

CopilotKit 1.50 ya está disponible.

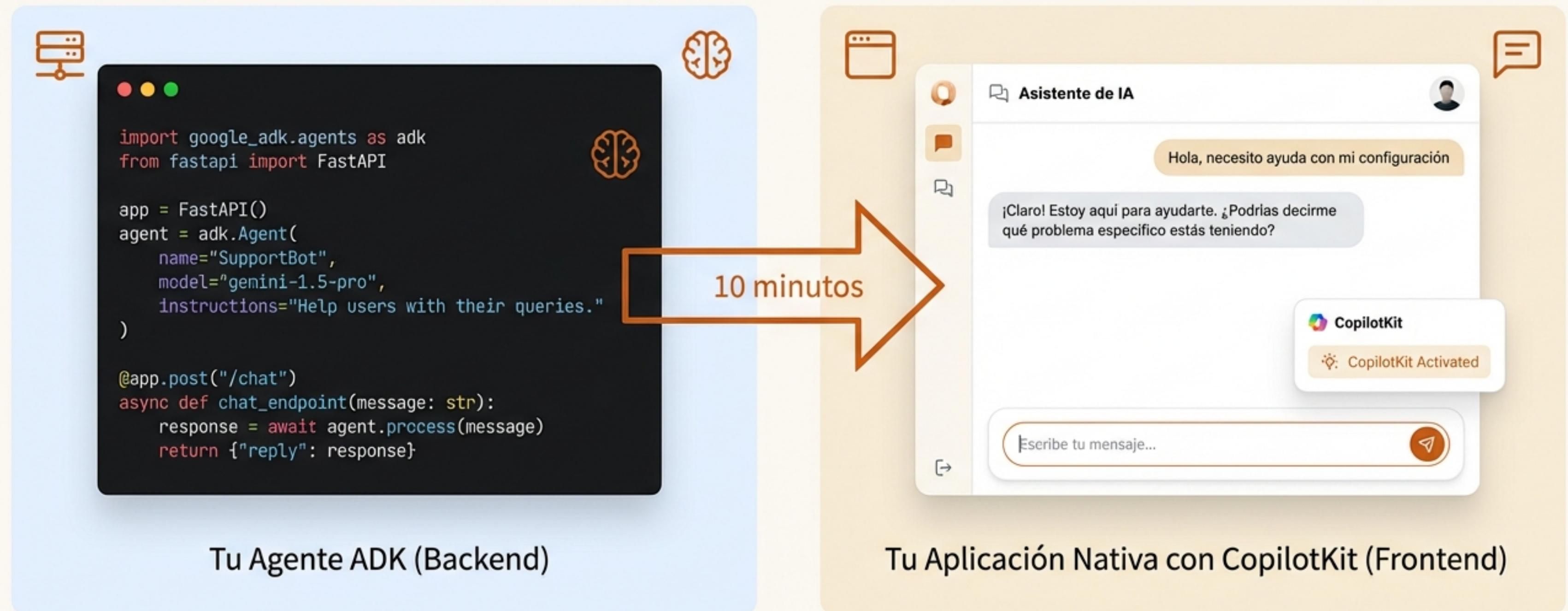
Transforma tus Agentes ADK en una aplicación nativa en 10 minutos.



[Ver Novedades](#)

De un agente backend a una aplicación de IA completa.

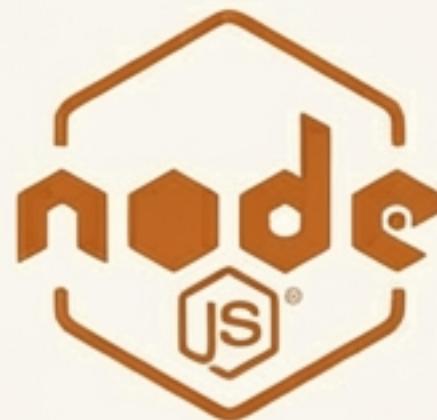
Vamos a tomar un agente de IA creado con el ADK de Google, que vive en un backend de Python, y le daremos una interfaz de chat nativa y potente usando CopilotKit en un frontend de React.



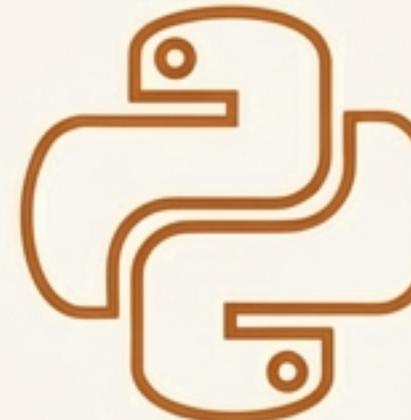
Requisitos para empezar el viaje.



Una clave de API de Google Gemini.



Node.js v20 o superior.



Python v3.9 o superior.



Tu gestor de paquetes favorito (npm, pnpm, yarn, bun).

Paso 1: Iniciamos el proyecto del agente.

Si aún no tienes un proyecto de Python, puedes crear uno rápidamente con **uv**, el gestor de paquetes de última generación.

```
$ uv init mi-agente  
$ cd mi-agente
```

Paso 2: Instalamos el soporte ADK para AG-UI.

Añadimos las librerías necesarias para que nuestro agente pueda comunicarse a través del protocolo AG-UI y ser servido con FastAPI.

```
$ uv add ag-ui-adk google-adk unicorn fastapi
```

¿Qué es exactamente AG-UI?

Frontend
(CopilotKit)

Backend
(Agente ADK)



AG-UI es un protocolo abierto diseñado para la comunicación entre un frontend y un agente de IA. El paquete `ag-ui-adk` que acabamos de instalar implementa este protocolo, creando un 'puente' estandarizado al que CopilotKit puede conectarse de forma nativa.

Paso 3: Exponemos el agente a través de FastAPI.

Actualizamos nuestro archivo `main.py` para instanciar el agente y exponerlo como una aplicación ASGI compatible con AG-UI.

```
# main.py
from fastapi import FastAPI
from ag_ui_adk import ADKAgent, add_adk_fastapi_endpoint ← El puente AG-UI
from google.adk.agents import LlmAgent

agent = LlmAgent(
    name="assistant",
    model="gemini-1.5-flash",
    instruction="Be helpful and fun!"
)

adk_agent = ADKAgent(
    adk_agent=agent,
    app_name="demo_app",
    user_id="demo_user",
    session_timeout_seconds=3600,
    use_in_memory_services=True
)

app = FastAPI()
add_adk_fastapi_endpoint(app, adk_agent, path="/") ← 3. Expón el agente en un endpoint de FastAPI
```

1. Define tu agente ADK estándar

2. Envuélvelo en un ADKAgent para AG-UI

3. Expón el agente en un endpoint de FastAPI

Nota: No olvides configurar tu clave de API: `export GOOGLE_API_KEY=tu_clave_de_google`.

Paso 4: Creamos nuestro frontend con Next.js.

CopilotKit funciona con cualquier frontend basado en React. Para este ejemplo, usaremos el popular framework Next.js.

```
$ npx create-next-app@latest mi-app-copilot  
$ cd mi-app-copilot
```

Paso 5: Instalamos los paquetes de CopilotKit.

Instalamos el core, la UI, el runtime y el cliente AG-UI para conectar todo.

npm

pnpm

yarn

bun

```
npm install @copilotkit/react-ui  
@copilotkit/react-core  
@copilotkit/runtime  
@ag-ui/client
```

Paso 6: Configuramos el Runtime para conectar con el agente.

Creamos una ruta API en Next.js.

Aquí es donde le decimos a CopilotKit dónde encontrar nuestro agente ADK. `HttpAgent` es la clave que apunta a nuestro backend en `http://localhost:8000`.

```
// app/api/copilotkit/route.ts
import { CopilotRuntime } from "@copilotkit/runtime";
import { HttpAgent } from "@ag-ui/client"; // <-- El cliente AG-UI
import { NextRequest } from "next/server";

const runtime = new CopilotRuntime({
  agents: {
    // Apuntamos a nuestro backend de Python
    my_agent: new HttpAgent({ url: "http://localhost:8000/" }),
  }
});

// Endpoint estándar de Next.js App Router
export const POST = /* ... boilerplate ... */ ;
```

Paso 7: Integramos CopilotKit en nuestra aplicación.

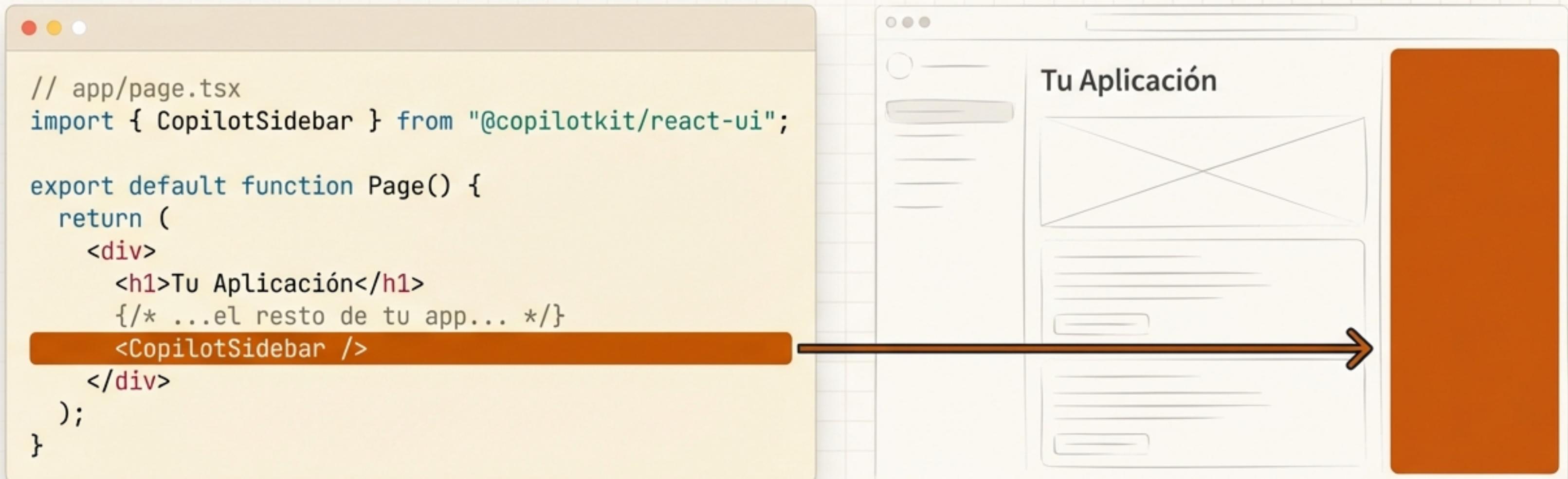
Envolvemos la raíz de nuestra aplicación con el componente `CopilotKit`, haciendo que sus funcionalidades estén disponibles en todo el árbol de componentes.

```
// app/layout.tsx
import { CopilotKit } from "@copilotkit/react-core";
import "@copilotkit/react-ui/styles.css"; ← No olvides los estilos

export default function RootLayout({ children }: { children: React.R
  return (
    <html lang="en">
      <body>
        <CopilotKit url="/api/copilotkit">
          {children}
        </CopilotKit>
      </body>
    </html>
  );
}
```

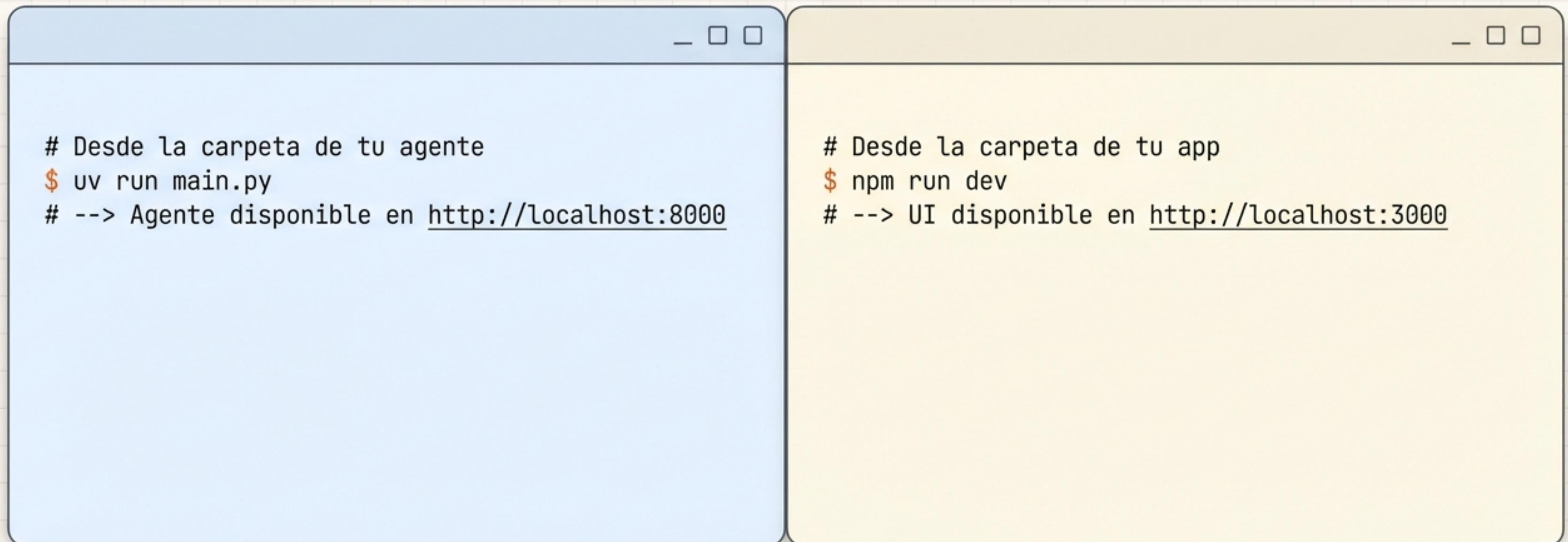
Paso 8: Añadimos la interfaz de chat.

Con todo configurado, añadir la interfaz de usuario es tan simple como importar y renderizar un componente: `CopilotSidebar`.



¡Es la hora de la verdad! Lanza tu agente y tu UI.

En dos terminales separadas, inicia el servidor del agente y el servidor de desarrollo del frontend.



The image shows two separate terminal windows side-by-side. The left terminal has a light blue background and displays the following text:

```
# Desde la carpeta de tu agente
$ uv run main.py
# --> Agente disponible en http://localhost:8000
```

The right terminal has a light yellow background and displays the following text:

```
# Desde la carpeta de tu app
$ npm run dev
# --> UI disponible en http://localhost:3000
```



¡Empieza a chatear!

¡Felicitaciones! Tu agente ADK ahora tiene una interfaz de chat nativa. Abre <http://localhost:3000> en tu navegador y prueba a preguntarle algo.

localhost:3000

CopilotSidebar

¿Puedes contarme un chiste?

¿Puedes ayudarme a entender la IA?

¿Qué opinas sobre React?

sum dolor sit amet,
ir uasiedion elt aorsi ait sed
iioro magna incislidunt. Ut
exzl ervollpat. Exeit ano
nrintand eanrrt:aaes avis
ilon amenta de contrnuto

auta. Ur cieysc, tibdtobisint in cuips pressibuln t enliptis velit nut dolorer magna
tempat nritr ut aliquip ex cammodo consequit.

NotebookLM

Tu viaje con CopilotKit acaba de empezar.

Has visto lo fácil que es integrar una UI en un agente existente.
Ahora puedes explorar más a fondo.



**Explora las novedades
de la v1.50**

[Ver el 'changelog' completo](#)



Empieza desde cero

¿Prefieres una plantilla lista para usar?

[Prueba nuestro 'starter' oficial](#)