

Módulo 2. IA generativa como asistente de código fuente de programación



☰ 1. Paradigmas modernos de desarrollo y asistencia al código

☰ 2. Asistentes de código fuente (code assistants)

☰ 3. Funcionamiento vite code vs. 8n8 y Make

☰ Descarga en PDF

1. Paradigmas modernos de desarrollo y asistencia al código

Vibe coding, *no-code* y *low-code* son enfoques de desarrollo de *software* que buscan aumentar la velocidad y la accesibilidad, principalmente reduciendo la necesidad de programación tradicional. Sin embargo, lo hacen de diferentes maneras y suelen ser adecuados para distintos usuarios y tipos de proyectos.

Vibe coding

Es un enfoque moderno (popularizado y usado por Walter Agüero desde 2023 en clases de grado y posgrado en las universidades UBA, Siglo 21, Universidad Nacional de Villa Mercedes y Universidad Nacional de Río Negro en Argentina). Usa inteligencia artificial (específicamente modelos de lenguaje grandes o LLM) para generar *software* a partir de indicaciones en lenguaje natural. Los usuarios describen la funcionalidad que desean en español, inglés,

etcétera, de forma sencilla, y la IA genera el código o la aplicación correspondiente.

Tiene varias características que hacen que sea una gran herramienta de aprendizaje que mejora los tiempos de aprendizaje y brinda soluciones de forma rápida, las cuales siempre deben ser validadas por expertos en el área.

Algunas características de vide coding son las siguientes:

- lo pueden aplicar usuarios no técnicos para la creación rápida de prototipos de ideas simples, o desarrolladores experimentados como un potente asistente de IA para automatizar código repetitivo y agilizar su flujo de trabajo.
- El usuario guía a la IA de forma conversacional y se centra en la intención y los resultados generales, a menudo sin revisar ni comprender completamente cada línea del código generado (en su forma «pura»).

- Es ideal para lograr prototipado rápido, proyectos personales sencillos y automatización de tareas de codificación repetitivas.

Algunas herramientas que generan *vibe coding* son ChatGPT, DeepSeek, Copilot, etcétera.

Ejemplo con ChatGPT

- **Pedido:** «haga un programa en lenguaje Java que tome una lista con los números enteros 1, 2, 3, 4, 5, 6 y que devuelva solo los pares elevados al cuadrado».
- Se espera que el programa devuelva números pares elevados al cuadrado: [4, 16, 36].

Figura 1: Pedido solicitado a ChatGPT y respuesta del código de programa

Realice un programa en lenguaje java que tome una lista con los números de números enteros 1, 2, 3, 4, 5, 6 y devuelva solo los pares elevados al cuadrado

Fuente: captura de pantalla de ChatGPT.

Figura 2: Código fuente Java

✓ Código en Java

```
java Copy code  
  
import java.util.*;  
import java.util.stream.Collectors;  
  
public class NumerosParesCuadrado {  
    public static void main(String[] args) {  
        // Lista de números enteros  
        List<Integer> numeros = Arrays.asList(1, 2, 3, 4, 5, 6);  
  
        // Filtrar pares y elevarlos al cuadrado  
        List<Integer> paresCuadrados = numeros.stream()  
            .filter(n -> n % 2 == 0) // Filtra los pares  
            .map(n -> n * n) // Eleva al cuadrado  
            .collect(Collectors.toList()); // Guarda en una lista nueva  
  
        // Mostrar el resultado  
        System.out.println("Números pares elevados al cuadrado: " + paresCuadrados);  
    }  
}
```

Fuente: elaboración propia.

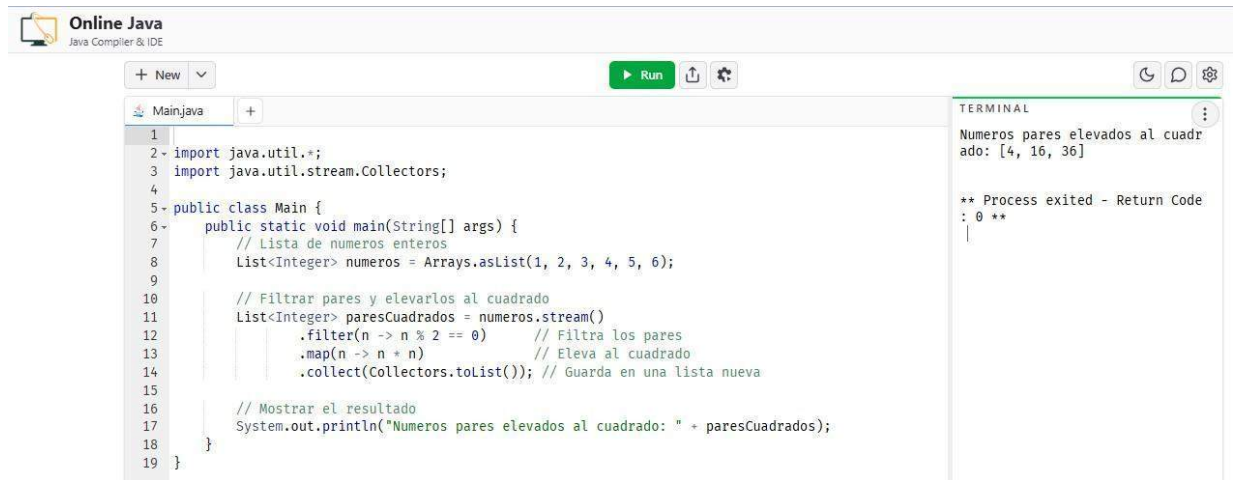
Además, devuelve la siguiente explicación:

- `filter(n -> n % 2 == 0)` selecciona solo los números pares.
- `map(n -> n * n)` transforma cada número par en su cuadrado.
- `collect(Collectors.toList())` convierte el flujo (*stream*) que es producto de una lista.

Haciendo un pequeño cambio en el texto del código fuente Java de la figura 2, en la que se reemplaza el texto que dice `NumerosParesCuadrado` por la palabra **Main**, podremos ejecutarlo haciendo clic en el siguiente enlace: <https://www.online-java.com/Ciyt8q536I>.

Cuando el alumno haga el clic sobre el enlace, observará el cambio del texto mencionado (ver figura 3, línea 5). Este programa podrá ejecutarlo si se presiona el botón de color verde: Run.

Figura 3: Botón verde Run



The screenshot shows an online Java IDE interface. At the top, there is a title bar with 'Online Java' and 'Java Compiler & IDE'. Below the title bar, there is a toolbar with a '+ New' button, a green 'Run' button, and other icons. The main area is divided into two panes. The left pane shows a code editor with a file named 'Main.java'. The code is as follows:

```
1
2- import java.util.*;
3 import java.util.stream.Collectors;
4
5- public class Main {
6-     public static void main(String[] args) {
7         // Lista de numeros enteros
8         List<Integer> numeros = Arrays.asList(1, 2, 3, 4, 5, 6);
9
10        // Filtrar pares y elevarlos al cuadrado
11        List<Integer> paresCuadrados = numeros.stream()
12            .filter(n -> n % 2 == 0) // Filtra los pares
13            .map(n -> n * n) // Eleva al cuadrado
14            .collect(Collectors.toList()); // Guarda en una lista nueva
15
16        // Mostrar el resultado
17        System.out.println("Numeros pares elevados al cuadrado: " + paresCuadrados);
18    }
19 }
```

The right pane is a terminal window titled 'TERMINAL'. It shows the output of the program: 'Numeros pares elevados al cuadrado: [4, 16, 36]'. Below the output, it shows the process exit information: '** Process exited - Return Code : 0 **'.

Fuente: elaboración propia.

Nota: como se ha observado, hemos combinado conceptos mencionados en la lectura 1 (cuando vimos las IDE *online*, también conocidas como marcos de trabajo o *frameworks*). Asimismo, se ejecutó un programa hecho en lenguaje de programación Java usando *vibe code* por medio de ChatGPT.

No-code

El paradigma *no-code* permite crear aplicaciones sin escribir código, lo que lo hace ideal para usuarios que no cuentan con conocimientos técnicos. Se basa en interfaces gráficas y componentes reutilizables, es decir, desarrolla, al igual que *vibe code*, aplicaciones sin escribir código, pero usando interfaces visuales y arrastrar-soltar.

Las características principales de *no-code* son las siguientes:

- *drag and drop* de elementos. Emplea un entorno de desarrollo completamente visual, generalmente con interfaces de arrastrar y soltar, componentes prediseñados y bloques lógicos predefinidos, para crear aplicaciones sin escribir código.
- Diseñado para desarrolladores no técnicos y usuarios empresariales sin conocimientos técnicos (por ejemplo: profesionales del *marketing*, propietarios de pequeñas empresas, equipos de operaciones) que desean crear aplicaciones de forma rápida y sencilla con una curva de aprendizaje mínima.

- Integración con API mediante conectores visuales.
- Abstracción de todo el código y la complejidad técnica, centrándose por completo en la configuración visual y los flujos de trabajo.

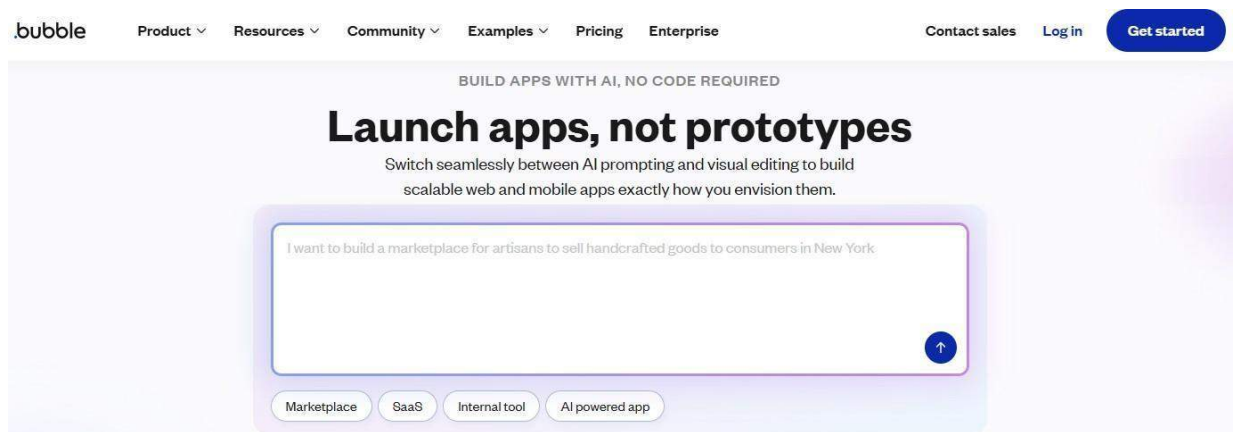
Existen algunas plataformas *no-code*, como, por ejemplo, las siguientes:

- Bubble (*app web*).
- Zapier (automatización).
- Webflow (*sitios web*).

Es ideal para prototipado rápido y democratización del desarrollo, pero siempre el experto debe avalarlo, dado que puede contener algún tipo de error.

Se muestra seguidamente la aplicación web Bubble (<https://bubble.io/>).

Figura 4: Portada del sitio web Bubble



Fuente: captura de pantalla de Bubble.

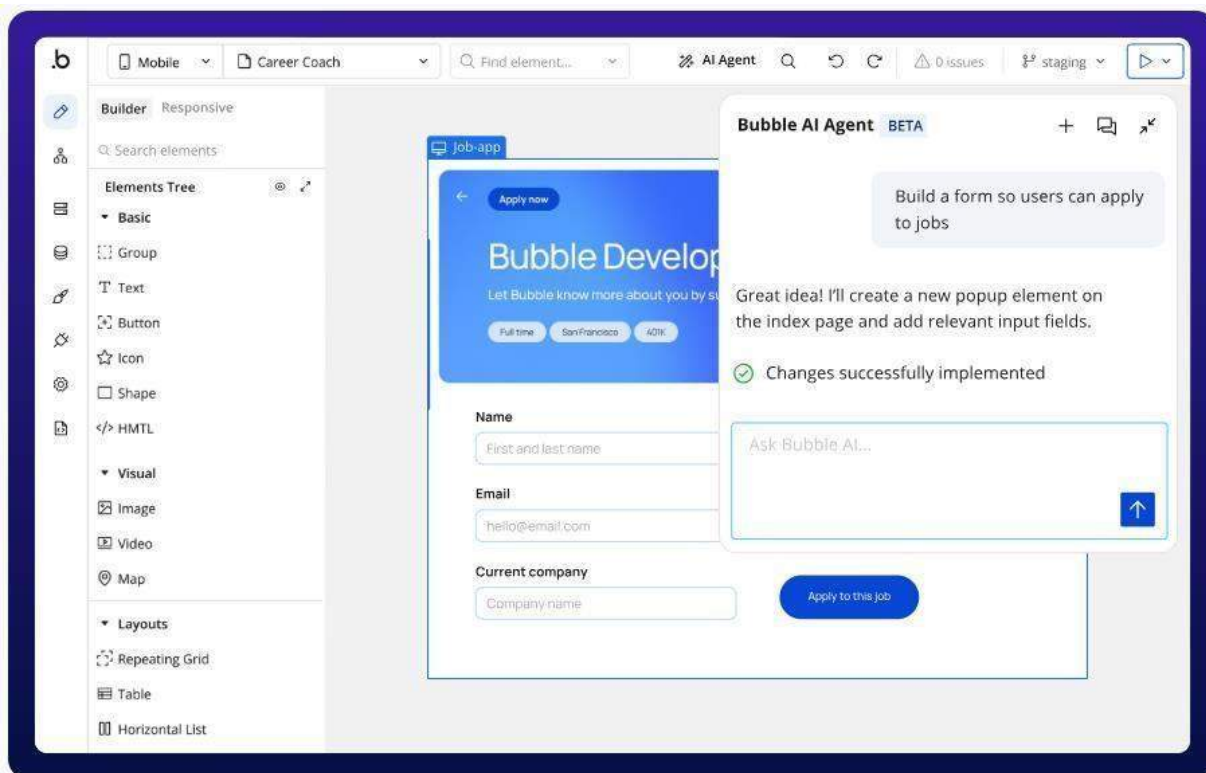
Tiene una versión gratis (*free*) que se puede visualizar en la opción Pricing con ciertas limitaciones, pero es posible apreciar el componente que se puede llevar a cabo.

Usando la opción IA Agent, el usuario puede solicitarle escribiendo en lenguaje natural a la IA de esta herramienta

que diseñe un formulario en el que los usuarios ingresen datos (ver figura 5).

En lugar de usar IA Agente, el diseñador u operador de esta herramienta podrá armar un formulario personalizado arrastrando los iconos de la izquierda como text, button, etcétera (ver figura 5).

Figura 5: Ambiente de trabajo de Bubble



Fuente: captura de pantalla de Bubble.



Nota: la principal diferencia entre el sistema *no-code* y el *vibe code* es que, en *no-code*, como en el caso de Bubble, no es posible ver o modificar el código fuente, es decir, no se puede exportar el código fuente generado por [Bubble.io](https://bubble.io) directamente desde la plataforma, dado que está diseñado para crear aplicaciones web y prototipos sin código, no para exportar el código subyacente. En cambio, en los sistemas *vibe code*, es natural la generación de código y, como caso particular, el usuario podrá generar el mismo formulario de la figura 5.

Low-code

Low-code es un enfoque intermedio; permite desarrollar aplicaciones con poco código, combinando bloques visuales con *scripts* personalizados, es decir, es un enfoque híbrido que ofrece una interfaz visual e intuitiva, similar a la de las plataformas sin código, pero que también permite a los desarrolladores profesionales insertar código personalizado (como, por ejemplo, JavaScript, HTML, CSS) para añadir funcionalidades específicas, integraciones o personalizaciones avanzadas que van más allá de las capacidades predeterminadas de la plataforma.

Sus características principales son las siguientes:

- permite el acceso a usuarios sin conocimientos técnicos y desarrolladores profesionales, lo que ofrece un equilibrio entre velocidad y flexibilidad.
- Proporciona mayor control y personalización que las plataformas sin código, a la vez que acelera el desarrollo en comparación con la

programación tradicional con código completo.

- Aplicaciones empresariales complejas y escalables, sistemas que requieren integraciones específicas y proyectos en los que la velocidad es importante, pero en los que también se requiere control total y flexibilidad.

Es útil en entornos empresariales en los que se requiere rapidez y flexibilidad, como, por ejemplo, los siguientes:

- OutSystems.
- Mendix.
- Microsoft Power Apps.

Nota

Low-code vs. no-code: la principal diferencia entre las plataformas para desarrollo *low-code* y *no-code* es cuánto conocimiento de codificación necesita el usuario. Las plataformas para desarrollo *low-code* (LCDP) requieren de algunas habilidades básicas de codificación para poder desarrollar e integrar aplicaciones complejas, mientras que las de *no-code* (NCDP) no requieren ninguna. Debido a que la mayoría de las organizaciones tienen una amplia variedad de conjuntos de habilidades técnicas dentro de su fuerza laboral, muchas plataformas ofrecen tanto herramientas *low-code* como *no-code*.

Tabla 1: Comparativa *vibe coding*, *no-code* y *low-code*

| Característica | <i>Vibe coding</i> | <i>No-code</i> | <i>Low-code</i> |
|-----------------------------|---|--|--|
| Entrada principal | Instrucciones en lenguaje natural(texto/voz). | Interfaz visual de arrastrar y soltar. | Interfaz visual más fragmentos de código personalizados. |
| Dependencia de la IA | Gran dependencia de | Puede usarse la IA para | Puede usarse la IA como |

| | | | |
|--------------------------------------|---|---|--|
| | la IA para la generación de código. | sugerencias/flujo de trabajo, pero la construcción principal es visual. | asistencia, pero la codificación humana es fundamental para las funciones avanzadas. |
| Conocimientos de programación | Mínimos o nulos para empezar | No se requieren conocimientos de programación | Algunos conocimientos de programación son beneficiosos para la Personalización. |
| Personalización | Alta flexibilidad mediante código generado editable. | Limitado a los componentes predefinidos de la plataforma. | Alta flexibilidad gracias a la capacidad de codificación personalizada. |
| Mejores casos de uso | Prototipado rápido, experimentos ágiles, tareas sencillas | Herramientas internas simples, aplicaciones básicas, MVP | Aplicaciones empresariales escalables, integraciones complejas, funcionalidades específicas. |

Fuente: elaboración propia.

CONTINUAR

2. Asistentes de código fuente (code assistants)

Son herramientas basadas en inteligencia artificial que ayudan a escribir, corregir y entender código.

Funciones comunes

- Autocompletado inteligente
 - Generación de funciones a partir de descripciones
 - Refactorización automática
 - Explicación de código existente
-

Ejemplos

- GitHub Copilot.
- Amazon CodeWhisperer.
- Tabnine.

Estos asistentes están transformando la enseñanza de la programación, lo que les permite a los estudiantes centrarse en la lógica y el diseño más que en la sintaxis.

Aplicaciones n8n y Make

Make es una plataforma de automatización más fácil de usar, ideal para principiantes y automatizaciones sencillas, la cual cobra por «operación». Por otro lado, n8n es más potente y flexible, con opciones de autoalojamiento, lo que la hace más económica para proyectos complejos y escalables. Sin embargo, tiene una curva de aprendizaje más pronunciada.

La elección depende del nivel técnico de alumno y de las necesidades del proyecto que se esté haciendo.

La sugerencia que se deja es elegir Make para rapidez y simplicidad, o n8n para mayor control, personalización y ahorro a gran escala.

En resumen, se describen las características de las dos herramientas.

Make

- Ideal para principiantes, validación de ideas y automatizaciones simples.
- **Facilidad de uso:** curva de aprendizaje más sencilla con una interfaz de arrastrar y soltar y muchas integraciones preconstruidas.
- **Precio:** se basa en el número de «operaciones» (acciones individuales dentro de un flujo de trabajo), lo que puede volverse caro en automatizaciones complejas.

- **Flexibilidad:** menos flexible que n8n; las personalizaciones avanzadas pueden requerir más esfuerzo o no ser posibles.

n8n

- Ideal para usuarios técnicos, proyectos que necesitan escalabilidad y un control total.
- Facilidad de uso: mayor dificultad de aprendizaje debido a su flexibilidad y a la necesidad de trabajar con código en algunos casos.
- Precio: puede autoalojarse gratis (pagando solo por el servidor) o su versión en la nube cobra por «ejecución» (el flujo de trabajo completo), lo que suele ser más económico para el uso a gran escala.
- Flexibilidad: muy flexible; permite añadir código personalizado (JavaScript), manejar API complejas e incluso permite el

autoalojamiento, lo que le da un control completo sobre los datos.

- En Argentina, el sitio de *hosting* [Hostinger.com](https://www.hostinger.com) ofrece el servicio de 8n8 con *hosting*, pero sin dominio a ochenta y tres mil pesos mensuales o a doscientos veinte mil pesos anuales (USD 153, valores tomados el 1/11/2025) y no hay cargos por visitantes ni por el uso de la *app* publicada en el sitio, solo por los mensajes que se usa para construirla.

El sitio web de **8n8** hace una descripción comparativa con Make en el siguiente enlace: <https://n8n.io/vs/make/>

8n8

8n8 (n8n) se describe como una plataforma *low-code*. Combina la facilidad de uso de una interfaz visual de arrastrar y soltar (característica del *no-code*) con la flexibilidad de poder añadir código personalizado (JavaScript o Python) cuando se necesita una lógica más avanzada.



N8N es una plataforma de automatización de flujos de trabajo de código abierto y de bajo código que permite conectar aplicaciones y servicios para automatizar tareas sin necesidad de programar. Sirve para sincronizar datos entre diferentes sistemas, automatizar tareas repetitivas como el envío de correos, crear flujos de trabajo personalizados, extraer y procesar datos web, o integrar herramientas de inteligencia artificial para tareas más complejas.

Esto significa lo siguiente:

- los usuarios no técnicos pueden crear flujos de trabajo de automatización completos sin escribir código (enfoque *no-code*).
- Los desarrolladores pueden ampliar las funcionalidades, crear integraciones personalizadas o aplicar lógica compleja mediante código para adaptarse a necesidades muy específicas, lo que la convierte en una herramienta potente para equipos técnicos (enfoque *low-code*).

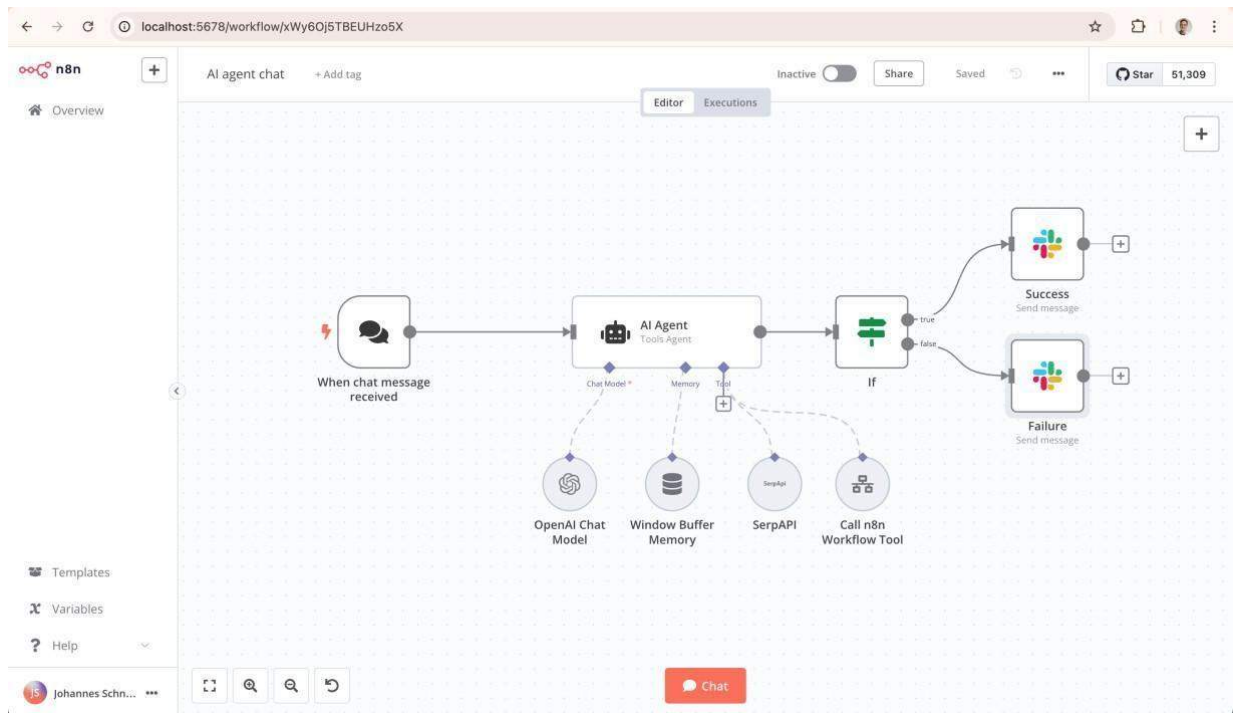
En su sitio web, n8n especifica los siguientes aspectos

(<https://n8n.io/pricing/>):

- **servidor propio para autohospedaje.** Sí, la edición gratuita Community Edition incluye un editor de flujo de trabajo completo. Los planes de pago Business y Enterprise ofrecen colaboración ampliada y funcionalidades de nivel empresarial.
- **Prueba gratuita:** sí, cuenta con una prueba gratuita de catorce días para los planes en la nube. Además, la edición Community Edition autohospedada proporciona acceso gratuito y permanente a todas las funciones principales, sin límite de tiempo.

En el mismo sitio web, se muestra el enlace a GitHub para clonarlo: <https://github.com/n8n-io/n8n>.

Figura 6: n8n

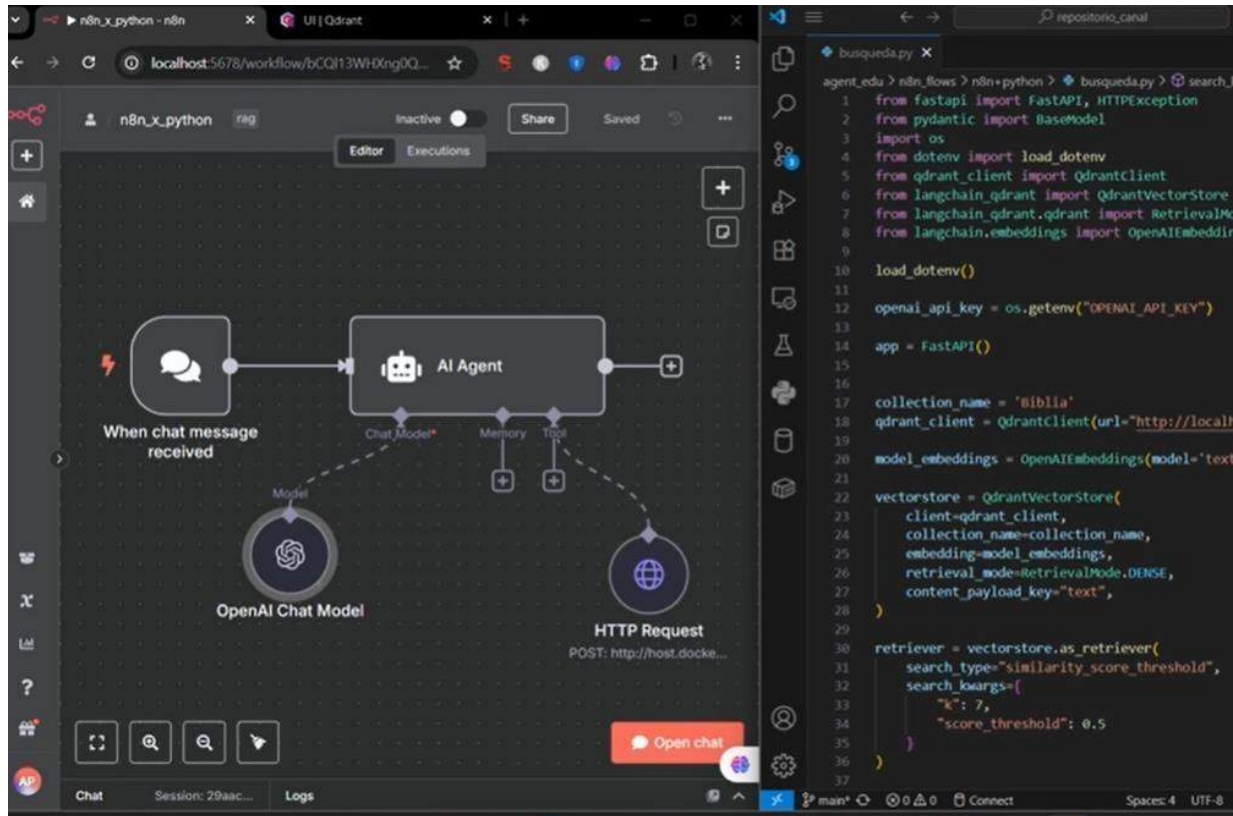


Fuente: captura de pantalla de GitHub.

En resumen, n8n ofrece lo mejor de ambos mundos, gracias a lo cual los usuarios pueden elegir el enfoque que mejor se adapte a su nivel de habilidad y a la complejidad del proyecto.

Otro punto a tener en cuenta es que es posible convertir el *workflow* en Python en poco tiempo usando la aplicación n8n2py.me con un costo que está en la página web.

Figura 7: n8npy.me



Fuente: captura de pantalla de n8npy.me.

Make

Make (<https://www.make.com/en>) es una herramienta de automatización visual, que permite conectar entre sí más de mil quinientas herramientas con lógica compleja.

Es una herramienta que, debido a sus funcionalidades, tiene una curva de aprendizaje mayor que Zapier (Zapier es una

herramienta de automatización que conecta aplicaciones en línea para que se comuniquen y hagan tareas entre sí sin necesidad de codificar) y similares, pero unas posibilidades infinitamente mayores (y a un costo mucho más reducido).

Resumidamente, Make es una herramienta para automatizar e integrar todas tus herramientas.

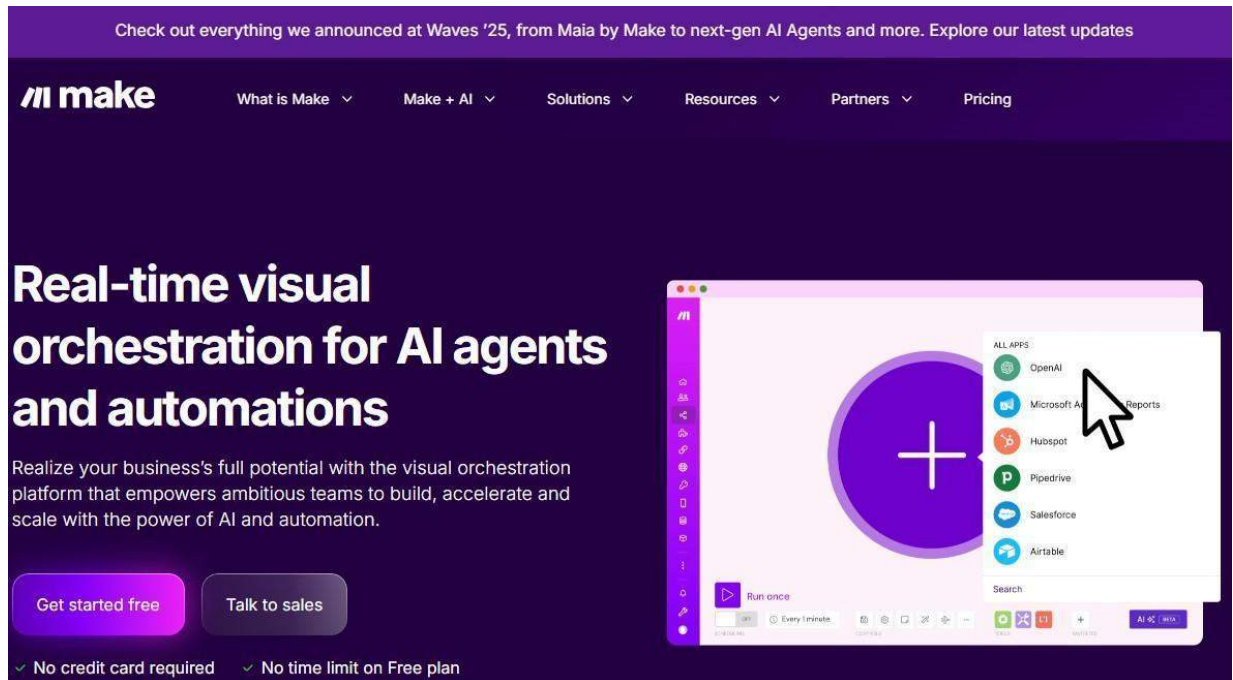
Funcionalidades y características

Make permite conectar mediante un sistema visual (basado en módulos) herramientas que tengan una API.

Cuentan con una serie de más de mil quinientos módulos predefinidos, pero con la posibilidad de conectarse a cualquier herramienta mediante su API gracias al módulo de HTTP Request.

Gracias a sus funcionalidades incorporadas, y usando terminología de esta herramienta, se puede crear lógica condicional con los routers y los filtros, para conseguir crear el flujo que se necesita. Además, se pueden programar las ejecuciones de los escenarios y usar sus webhooks de manera muy sencilla.

Figura 8: Make en Internet



Fuente: captura de pantalla de Make (<https://www.make.com/en>).

Funciona mediante operaciones y tasas de transferencias de datos; es un modelo de pago por uso, según las operaciones que se lleven a cabo (cada módulo es una operación). Por lo tanto, se pagará más o menos. Cuenta, además, con un plan gratuito que puede servir para iniciarse y un plan básico muy suficiente para la mayoría de las aplicaciones.

El grado de personalización y control que se tiene sobre cada escenario y módulo hace que sea muy práctica la automatización con esta herramienta, lo que la hace un gran

apoyo a las operaciones de cualquier empresa emergente (startup) o proyecto.

Además, es una herramienta muy visual que te permite tener un control muy preciso de cómo va tu automatización, y permite detectar errores y solucionarlos de manera muy sencilla gracias a su sistema de gestión de errores.

Pros y contras de Make

- **Lógica condicional incorporada:** la propia naturaleza de su constructor visual hace que sea muy sencillo crear flujos complejos.
- **Flujos ilimitados:** puedes crear escenarios tan largos como necesites sin costes adicionales.
- Los planes de *pricing* tienen un costo.
- Gestión de errores: es muy sencillo identificar y corregir errores en tus escenarios.
- **Webhooks** y **HTTP request:** te puedes conectar con cualquier herramienta que

disponga de API.

- Regex y funciones: puedes añadir una capa extra con las funciones e incluso tratar los datos con Regex si quieres más control aún.

CONTINUAR

3. Funcionamiento vibe code vs. 8n8 y Make

Seguidamente, se explicará en un nivel abstracto cómo funcionan estas herramientas. Esta explicación le permitirá al alumno entender cómo ahorra tiempo su uso.

Usando *vibe code*

Paso 1: usando los conceptos de *vibe code* expuestos anteriormente con DeepSeek, solicitamos a la IA por medio del *prompt* lo que se puede observar en la siguiente figura.

Figura 9: DeepSeek

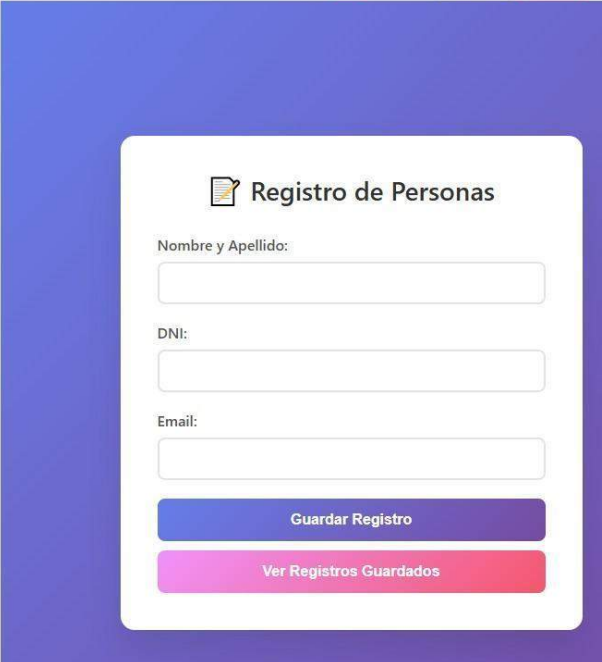


Desarrolle un programa en Java que usando un formato gráfico como JFrame, un usuario ingrese por teclado: a) nombre y apellido, b) dni y c) mail y luego lo guarde en una tabla personas de la base de datos poblacion

Fuente: captura de pantalla de Deepseek.

Vibe code: DeepSeek devuelve el código HTML del formulario de la figura 9. El alumno puede verlo y ejecutarlo haciendo clic en <https://onecompiler.com/html/443kxu8qx>.

Figura 10: Formulario generado por *vibe code*



```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Registro de Personas - n8n</title>
7   <style>
8     * {
9       margin: 0;
10      padding: 0;
11      box-sizing: border-box;
12    }
13
14    body {
15      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
16      background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
17      min-height: 100vh;
18      display: flex;
19      justify-content: center;
20      align-items: center;
21      padding: 20px;
22    }
23
24    .container {
25      background: white;
26      border-radius: 15px;
27      box-shadow: 0 20px 40px rgba(0,0,0,0.1);
28      padding: 40px;
29      width: 100%;
30      max-width: 500px;
31    }
32
33    h1 {
34      color: #333;
35      text-align: center;
36      margin-bottom: 30px;
37      font-size: 28px;
38      font-weight: 600;
39    }
40
41    .form-group {
42      margin-bottom: 20px;
```

Fuente: captura de pantalla de OneCompiler.

Paso 2: Creación tabla en una base de datos existentes

Luego, se puede crear la base de datos por medio de algún motor de base de datos. Abajo, se sugiere usar MySQL y el conector para establecer conexión desde la aplicación. Toda esta información es aportada por pedido hecho desde el *prompt*.

Figura 11: Líneas de comando para crear la tabla personas de una base de datos población

Primero, crea la base de datos y tabla

```
sql

-- Crear base de datos (si no existe)
CREATE DATABASE IF NOT EXISTS poblacion;

-- Usar La base de datos
USE poblacion;

-- Crear tabla personas
CREATE TABLE IF NOT EXISTS personas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre_apellido VARCHAR(100) NOT NULL,
    dni VARCHAR(20) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL,
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Fuente: elaboración propia.

Paso 3: Generación de código que maneja el formulario (figura 9) de ingreso de datos

DeepSeek también aporta el código fuente para ingresar los datos y lo construye en ciento noventa y siete líneas de código, las cuales pueden verse en su totalidad en el siguiente enlace: <https://www.online-java.com/oaj7DP8CKq>.

Figura 12: Código fuente Java generado por DeepSeek

The screenshot shows an online Java IDE interface. At the top, the URL 'online-java.com/oaj7DP8CKq' is visible. Below it, there's a header indicating '197 líneas de código' (197 lines of code) and 'Created yesterday • Never expires'. The main area is split into two panes. The left pane shows a code editor with a file named 'Main.java' containing 23 lines of Java code. The code defines a class 'RegistroPersonasMejorado' that extends 'JFrame'. It includes imports for 'javax.swing.*', 'java.awt.*', 'java.awt.event.*', 'java.sql.*', and 'java.util.regex.Pattern'. The class has private fields for 'JTextField' (txtNombreApellido, txtDNI, txtEmail) and 'JButton' (btnGuardar, btnLimpiar, btnVerRegistros). It also has static final fields for database configuration: 'URL', 'USER', and 'PASSWORD'. The class has a constructor 'RegistroPersonasMejorado()' that calls 'inicializarInterfaz()', and a private void method 'inicializarInterfaz()' that sets the window title to 'Sistema de Registro de Personas' and the default close operation to 'EXIT_ON_CLOSE'. The right pane is a terminal window titled 'TERMINAL' showing the output: 'Welcome to online-java.com!' and 'Run the code and see the output here.'

```
1- import javax.swing.*;
2- import java.awt.*;
3- import java.awt.event.*;
4- import java.sql.*;
5- import java.util.regex.Pattern;
6-
7- public class RegistroPersonasMejorado extends JFrame {
8-     // Componentes de la interfaz
9-     private JTextField txtNombreApellido, txtDNI, txtEmail;
10-    private JButton btnGuardar, btnLimpiar, btnVerRegistros;
11-
12-    // Configuración BD
13-    private static final String URL = "jdbc:mysql://localhost:3306/pobl";
14-    private static final String USER = "root";
15-    private static final String PASSWORD = "";
16-
17-    public RegistroPersonasMejorado() {
18-        inicializarInterfaz();
19-    }
20-
21-    private void inicializarInterfaz() {
22-        setTitle("Sistema de Registro de Personas");
23-        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

Fuente: captura de pantalla de DeepSeek.

Paso 4: Dependencias necesarias

Es necesario instalar el conector que permitirá guardar los datos ingresados en la tabla descritas en los puntos anteriores.

Figura 13: Respuesta que dio DeepSeek

Dependencias necesarias

Necesitarás el conector JDBC de MySQL. Agrégalo a tu proyecto:

Maven:

```
xml Copy Download Run  
  
<dependency>  
  <groupId>mysql</groupId>  
  <artifactId>mysql-connector-java</artifactId>  
  <version>8.0.33</version>  
</dependency>
```

Gradle:

```
gradle Copy Download  
  
implementation 'mysql:mysql-connector-java:8.0.33'
```

O descarga manualmente:

- Descarga el JAR desde: <https://dev.mysql.com/downloads/connector/j/>
- Agrégalo al classpath de tu proyecto

Fuente: captura de pantalla de DeepSeek.

Resumidamente, las características del programa entregado son las siguientes:

- interfaz gráfica con JFrame.
- Validación de campos obligatorios.

- Validación de formato de *e-mail*.
- Manejo de errores de base de datos.
- Prevención de duplicados por DNI.
- Limpieza automática de campos después de guardar.
- Función para ver registros existentes.
- Manejo seguro de conexiones a BD.

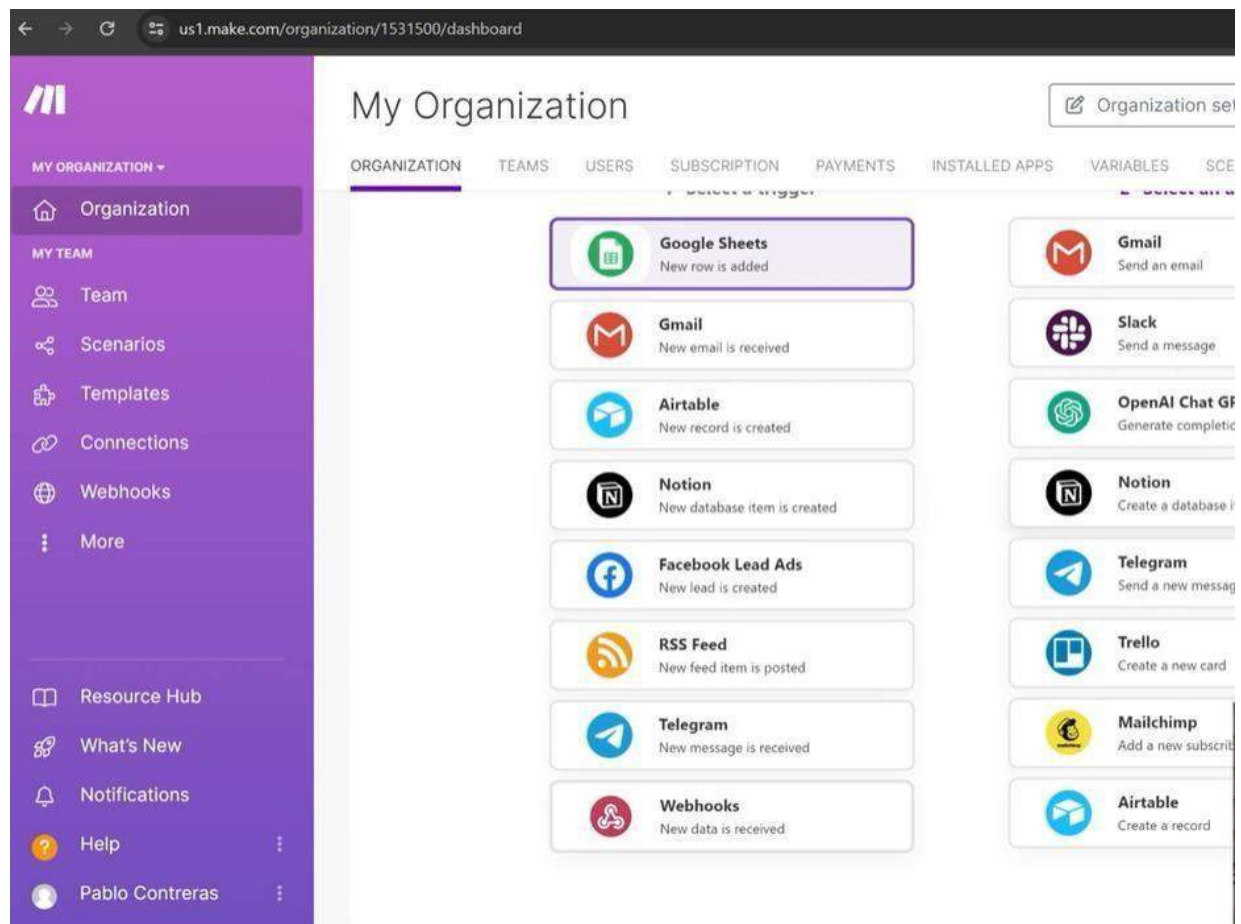
Si bien el programa está listo para usar, puede que haya que ajustarlo a la conexión con la base de datos y alguna que otra característica. En relación con esto, hay que asegurarse de contemplar los siguientes aspectos:

- tener MySQL instalado y en ejecución.
- Crear la base de datos y tabla como se indica.
- Actualizar las credenciales de conexión en el código.
- Tener el conector JDBC en el *classpath*.

Usando Make

Si bien no es exactamente el mismo ejemplo que se mostró en el punto anterior, se presentará con base en otro formulario y cómo con simples acciones es posible crearlo.

Figura 14: *Dashborad* obtenida de Internet



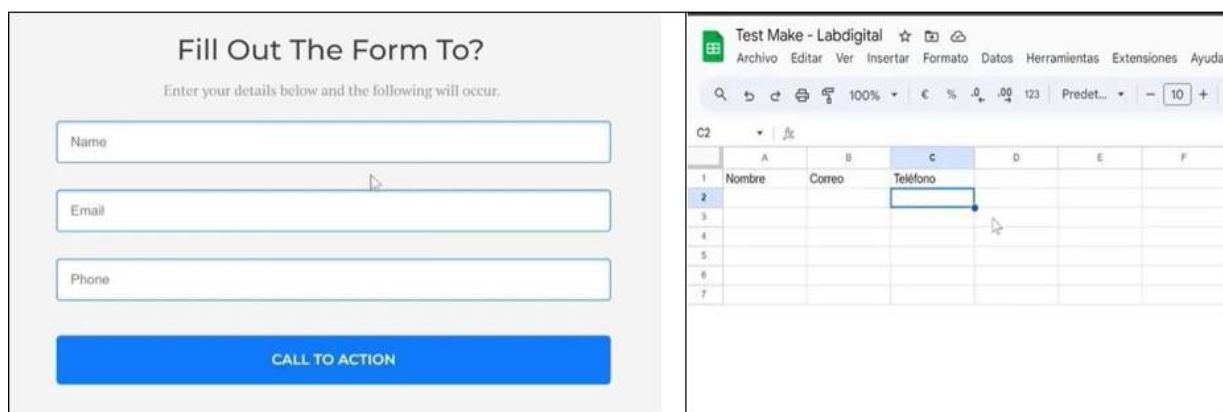
Fuente: captura de pantalla de us1.make.com.

La idea consiste en que cada vez que alguien llena los datos, los envía presionando el botón Call to action. Estos van a parar a una planilla de cálculo de Google Sheets.

Para llevar a cabo esta tarea, se hacen los siguientes pasos.

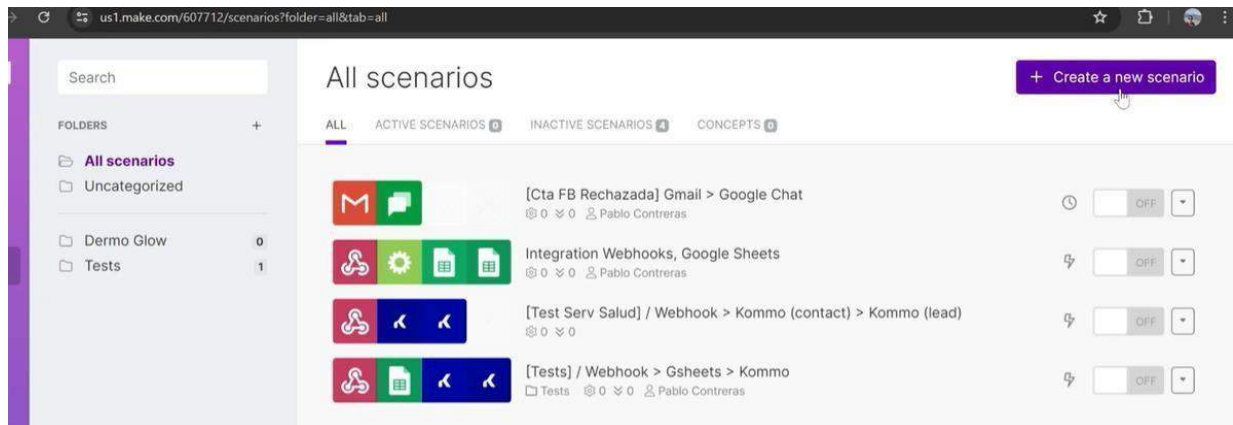
Paso 1: Ir a Scenarios y luego presionar botón Create a new scenario

Figura 15: Formulario que se crea con Make y planilla de cálculo en la que irán los datos



Fuente: elaboración propia.

Figura 16: *Scenarios* de la aplicación Make

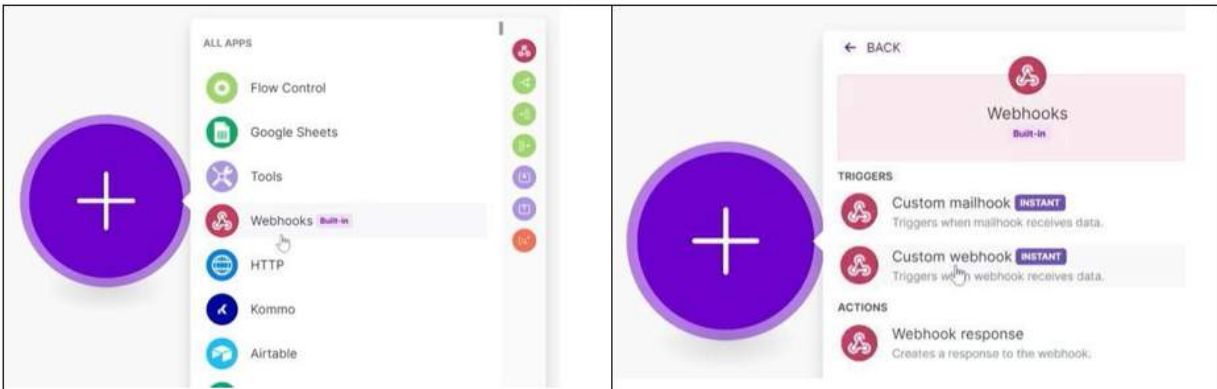


Fuente: captura de pantalla de us1.make.com.

Paso 2: Seleccionar Webhooks (imagen izquierda) y luego Custom (personalizarlo)

La función de los *webhooks* en Make es actuar como un *listener* que recibe datos de otras aplicaciones en tiempo real y activa automáticamente un escenario de automatización en Make. Al recibir un evento específico de una aplicación externa (como una nueva compra o un nuevo registro), el *webhook* en Make recibe la información instantáneamente y puede usarla para llevar a cabo acciones, como actualizar una base de datos, enviar un correo electrónico o crear un nuevo registro en otra plataforma.

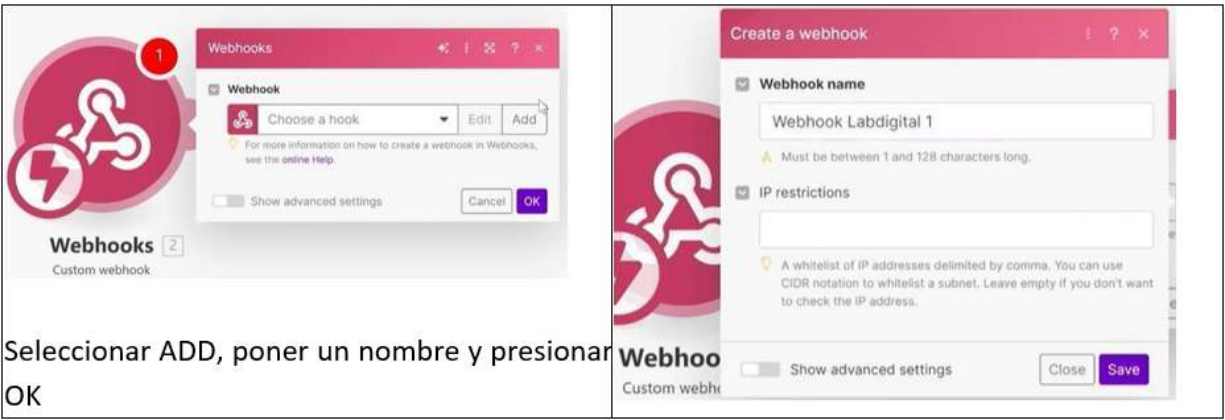
Figura 17: Selección de Webhooks y luego Custom



Fuente: captura de pantalla de us1.make.com.

Paso 3: Configurar el *webhook* poniendo un nombre (ver figura 17; el nombre seleccionado es webhooj labdigital 1, pero puede elegirse cualquier nombre).

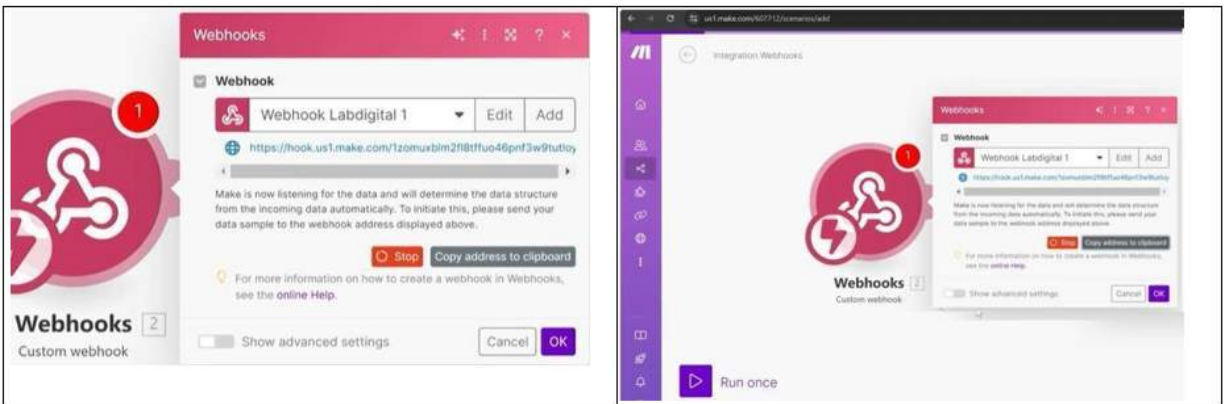
Figura 18: Configuración de Webhooks



Fuente: captura de pantalla de us1.make.com.

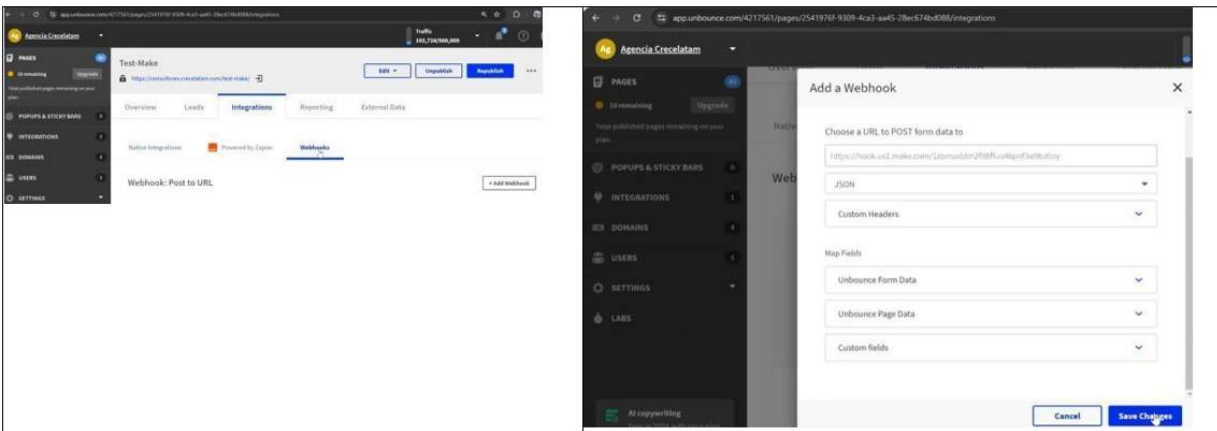
Paso 4: el tercer paso genera la dirección de URL que se inserta aquí (página de *landing page*) y se confirma presionando Ok. También debe presionarse **Run once** para que se ejecute y quede ejecutando a la espera de datos.

Figura 19: Agregar la URL. Se ejecuta con el comando Run once



Fuente: captura de pantalla de us1.make.com.

Figura 20: Se conecta al formulario

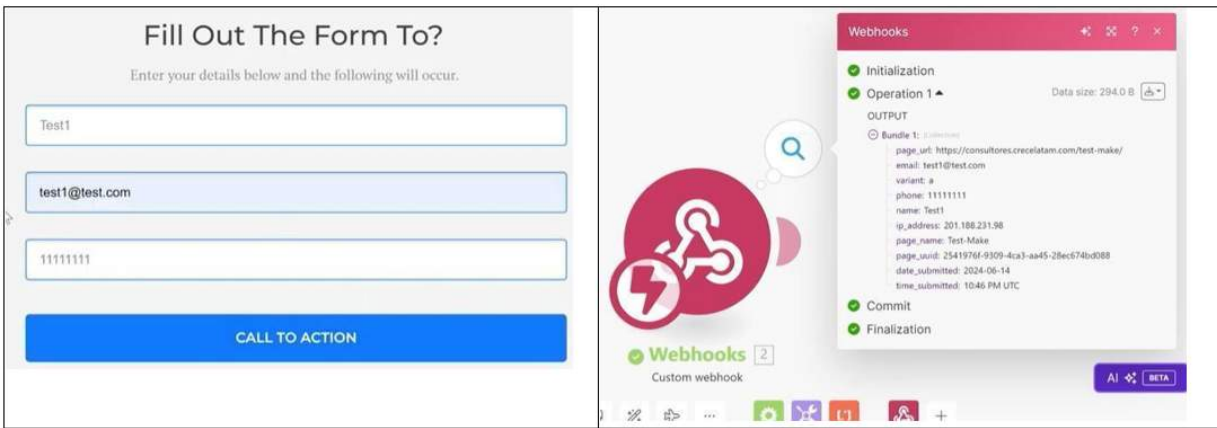


Fuente: elaboración propia.

Paso 6: desde lo que sería el formulario, relacionamos poniendo la URL copiada.

Paso 7: ahora, ingresando datos en el formulario de la figura 14, se puede apreciar que Webhook lo capturó.

Figura 21: Prueba de la captura de datos

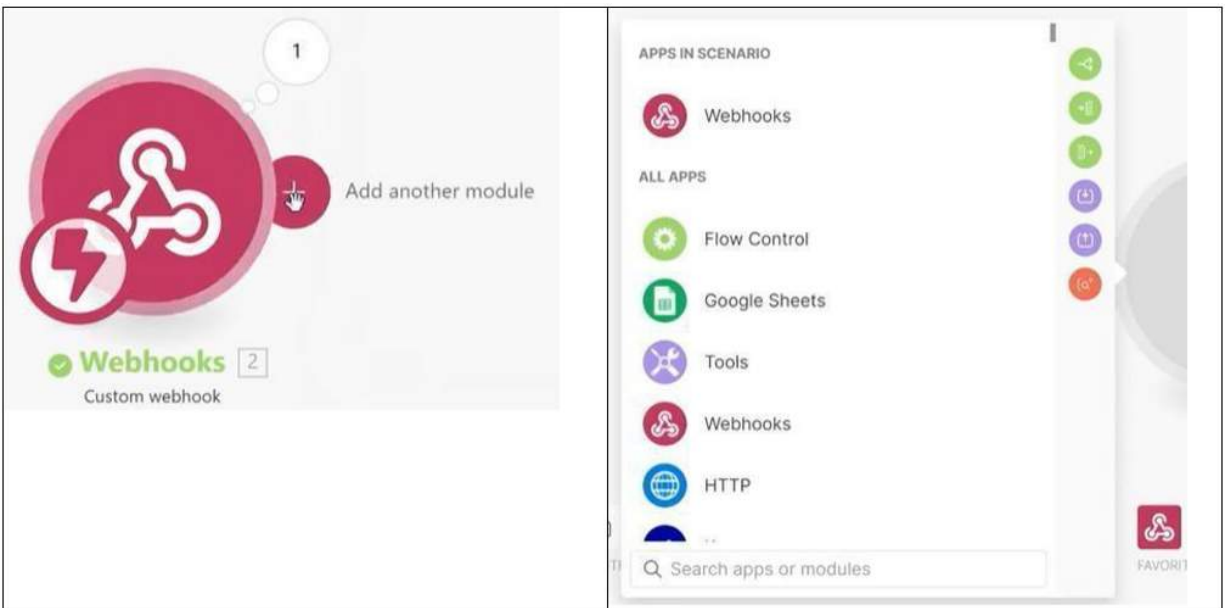


Fuente: elaboración propia.

Paso 8: Conectarlo a Google Sheets

Para que los datos ingresados desde el formulario vayan directo a la planilla de cálculo de Google Sheets, es necesario hacer la conexión con el nuevo módulo de Google Sheets.

Figura 22: Agregar Google Sheets



Fuente: elaboración propia.

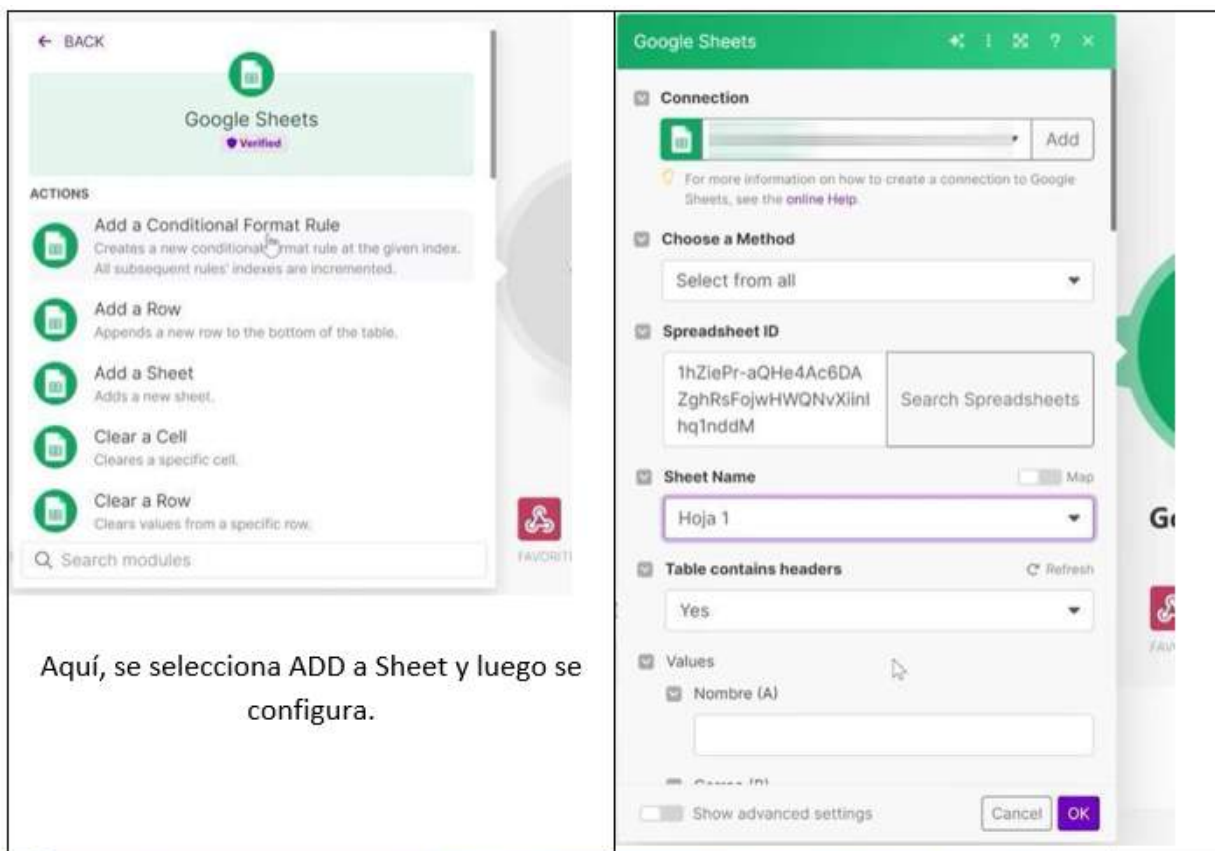
Paso 9: Configuración del punto en el que se pondrán los datos

Con la idea de que cada dato ingresado se agregue automáticamente como una nueva fila en Google Sheets cada vez que estos se confirman desde el formulario, se explica en las siguientes imágenes cómo hacerlo. Como se puede ver en la figura 22, se debe agregar el ID del formulario, nombre de columna, que, por lo general, es igual a los datos cargados en el formulario, y el nombre de la hoja a usar (por ejemplo, hoja 1, hoja 2, etcétera).

El primer dato que se configura es el correo electrónico del usuario, es decir, el formulario de Google Sheets lo agregó en

Google Drive usando su *e-mail*. Ese mismo correo electrónico es al que se hace referencia. Este dato se agrega en la primera fila que dice Connection.

Figura 23: Configuración de Sheet



Aquí, se selecciona ADD a Sheet y luego se configura.

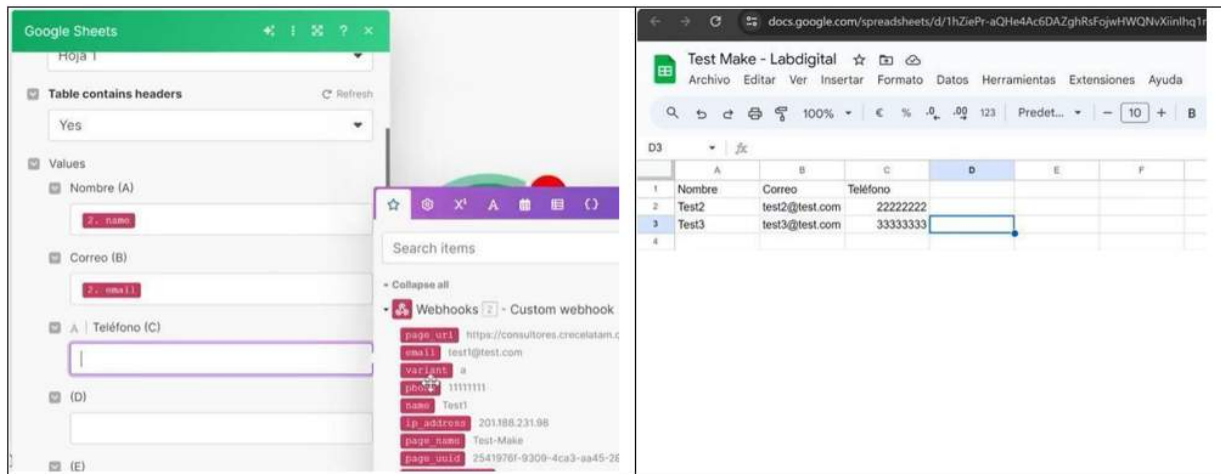
El valor que está resaltado en la UR es el que se pone en Spreadsheets ID.

The image displays the Google Sheets configuration interface. On the left, the 'ACTIONS' menu is visible, with 'Add a Sheet' selected. The right panel shows the configuration dialog for 'Add a Sheet', including fields for 'Spreadsheet ID', 'Sheet Name', and 'Table contains headers'. The 'Spreadsheet ID' field is highlighted, and the URL in the browser address bar shows the corresponding ID: `1hZiePr-aQHe4Ac6DAZghRsFojwHWQNvXiinhq1nDDM`.

Fuente: elaboración propia.

Paso 10: se asocian a cada columna de la planilla de cálculos los datos que se transferirán desde el formulario. Los datos son transferidos desde el formulario a la planilla por medio de Webhooks e impactan tal como se configuró según las columnas asociadas, línea por línea, a medida que estos van ingresando, tal como se muestra en la siguiente figura.

Figura 24: Transferencia de datos a la planilla

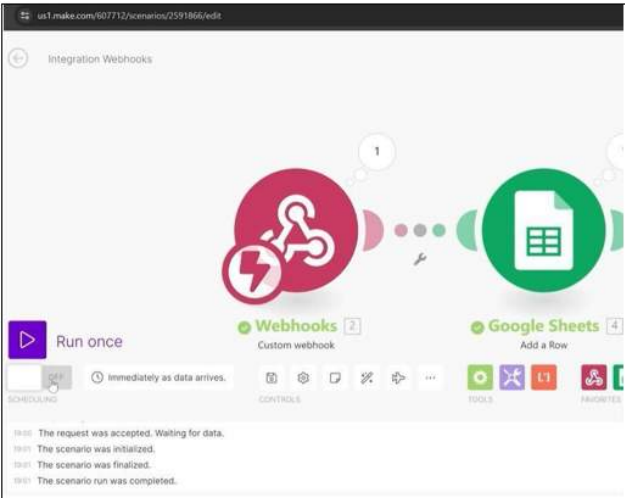


Fuente: elaboración propia.

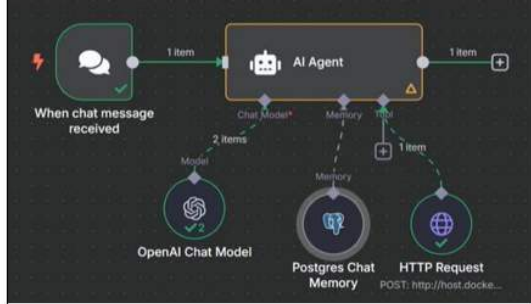
El procedimiento de trabajar con 8n8 tiene similitudes al de Make. En las diferencias, se nota el uso de nombres propios en cada herramienta, pero, conceptualmente, si se aprende a usar una, la otra es de fácil aprendizaje. En el siguiente ejemplo de 8n8, se usa **endpoint** (punto de contacto entre un cliente API y un servidor, en el que se dirigen las llamadas API). Este concepto es similar al de **webhook** usado en el ejemplo explicado de Make y, como en este caso, también debe ser configurado. Por otro lado, la posibilidad de migrar el código fuente al lenguaje Python le da a 8n8, desde el punto de vista de la programación *coding*, una gran aceptación.

Seguidamente, se muestra cómo quedan finalizados los ejemplos de Make y el 8n8 a desarrollar.

Figura 25: Modelo Make vs. modelo n8n



Modelo Make



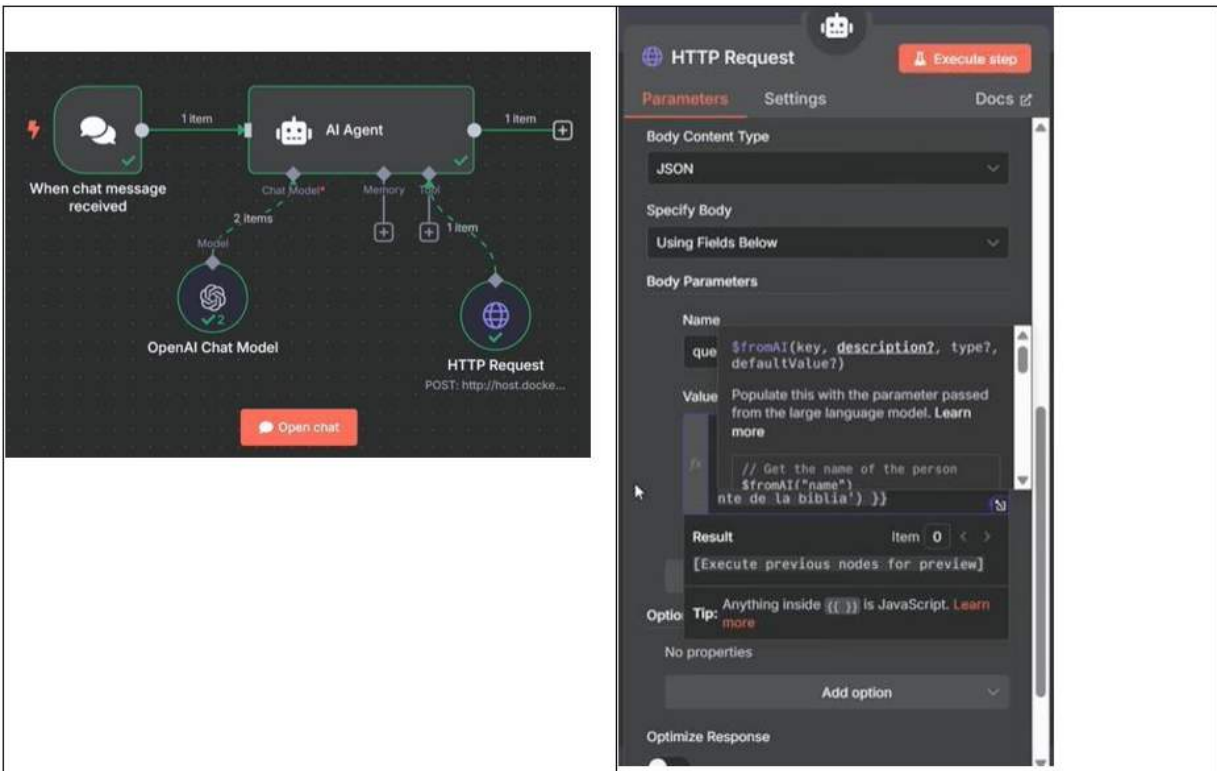
Modelo n8n

Fuente: elaboración propia.

En ambas imágenes, cada automatización debe ser configurada según sus parámetros (dependiendo de la herramienta que se use).

En la siguiente figura, se observa la configuración de algunos parámetros de HTTP Request.

Figura 26: Se configura HTTP Request



Fuente: elaboración propia.

En la siguiente figura, se puede observar n8n y el lenguaje Python en un *framework* de escritorio.

Figura 27: Lenguaje Python en Visual Studio Code trabajando con n8n

```
12 openai_api_key = os.getenv("OPENAI_API_KEY")
13
14 app = FastAPI()
15
16
17 collection_name = 'Biblia'
18 qdrant_client = QdrantClient(url="http://localhost:6333")
19
20 model_embeddings = OpenAIEmbeddings(model='text-embedding-3-small')
21
22
23 vectorstore = QdrantVectorStore(
24     client=qdrant_client,
25     collection_name=collection_name,
26     embedding_model=model_embeddings,
27     retrieval_mode=RetrievalMode.DENSE,
28     content_payload_key="text",
29 )
30
31 retriever = vectorstore.as_retriever(
32     search_type="similarity_score_threshold",
33     search_kwargs={
34         "k": 7,
35         "score_threshold": 0.5
36     }
37 )
38
39 # ===== DEFINICIÓN DEL MODELO DE ENTRADA =====
40 class QueryInput(BaseModel):
41     query: str
42
43
44
45 @app.post("/search_biblia")
46 def search_biblia(input: QueryInput):
47     try:
48         docs = retriever.get_relevant_documents(input.query)
49         results = [{"content": doc.page_content, "metadata": doc.metadata} for doc in docs]
50         return {"results": results}
51     except Exception as e:
52         raise HTTPException(status_code=500, detail=str(e))
53
54 # ===== Comando para ejecutar el servidor FastAPI =====
55 # uvicorn busqueda:app --host 0.0.0.0 --port 8000 --reload
56
57 # ===== ENDPOINT PARA VER EN NAVEGADOR CON GET =====
58
59 @app.get("/buscar")
60 def buscar(query: str):
61     docs = retriever.invoke(query)
62     return docs
63
64 # ===== Ej: http://localhost:8000/buscar?query=quien%20es%20jesus%20en%20el%20nuevo%20testamento =====
```

Fuente: elaboración propia.

Conclusión

Podemos apreciar la gran funcionalidad y el ahorro de tiempo que prometen las potentes herramientas mencionadas. Sin lugar a duda, los alumnos podrán aprovechar al máximo las bondades que ofrecen. Si bien algunos usos inmediatos en la versión gratuita se pueden aprovechar, el trabajo será mejor cuando se tenga el conocimiento de la programación adecuado, dado que se podrán incorporar estas tecnologías a los proyectos más avanzados.

Si bien las herramientas mencionadas varían en costos según sus versiones, el ahorro en tiempo de desarrollo de

software podrá, en algunos casos, justificar su uso. Los fabricantes de estas herramientas resaltan que sus *apps* (aplicaciones) preconstruidas son robustas y sencillas de integrar por medio de una interconexión con otros modelos o agentes de IA usando API.

Un alumno que se inicia en programación, seguramente, en los ejemplos mostrados, encontrará una motivación respecto al uso de las herramientas y, por lo tanto, en relación con avanzar en el aprendizaje.

CONTINUAR

Descarga en PDF



**Módulo 2. IA generativa como asistente de código
fuente de programación.pdf**

1.4 MB

