sólo del parámetro temperature).
- Si **top_p** es muy bajo, como 0.1, el modelo solo considerará los tokens más probables hasta que la suma de sus probabilidades alcance el 10%.

Parámetros Opcionales - Relación entre temperature y top_p

temperature	top_p	Resultado
0.2	0.2	Resultados muy conservadores y coherentes, con poca creatividad.
0.2	0.5	Resultados conservadores y coherentes, con creatividad limitada.
0.5	0.5	Equilibrio entre creatividad y coherencia, con resultados interesantes y consistentes.
0.8	0.5	Buen equilibrio entre creatividad y coherencia, con resultados interesantes y consistentes.
2.0	0.9	Máxima creatividad y diversidad, con menor foco en la coherencia y potencialmente respuestas más inesperadas.

Parámetros Opcionales - n

Permite establecer la cantidad de respuestas al mismo prompt.

Request

```
"model": "gpt-4o-mini",
"messages": [
  {
    "role": "user",
    "content": "What are the main benefits of using solar energy?"
  }
],
"max_tokens": 50,
"temperature": 2,
"top_p": 1,
"n": 3
```

Response

Con tres objetos choice.


```
{
    "id": "chatcmpl-9t1Uoqfe4r343dXMUJrkkrBQ7hB3q",
    "object": "chat.completion",
    "created": 1722900818,
    "model": "gpt-4o-mini-2024-07-18",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "Solar energy offers a variety of benefits, which can be
categorized into economic, environmental, energy efficiency, health, and technological
categories. Here are the main benefits:\n\n### Economic Benefits:\n1. **Abundant and
Free Resource:** Shortly embark means tight-bound"
            },
            "logprobs": null,
            "finish_reason": "length"
        },
        {
            "index": 1,
            "message": {
                "role": "assistant",
                "content": "Solar energy offers numerous benefits, making it a popular
alternative to traditional fossil fuels. Here are some of the main advantages:\n\n1.
**Renewable Source**: Solar energy is abundant and renewable as it relies on sunlight,
which is constantly available and not"
            },
            "logprobs": null,
            "finish_reason": "length"
        }
    ],
    "usage": {
        "prompt_tokens": 17,
        "completion_tokens": 100,
        "total_tokens": 117
    },
```

Parámetros Opcionales - frequence_penalty

`frequency_penalty` ayuda a reducir la repetición de palabras o frases en el texto generado. Aumenta la penalización para palabras que ya han aparecido, disminuyendo su probabilidad de ser elegidas de nuevo.

Es útil para mejorar la calidad del texto, bajando la repetición y aumentando la coherencia del resultado.

Aplicación de penalización: Cada vez que una palabra se usa en el texto generado, su probabilidad de ser seleccionada nuevamente se reduce según el valor de **`frequency_penalty`**.

Valor del parámetro: El valor de **`frequency_penalty`** puede variar típicamente entre -2 y 2 aunque puede ser mayor en algunos casos. Un valor más alto implica una mayor penalización para palabras repetidas, lo que fomenta la diversidad en el texto.

Ejemplo

frequency_penalty: -2

- **Prompt:** "Describe the use of AI in modern healthcare."
- **Posible Respuesta:** "AI in modern healthcare is transforming healthcare by using AI to diagnose diseases. AI is also being used to manage patient data, and AI is providing real-time support in surgeries."
- **Explicación:** Con un valor de -2, el modelo tiende a repetir ciertas palabras o frases, como "AI", para enfatizar su importancia o simplemente porque está incentivado a hacerlo.

frequency_penalty: 0

- **Prompt:** "Describe the use of AI in modern healthcare."
- **Posible Respuesta:** "Artificial intelligence (AI) is increasingly being used in healthcare for diagnostics, treatment recommendations, and patient management. It allows for more accurate diagnoses, personalized treatment plans, and efficient data management."
- **Explicación:** Con un valor de 0, el modelo genera una respuesta equilibrada sin ninguna inclinación a repetir o evitar la repetición de palabras.

Parámetros Opcionales - frequence_penalty

frequency_penalty: 2

- **Prompt:** "Describe the use of AI in modern healthcare."
- **Posible Respuesta:** "The integration of artificial intelligence in healthcare is revolutionizing the industry by enhancing diagnostic accuracy, improving patient care, and optimizing data management. AI-driven technologies are providing innovative solutions to complex medical challenges."
- **Explicación:** Con un valor de 2, el modelo evita la repetición de palabras como "AI" o "healthcare", buscando mayor diversidad en el vocabulario y expresando las ideas de manera distinta.

Parámetros Opcionales - presence_penalty

Este parámetro penaliza la aparición de **nuevas instancias de palabras o temas ya mencionados** en la conversación o en la generación actual. Un valor alto incentiva al modelo a introducir nuevas ideas o palabras que aún no han aparecido, mientras que un valor bajo o negativo permite que se mantenga en las mismas ideas o palabras.

Ejemplo de uso:

- **Valor positivo (e.g., 2):** El modelo buscará constantemente introducir nuevas palabras o ideas, evitando quedarse estancado en las mismas frases o conceptos.
- **Valor negativo (e.g., -2):** El modelo puede sentirse "cómodo" reiterando las mismas ideas o palabras.

Cuándo usarlo:

- Cuando deseas que el modelo **explore nuevas ideas o temas sin quedarse en lo mismo**. Es útil en contextos donde se busca innovación o creatividad en la generación de texto.

Ejemplos

presence_penalty: -2

- **Prompt:** "Tell me about the benefits of exercise."
- **Posible Respuesta:** "Exercise is great for improving cardiovascular health. Exercise helps to strengthen muscles. Additionally, exercise can boost mental health, and exercise is known to help with weight management."
- **Explicación:** Con un valor de -2, el modelo es más propenso a repetir palabras o ideas que ya ha mencionado, como "exercise". La repetición es más común, lo que podría hacer que la respuesta parezca redundante.

Parámetros Opcionales - presence_penalty

presence_penalty: 0

- **Prompt:** "Tell me about the benefits of exercise."
- **Posible Respuesta:** "Exercise has numerous benefits, including improving cardiovascular health, strengthening muscles, boosting mental health, and aiding in weight management."
- **Explicación:** Con un valor de 0, el modelo no tiene ninguna penalización o incentivo para repetir o evitar la repetición de ideas o palabras. La respuesta es equilibrada, sin esfuerzo específico para introducir nuevas palabras ni para evitar repetir las ya mencionadas.

presence_penalty: 2

- **Prompt:** "Tell me about the benefits of exercise."
- **Posible Respuesta:** "Engaging in physical activity offers a variety of health advantages, such as enhanced heart function, increased muscle tone, better mental well-being, and effective weight control. Regular movement can also improve your overall energy levels."
- **Explicación:** Con un valor de 2, el modelo busca activamente introducir nuevas palabras o ideas. Evita repetir términos ya mencionados (como "exercise"), y trata de expresar los beneficios de formas más variadas y creativas.

Parámetros Opcionales - Casos de uso frequency_penalty

Casos de uso para `frequency_penalty`

1. **Evitar Repetición de Palabras Comunes**:
   - **Uso**: Redacción de artículos, ensayos, o narrativas largas.
   - **Por qué**: Ayuda a evitar que las palabras comunes se repitan demasiado, mejorando la fluidez y la legibilidad del texto.
2. **Generación de Resúmenes**:
   - **Uso**: Resúmenes automáticos de documentos largos.
   - **Por qué**: Garantiza que las ideas principales no se repitan excesivamente, proporcionando un resumen más claro y conciso.
3. **Chatbots y Asistentes Virtuales**:
   - **Uso**: Conversaciones prolongadas con usuarios.
   - **Por qué**: Mejora la calidad de las respuestas evitando que el bot repita las mismas frases o palabras, haciendo las interacciones más naturales.
4. **Generación de Contenido Creativo**:
   - **Uso**: Escritura de poemas o cuentos.
   - **Por qué**: Mantiene la creatividad del contenido al evitar la repetición de palabras y frases, fomentando la diversidad en la elección de palabras.

Parámetros Opcionales - Casos de uso presence_penalty

Casos de uso para `presence_penalty`

1. **Exploración de Nuevas Ideas**:
   - **Uso**: Generación de lluvia de ideas o brainstorming.
   - **Por qué**: Fomenta la inclusión de nuevas palabras y conceptos, ayudando a explorar una gama más amplia de ideas.
2. **Descripción de Productos**:
   - **Uso**: Redacción de descripciones de productos en e-commerce.
   - **Por qué**: Garantiza que las descripciones sean variadas y no se repitan, lo que puede mejorar el interés del cliente.
3. **Generación de Diálogos**:
   - **Uso**: Escritura de guiones para películas, videojuegos, o teatro.
   - **Por qué**: Asegura que los personajes utilicen un vocabulario diverso, haciendo los diálogos más dinámicos y realistas.
4. **Aprendizaje de Vocabulario**:
   - **Uso**: Herramientas educativas para aprender nuevos idiomas.
   - **Por qué**: Introduce constantemente nuevas palabras, lo que ayuda a los estudiantes a expandir su vocabulario de manera más efectiva.

Response

El objeto `response` generalmente tiene la siguiente estructura:

id

- **Descripción**: Identificador único para la respuesta generada.
- **Tipo**: Cadena (string)
- **Ejemplo**: "chatcmpl-xxxxxxxxxxxxxxx"

object

- **Descripción**: Tipo de objeto devuelto.
- **Tipo**: Cadena (string)
- **Ejemplo**: "chat.completion"

created

- **Descripción**: Marca de tiempo de Unix que indica cuándo se creó la respuesta.
- **Tipo**: Entero (integer)
- **Ejemplo**: 1616175763

model

- **Descripción**: Nombre del modelo utilizado para generar la respuesta.
- **Tipo**: Cadena (string)
- **Ejemplo**: "gpt-4-turbo"

```
{
  "id": "chatcmpl-xxxxxxxxxxxxxxx",
  "object": "chat.completion",
  "created": 1616175763,
  "model": "gpt-4-turbo",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "This is the generated response."
      },
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 5,
    "completion_tokens": 10,
    "total_tokens": 15
  }
}
```

Response

choices

- **Descripción**: Lista de posibles respuestas generadas por el modelo.
- **Tipo**: Matriz de objetos
- **Componentes de choices**:
  - **index**: Índice de esta respuesta en la lista de respuestas posibles.
    - **Tipo**: Entero (integer)
    - **Ejemplo**: 0
  - **message**: Objeto que contiene el mensaje generado.
    - **role**: Rol del emisor del mensaje (siempre "assistant" en las respuestas generadas).
      - **Tipo**: Cadena (string)
      - **Ejemplo**: "assistant"
    - **content**: Contenido del mensaje generado.
      - **Tipo**: Cadena (string)
      - **Ejemplo**: "This is the generated response."
  - **finish_reason**: Razón por la cual la generación de la respuesta se detuvo.
    - **Tipo**: Cadena (string)
    - **Ejemplo**: "stop" (puede ser "stop", "length", "content_filter", etc.)

usage

- **Descripción**: Información sobre el uso de tokens para esta solicitud.
- **Tipo**: Objeto
- **Componentes de usage**:
  - **prompt_tokens**: Número de tokens en el prompt (entrada) proporcionado.
    - **Tipo**: Entero (integer)
    - **Ejemplo**: 5
  - **completion_tokens**: Número de tokens en la respuesta generada.
    - **Tipo**: Entero (integer)
    - **Ejemplo**: 10
  - **total_tokens**: Número total de tokens utilizados (suma de prompt_tokens y completion_tokens).
    - **Tipo**: Entero (integer)
    - **Ejemplo**: 15

Consideraciones de Seguridad

1. Almacenamiento Seguro

- **Evitar el código fuente:** No almacenes la API Key en el código fuente, especialmente en repositorios públicos. Utiliza archivos de configuración o variables de entorno.
- **Gestión de secretos:** Utiliza herramientas como AWS Secrets Manager, Azure Key Vault o HashiCorp Vault para almacenar y gestionar tus claves de manera segura.

2. Transmisión Segura

- **Usa HTTPS:** Asegura todas las comunicaciones con la API utilizando HTTPS para evitar la intercepción de la clave en tránsito.
- **Verificación de certificados:** Valida los certificados SSL del servidor para prevenir ataques de tipo man-in-the-middle.

3. Rotación de Claves

- **Rotación periódica:** Cambia tus claves regularmente para minimizar el impacto en caso de una filtración.
- **Automatización:** Implementa procesos automatizados para la rotación de claves y la actualización de aplicaciones.

4. Límites y Permisos

- **Principio de privilegios mínimos:** Configura las claves con los permisos mínimos necesarios.
- **Límites de tasa:** Aplica límites para evitar abusos y ataques de denegación de servicio (DoS).

5. Monitoreo y Alertas

- **Monitoreo continuo:** Realiza un seguimiento del uso de la API para detectar patrones inusuales.
- **Alertas:** Configura alertas para actividades sospechosas.

Consideraciones de Seguridad

6. Registros y Auditorías

- **Mantén registros detallados:** Registra todas las solicitudes y respuestas de la API para tener un historial completo de uso.
- **Auditorías periódicas:** Realiza auditorías periódicas de los registros para identificar y responder a posibles incidentes de seguridad.

7. Gestión de Errores

- **Manejo seguro de errores:** No expongas información sensible en los mensajes de error. Configura mensajes genéricos para los usuarios y registra los detalles completos solo en el lado del servidor.

8. Autenticación y Autorización

- **Autenticación fuerte:** Implementa mecanismos de autenticación fuertes para los usuarios que acceden a la API.
- **Autorización basada en roles:** Utiliza un sistema de autorización basado en roles para controlar el acceso a diferentes funcionalidades de la API.

9. Implementación de Seguridad Adicional

- **Uso de proxies de seguridad:** Considera el uso de proxies de seguridad para monitorear y controlar el tráfico de la API.
- **Reforzamiento de endpoints:** Asegura que todos los endpoints de la API estén protegidos contra ataques comunes como inyección SQL, cross-site scripting (XSS) y otros.

Links Útiles

Documentación General de la API de OpenAI: OpenAI API Documentation

- Este es el punto de partida principal para entender cómo funciona la API de OpenAI, incluyendo guías de inicio, ejemplos de código y detalles técnicos.

Guía de Inicio Rápido: Quickstart

- Una guía paso a paso para comenzar a usar la API de OpenAI rápidamente.

Referencias de la API: API Reference

- Detalles específicos sobre los endpoints disponibles, parámetros y respuestas.

Ejemplos de Uso: Examples

- Ejemplos prácticos de cómo usar la API para diversas aplicaciones, desde chatbots hasta generación de contenido.

Guía de Modelos: Model Documentation

- Información detallada sobre los diferentes modelos disponibles, incluyendo sus capacidades y limitaciones.

Preguntas Frecuentes: FAQ

- Respuestas a preguntas comunes sobre la API y su uso.

Tablas de Precios de OpenAI: Pricing

- Aquí encontrarás información detallada sobre los costos asociados con el uso de los diferentes modelos y servicios ofrecidos por OpenAI.

Términos de Uso: Terms of Use

- Información sobre los términos y condiciones de uso de la API de OpenAI.

Política de Uso Aceptable: Usage Policies

- Reglas sobre lo que está permitido y prohibido hacer con la API.

1950LABS

Santiago Vespa | Diego Merentiel | Gonzalo Rodríguez
Consultas: gonzalo.rodriguez@1950labs.com