Un **chatbot** es un programa de software diseñado para simular una conversación con personas, ya sea por texto o voz. Su objetivo principal es interactuar con los usuarios de manera automática y brindar respuestas útiles, como si fuera una persona real.

**Tipos de chatbots**

1. **Basados en reglas:** Siguen un conjunto de instrucciones predefinidas. Solo entienden frases específicas y tienen respuestas limitadas.
2. **Basados en inteligencia artificial (IA):** Como yo, utilizan técnicas de procesamiento del lenguaje natural (NLP) y aprendizaje automático para entender y generar respuestas más naturales, incluso a preguntas nuevas o complejas.

**Usos comunes**

* Atención al cliente en sitios web.
* Asistentes virtuales (como Siri, Alexa o Google Assistant).
* Automatización de tareas (reservas, pedidos, soporte técnico).
* Educación o aprendizaje de idiomas.
* Marketing o ventas.

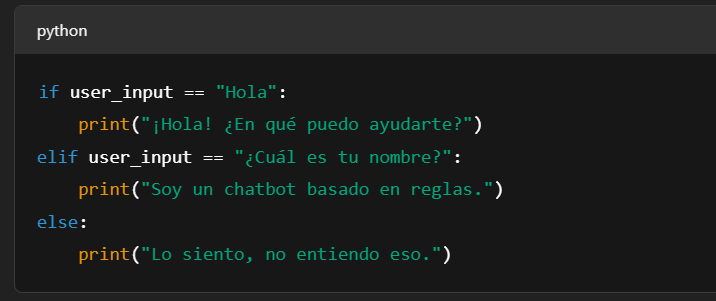
**Basados en reglas:**

Un **chatbot basado en reglas** es el tipo más simple de chatbot. Funciona siguiendo un conjunto de **instrucciones predefinidas** para responder a las preguntas del usuario. No utiliza inteligencia artificial, sino que **responde solo si reconoce exactamente lo que el usuario escribe**.

**🔧 ¿Cómo funciona?**

* Usa **condicionales ("if-else")** o **árboles de decisión**.
* El programador define las **preguntas posibles** y sus **respuestas exactas**.
* Si el usuario escribe algo fuera de lo esperado, el chatbot no sabe qué hacer o responde con algo genérico como: *“No entiendo tu pregunta.”*

🧠 Ejemplo simple en pseudocódigo:



**✅ Ventajas:**

* Fácil de crear y mantener.
* Predecible y controlado (ideal para tareas específicas y repetitivas).
* No necesita gran capacidad de cómputo.

**❌ Desventajas:**

* No entiende lenguaje natural ni variaciones en la forma de escribir.
* No aprende con el tiempo.
* Se vuelve muy limitado si las preguntas se vuelven más complejas o variadas.

¡Perfecto! Vamos a crear un **chatbot basado en reglas** en Python paso a paso. Te daré las instrucciones claras, con explicación de cada parte.

**🤖 Objetivo:**

Hacer un chatbot que pueda responder a frases simples como:

* "Hola"
* "¿Cómo estás?"
* "¿Qué puedes hacer?"
* "Adiós"

**🧰 Requisitos:**

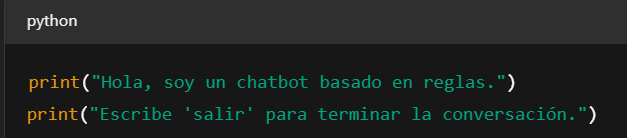
Solo necesitas tener **Python instalado**. No necesitas instalar ninguna librería adicional.

**🧱 Paso 1: Crea tu archivo Python**

Abre tu editor (como VSCode, PyCharm o el Bloc de notas) y guarda un nuevo archivo con el nombre chatbot\_reglas.py.

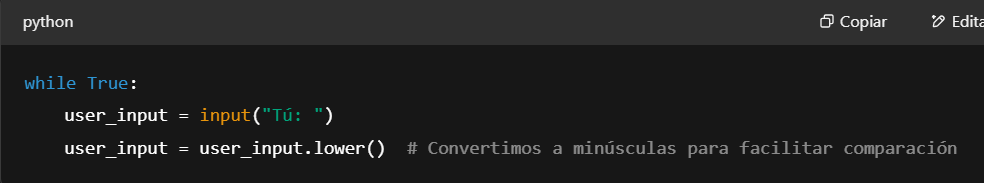
**🪜 Paso 2: Escribe el código paso a paso**

**1. Mostrar un mensaje de bienvenida**



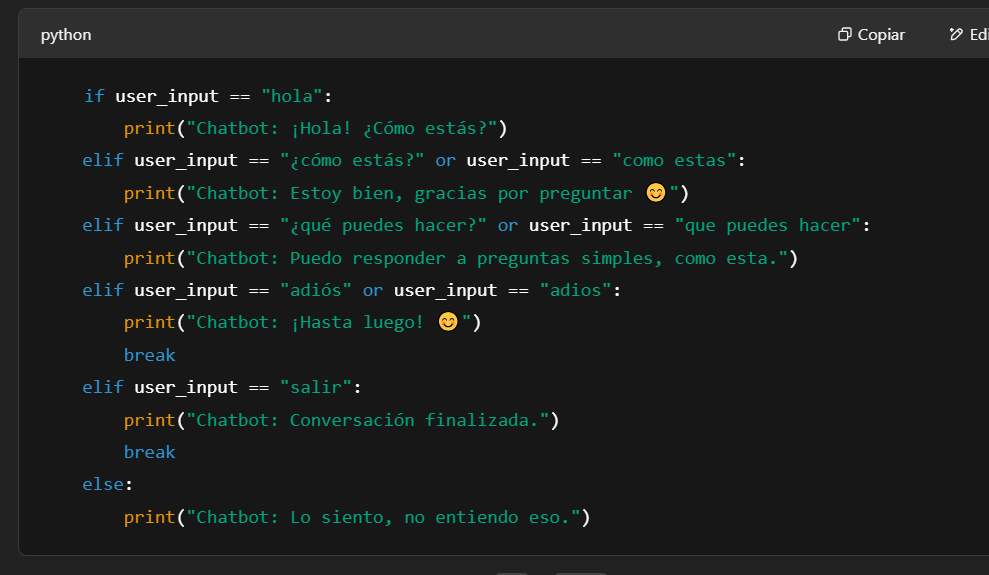
Esto simplemente saluda al usuario y le dice cómo salir.

2. Crear un bucle para que el usuario pueda hablar varias veces



Este bloque permite que el programa escuche al usuario continuamente.

3. Agregar las reglas (respuestas)



Este bloque contiene las **reglas del chatbot**. Cada if o elif representa una frase que el chatbot entiende.

**▶️ Paso 3: Ejecuta el chatbot**

Abre una terminal o consola y ejecuta el archivo con

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Escribe frases como “hola”, “¿cómo estás?” o “adiós” y verás las respuestas del chatbot.

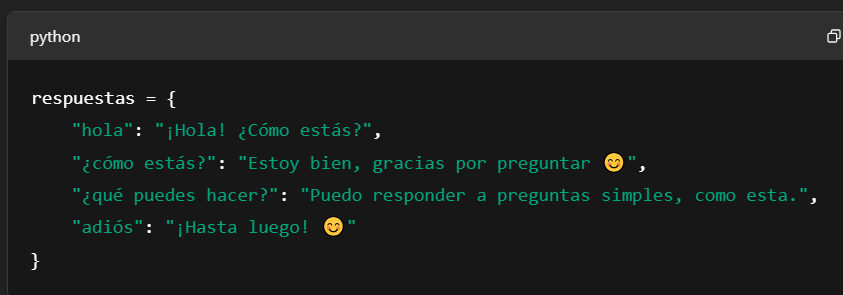
**🧠 Consejo extra:**

Puedes mejorar el chatbot agregando más frases o usando un diccionario en lugar de muchos if… ¿Te gustaría ver esa versión mejorada después?

¡Genial! Ahora vamos a **mejorar el chatbot basado en reglas** usando un **diccionario de respuestas** en lugar de muchos if y elif. Esto hará el código más limpio, más fácil de mantener y más flexible si quieres agregar más frases.

**🧠 ¿Qué es un diccionario en Python?**

Un **diccionario** (dict) en Python es una estructura que guarda pares de datos en formato **clave: valor**.  
Ejemplo:



Aquí, cada **clave** es una frase que el usuario puede escribir, y cada **valor** es la respuesta del chatbot.

**🪜 Paso a paso para usar el diccionario**

**1. Código completo y explicado**

Texto, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.  
Texto

El contenido generado por IA puede ser incorrecto.

**🔍 Explicación:**

* .lower() convierte el texto a minúsculas para que no importe si el usuario escribe "Hola" o "hola".
* .strip() elimina espacios extra al principio o al final.
* if user\_input in respuestas: busca si la frase escrita está en el diccionario.
* Si está, responde con el valor correspondiente.
* Si no está, responde con un mensaje de error.

**✅ Ventajas de usar un diccionario:**

* Puedes agregar nuevas frases fácilmente.
* No necesitas muchos if y elif, lo que hace el código más limpio.
* Puedes incluso guardar las reglas en un archivo si creces más adelante.

¿Te gustaría agregar **respuestas múltiples** para una misma pregunta o incluso incluir **respuestas aleatorias**? También te lo puedo mostrar si quieres.

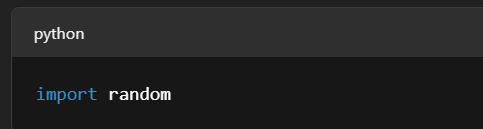
¡Perfecto! Ahora te enseño cómo hacer que el chatbot pueda dar **respuestas diferentes para una misma pregunta**, y elegir una **al azar** cada vez que el usuario escriba.

Esto hace que el chatbot parezca más "natural" y menos repetitivo.

**🎲 Paso 1: Importar random**

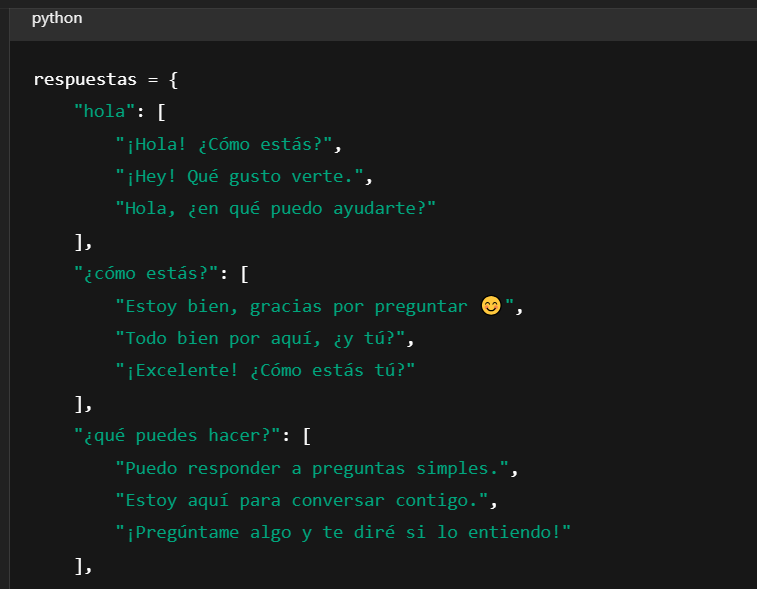
Python tiene una librería llamada random que nos permite **elegir una opción al azar** de una lista.

Primero, hay que importar esa librería:



**🧠 Paso 2: Usar listas como valores en el diccionario**

En lugar de guardar solo una respuesta, ahora cada entrada del diccionario tendrá una **lista de respuestas posibles**.

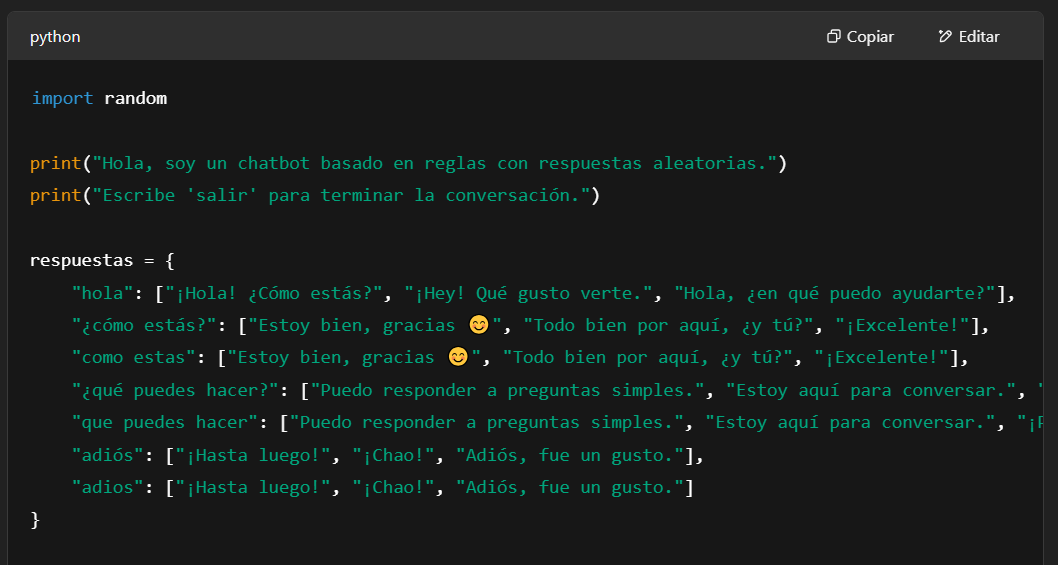
  
Texto

El contenido generado por IA puede ser incorrecto.

También puedes duplicar las claves sin tilde (como "adios") o usar otras variantes si quieres.

**🪜 Paso 3: Modificar el chatbot para usar random.choice()**

Este método elige **una respuesta aleatoria** de la lista:

  
Texto

El contenido generado por IA puede ser incorrecto.

**✅ ¿Qué logras con esto?**

* El chatbot se siente más humano.
* Puedes hacer que las respuestas sean más variadas o incluso divertidas.
* Aprendes una estructura más flexible, ideal para proyectos más grandes.

¡Perfecto! Ahora vamos a dar un paso más avanzado y convertir nuestro chatbot **basado en reglas** en uno que pueda entender **preguntas parecidas**, aunque no sean exactamente iguales.

Para esto, vamos a usar una técnica básica de **procesamiento de lenguaje natural (NLP)** llamada **coincidencia difusa (fuzzy matching)**.

**🎯 Objetivo:**

Queremos que el chatbot entienda cosas como:

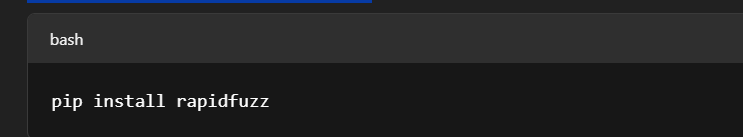
* "Hola" → ✅ reconoce
* "Holaa" → ✅ reconoce (aunque tenga una letra de más)
* "Cómo te encuentras" → ✅ lo interpreta como "¿cómo estás?"

**🧰 Herramienta: fuzzywuzzy (o rapidfuzz, que es más rápida)**

Usaremos la librería **fuzzywuzzy** o su versión más moderna **rapidfuzz** para comparar frases y ver cuál es la más parecida.

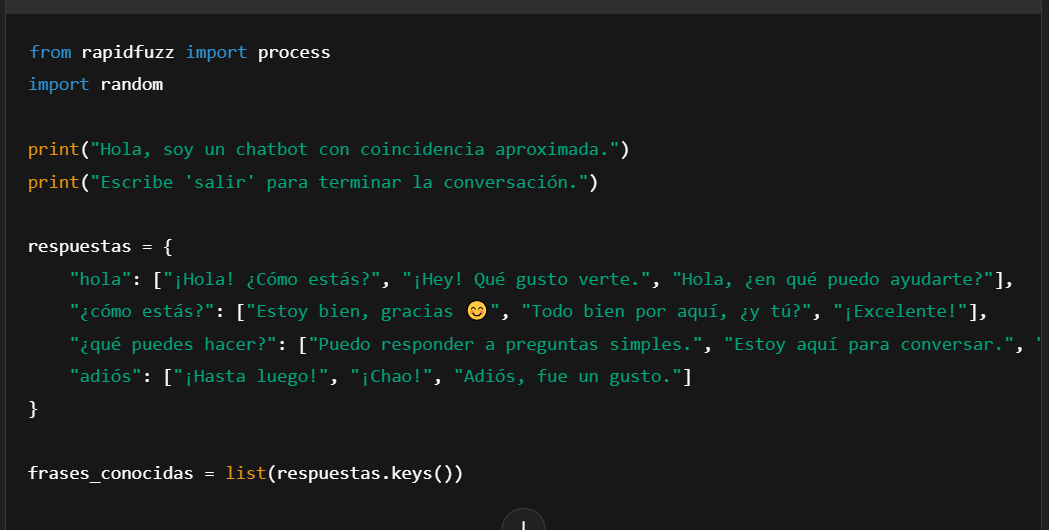
**✅ Paso 1: Instalar la librería**

Abre la terminal o consola y ejecuta:



**🧠 Paso 2: Código del chatbot con coincidencia aproximada**

Aquí tienes un chatbot que compara lo que escribe el usuario con todas las frases que conoce, y elige la más parecida si supera un **porcentaje mínimo de coincidencia** (por ejemplo, 80%).

  
Texto

El contenido generado por IA puede ser incorrecto.

**🔍 ¿Cómo funciona esto?**

* process.extractOne() compara lo que el usuario escribió con todas las frases del diccionario.
* Devuelve:
  + mejor\_coincidencia: la frase más parecida.
  + puntuacion: cuán parecida es, en porcentaje (de 0 a 100).
* Si la puntuación es **mayor o igual al 80%**, el chatbot responde como si fuera esa frase.

**✅ Ventajas:**

* El usuario no tiene que escribir frases exactas.
* Más flexible, sin usar inteligencia artificial completa.
* Aún fácil de mantener.