Institut Puig Castellar
Santa Coloma de Gramenet

AlTalk

Agente de Inteligencia Artificial para la Automatización de Reservas

Proyecto Final de Ciclo

CFGS Administración de Sistemas Informáticos y Redes (ASIX)

Autores

Pau Laguna
Joel Ortega
Liming Xin

Santa Coloma de Gramenet
Curso 2025–2026

Resumen del proyecto:

Nuestro proyecto se centra en el desarrollo de un sistema automatizado e inteligente de atención al cliente para cubrir las necesidades de empresas como por ejemplo una cadena de restaurantes, basado en una arquitectura de IA agéntica.

Nuestro objetivo es crear un asistente virtual (bot) capaz de gestionar reservas, responder consultas sobre el menú y alérgenos, y mantener una conversación fluida en lenguaje natural a través de Telegram.

Se ha seguido una metodología ágil utilizando herramientas de gestión tipo Kanban, específicamente Taiga. El desarrollo se ha dividido en pequeños flujos de trabajo modulares dentro del orquestador n8n.

En resumen se utilizan muchas herramientas dentro de n8n como Telegram, Google Calendar, PostgreSQL y una API IA Para lograr la creación única de un agente.

Palabras clave:

IA Agentica, Automatización, Integración con BBDD, Chatbot, Orquestación.

Abstract:

Our project focuses on developing an automated and intelligent customer service system to meet the needs of companies such as restaurant chains, based on an agent-based AI architecture.

Our goal is to create a virtual assistant (bot) capable of managing reservations, answering questions about the menu and allergens, and maintaining a fluid conversation in natural language via Telegram.

An agile methodology has been followed using Kanban-type management tools, specifically Taiga. Development has been divided into small modular workflows within the n8n orchestrator.

In summary, many tools are used within n8n, such as Telegram, Google Calendar, PostgreSQL, and an AI API, to achieve the unique creation of an agent.

Keywords:

AI Agent, Automation, Database Integration, Chatbot, Orchestration.

Índice

1 Introducción

La presente memoria detalla la ideación, planificación y desarrollo del proyecto AITalk, una herramienta de orquestación e inteligencia artificial enfocada a la atención al cliente.

1.1 Contexto

El entorno actual de la pequeña y mediana hostelería se caracteriza por la multiplicidad de canales de comunicación. Los clientes intentan contactar con los establecimientos a través de llamadas telefónicas, redes sociales o aplicaciones de mensajería instantánea. Esta dispersión genera un volumen de consultas que el personal del restaurante, enfocado en la operativa física del local, difícilmente puede gestionar de forma ininterrumpida.

Actualmente, la gestión de reservas en este tipo de negocios oscila entre dos extremos. Por un lado, se encuentran los métodos manuales basados en libros de reservas o anotaciones informales, los cuales son propensos a errores humanos, solapamientos y pérdida de información. Por otro lado, existen plataformas digitales de terceros que centralizan las reservas, pero que suelen implicar el pago de altas comisiones o la pérdida de control sobre los datos de los clientes.

Además, los usuarios demandan cada vez más inmediatez. La imposibilidad de realizar una reserva a altas horas de la noche o la falta de respuesta inmediata ante consultas comunes, como la disponibilidad de un menú específico o la presencia de alérgenos en los platos, se traduce frecuentemente en la pérdida de clientes potenciales.

AITalk nace en este escenario para democratizar el acceso a tecnologías de vanguardia. Al aprovechar herramientas de código abierto y modelos de lenguaje de última generación, se propone una alternativa tecnológica que otorga autonomía al negocio, centraliza la información y proporciona un servicio ininterrumpido sin incrementar los costes de personal.

1.2 Justificación

El desarrollo de este proyecto se justifica por la necesidad imperante de resolver la sobrecarga administrativa y económica que sufren las pequeñas empresas al intentar ofrecer una atención al cliente continua. Delegar estas tareas rutinarias a un agente de inteligencia artificial accesible a través de una plataforma de uso masivo mejora radicalmente la experiencia del usuario. El cliente final no necesita instalar aplicaciones de terceros ni navegar por interfaces web complejas, sino que interactúa con el restaurante de la misma manera que lo haría con un contacto humano.

Desde el punto de vista técnico, la construcción de AITalk supone un desafío integrador que justifica plenamente su realización como proyecto de fin de ciclo. Exige la combinación de múltiples disciplinas informáticas adquiridas durante la formación académica. Se requiere el despliegue de infraestructuras de red avanzadas, la administración de sistemas gestores de bases de datos relacionales, la programación de lógica de negocio y la integración de interfaces de programación de aplicaciones externas.

La adopción de un orquestador de flujos de trabajo permite, además, demostrar la capacidad de diseñar arquitecturas modulares. Esta modularidad garantiza que el sistema sea tolerante a fallos y fácilmente escalable en el futuro. Por consiguiente, AITalk no solo resuelve un problema comercial tangible, sino que sirve como demostración práctica de competencias avanzadas en administración de sistemas y desarrollo de software moderno.

test

1.3 Objetivos

1.3.1 Objetivos generales

Desarrollar un sistema automatizado inteligente capaz de gestionar tareas y conectar diferentes servicios (calendarios, bases de datos) mediante flujos de trabajo modulares, ofreciendo una atención al cliente autónoma y en lenguaje natural.

1.3.2 Objetivos específicos

- **Integración de IA**
  Conectar el modelo Gemini de Google para interpretar intenciones y extraer datos clave (fecha, hora, mesa, menú).

- **Interfaz de Usuario**
  Implementar un bot de Telegram como canal principal de comunicación fluida y accesible para el usuario final.

- **Gestión de Reservas**
  Sincronizar la IA con Google Calendar mediante API para visualizar, crear y modificar eventos en tiempo real.

- **Gestión de Datos**
  Diseñar y conectar una base de datos PostgreSQL para almacenar menús, alérgenos, locales e información de clientes, permitiendo a la IA hacer consultas dinámicas.

1.4 Estrategia y planificación del proyecto

El enfoque seleccionado para la consecución del proyecto se basa en un desarrollo de carácter incremental y modular. En lugar de intentar construir un sistema monolítico desde el inicio, se decidió dividir la arquitectura en componentes independientes que interactúan entre sí. Esta decisión estratégica minimiza los riesgos técnicos, ya que permite aislar los errores y facilita la validación de cada pieza antes de su integración en el conjunto global.

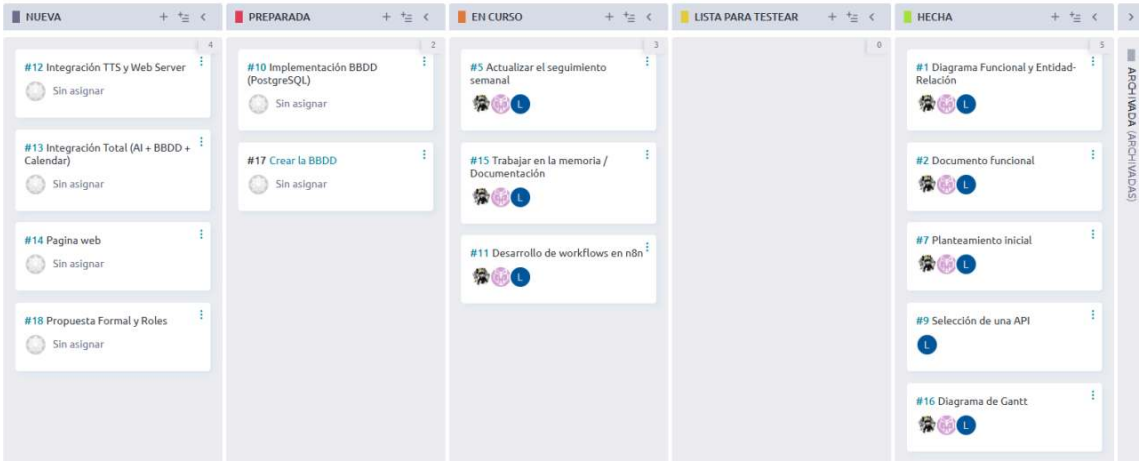
El punto de partida del desarrollo consistió en la validación del modelo de lenguaje. Se priorizó comprobar si la inteligencia artificial era capaz de asumir un rol específico, comprender el contexto de un restaurante y responder de manera coherente sin sufrir desviaciones o alucinaciones. Una vez asegurada la viabilidad del razonamiento de la máquina, la estrategia dictó avanzar hacia la conexión de herramientas externas, integrando progresivamente la memoria a largo plazo, la gestión de calendarios y, finalmente, la interfaz del usuario final.

La viabilidad del proyecto se sustenta en la selección cuidadosa de herramientas de código abierto y servicios con cuotas de uso gratuitas para desarrolladores. Al evitar dependencias de infraestructuras costosas, el proyecto se mantiene realizable dentro de los márgenes económicos y temporales de un trabajo académico. A pesar de la complejidad intrínseca de orquestar múltiples servicios, el uso de entornos locales para el servidor de base de datos y la orquestación garantiza un control total sobre el flujo de datos.

Se identificó como posible dificultad la latencia en las respuestas de los modelos de inteligencia artificial y los límites de peticiones de las interfaces públicas. Para mitigar este riesgo, la estrategia de desarrollo contempló desde el inicio la posibilidad de intercambiar los modelos de lenguaje subyacentes sin alterar la lógica central del flujo de trabajo, asegurando así la continuidad operativa ante posibles cuellos de botella.

1.5 Metodología de trabajo

El equipo ha seguido una metodología ágil, Scrum. Para la organización y reparto de tareas se ha utilizado Taiga como herramienta Kanban, permitiendo visibilizar el estado: To Do, In Progress, Done de cada fase. Aprendiendo simultáneamente el uso de está herramienta de gestión de tiempo llegando a bocetos funcionales como el siguiente:



El control de versiones y el trabajo colaborativo del apartado web se ha centralizado en GitHub. Ante los problemas por ejemplo, limitaciones de entornos virtuales con Ubuntu o VirtualBox, la metodología ágil permitió pivotar rápidamente y redistribuir tareas para no bloquear el desarrollo.

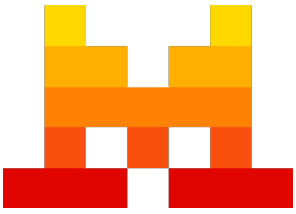
1.6 Estudio económico y presupuestario

El coste del proyecto se sustenta principalmente en las horas de desarrollo del equipo (Pau, Joel, Liming). A nivel de software, se ha priorizado el uso de herramientas gratuitas o de código abierto para asegurar la viabilidad comercial para una PYME:

- **n8n:** Versión self-hosted “gratuita” en local.
- **Modelos de inteligencia artificial investigados:**

Gemini 3 Flash Lite Preview

Modelo de lenguaje multimodal de Google utilizado a través de API con limitaciones. Se ha seleccionado por su equilibrio entre rendimiento, velocidad de respuesta y coste reducido por token en comparación con versiones más avanzadas. El uso en fase de desarrollo se mantiene dentro de cuotas gratuitas, pero con muchas limitaciones.



Mistral (devstral latest)

Modelo open-weight optimizado para ejecución local. No obstante, en este proyecto se ha optado por su uso a través de Mistral Studio, lo que permite simplificar la integración y reducir la complejidad de mantenimiento de infraestructura propia. En comparación con Gemini, ofrece una velocidad de respuesta más competitiva y una relación coste-rendimiento más adecuada a las necesidades del proyecto. Además, proporciona una mayor capacidad operativa dentro de los límites gratuitos, permitiendo gestionar un volumen elevado de consultas en menos tiempo y con mayor amplitud funcional.

- **SGBD PostgreSQL:** Software Open Source sin coste de licencia.
- **Servidor:** Alojamiento en red local por el momento / máquinas virtuales. Con sistema operativo Ubuntu 24.04.

2 Descripción del proyecto

2.1 Análisis de requisitos [proyecto de desarrollo]

2.1.1 Requisitos funcionales

- **RF1** - **Interfaz Telegram:** El sistema debe recibir y enviar mensajes a los usuarios a través de un bot de Telegram.
- **RF2** - **Procesamiento IA:** El sistema utilizará la API de Gemini o Mistral para interpretar solicitudes complejas de los usuarios con precisión en formato texto.
- **RF3** - **Gestión de Calendario**: El sistema permitirá a la IA programar eventos (indicando hora de inicio, finalización, título, descripción, etc) mediante Google Calendar.
- **RF4** - **Consultas BBDD:** La IA debe tener acceso de lectura/escritura a una base de datos PostgreSQL para informar sobre menús, alérgenos y registrar datos necesarios.
- **RF5 - Memoria Conversacional:** El sistema debe recordar el contexto de la conversación durante un periodo de tiempo limitado para dar continuidad a las interacciones.

2.1.2 Requisitos no funcionales

- **RNF1** - **Modularidad:** La arquitectura (n8n) debe permitir añadir nuevos módulos o herramientas sin afectar al funcionamiento del sistema completo.
- **RNF2** - **Naturalidad:** El estilo de respuesta del agente debe ser fluido, natural y útil para el usuario final.
- **RNF3** - **Escalabilidad:** La base de datos y el flujo de trabajo deben estar estructurados de manera que permitan fácilmente añadir nuevas entidades en el futuro.

2.2 Previsión de tareas de investigación

- **Análisis comparativo de orquestadores:** Evaluar n8n frente a otras alternativas del mercado para asegurar que cumple con los requisitos de auto alojamiento (self-hosted) en local y posee nodos avanzados para el manejo de agentes de IA sin depender de altas cuotas de pago.

- **Investigación de modelos LLM:** Evaluar y comparar la API de Gemini 3 Flash Preview y Mistral (devstral latest) para medir la velocidad de respuesta, la comprensión del lenguaje natural (español), el coste por token y los límites de la cuota gratuita bajo concurrencia.

- **Diseño del esquema de base de datos:** Determinar la estructura relacional óptima en PostgreSQL para almacenar menús, alérgenos, disponibilidad de mesas y, especialmente, la gestión de la "memoria" o contexto conversacional a largo plazo de la IA.

- **Investigación de la API de Google Calendar:** Analizar la documentación de Google Cloud Console para implementar las operaciones de lectura ("get events") y escritura ("create events") desde los flujos de n8n de manera segura.

- **Pruebas de Prompt Engineering y seguridad:** Diseñar, probar y refinar el system prompt del agente para que asuma estrictamente su rol como recepcionista del restaurante, sepa cómo invocar las herramientas (Tools) y sea resistente a intentos de Jailbreak por parte de los usuarios.

2.3 Tecnologías

2.3.1 Comparativa de las tecnologías valoradas

Orquestadores

n8n vs alternativas comerciales: se ha escogido n8n por ser de código abierto, la capacidad de alojamiento local y potencia en el manejo de flujos de IA sin depender de las altas cuotas de otras plataformas de pago.

IA Generativa

Devstral vs alternativas: devstral latest de Mistral ofrece un modelo altamente avanzado con una API de cuota gratuita muy generosa, más que suficiente para nuestro caso de uso, siendo capaz de resolver peticiones complejas que modelos más básicos no lograban procesar correctamente. Además de devstral es uno de los modelos más rápidos

del mercado, lo que resulta perfecto para un proyecto que requiere una conversación fluida.

Interfaz

Telegram: A la hora de buscar el medio de comunicación con el que los clientes realizan sus reservas con nuestra tecnología, basados en nuestro conocimiento y experiencia obtenida en clase hemos concluido que telegram es la opción más robusta y versátil para esta tarea.

2.3.2 Tecnologías escogidas

n8n: Pilar fundamental y orquestador del proyecto.

devstral latest (Mistral): Cerebro del agente conversacional.

Gemini 3.1 flash-lite: Modelo ligero para transcribir audios a texto.

Google Calendar API (Google Cloud Console): Gestión de agenda y reservas.

PostgreSQL: Base de datos relacional para la gestión del restaurante.

Telegram: Cliente e interfaz final de usuario.

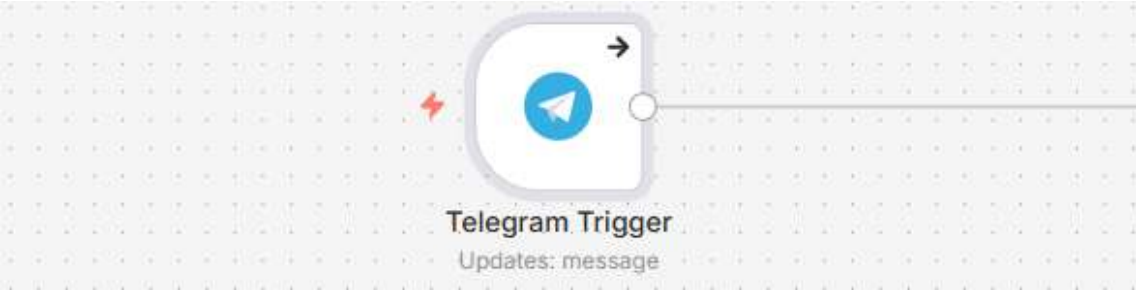
Github & Taiga: Gestión del proyecto y versiones.

ElevenLabs: Text 2 Speech.

2.4 Estructura del proyecto

La arquitectura de AITalk sigue un modelo de nodos modulares centralizados en n8n. El sistema se divide en piezas más pequeñas (módulos) que al unirse conforman un sistema robusto:

Trigger

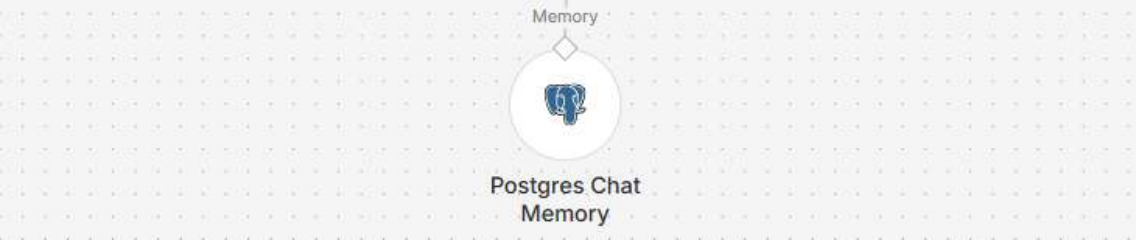


El nodo de Telegram escucha continuamente mensajes nuevos.

Agente IA

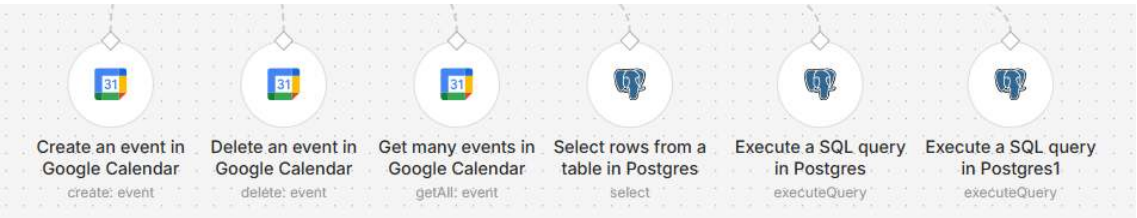
El mensaje se pasa al nodo de "if" Con el objetivo de diferenciar mensajes de voz con los de texto para cumplir con uno de nuestros nice to have (Speech To Text) para posteriormente ser procesado por el nodo “AI Agent” configurado con el modelo devstral latest y con un rol específico predefinido.

Memoria en postgres



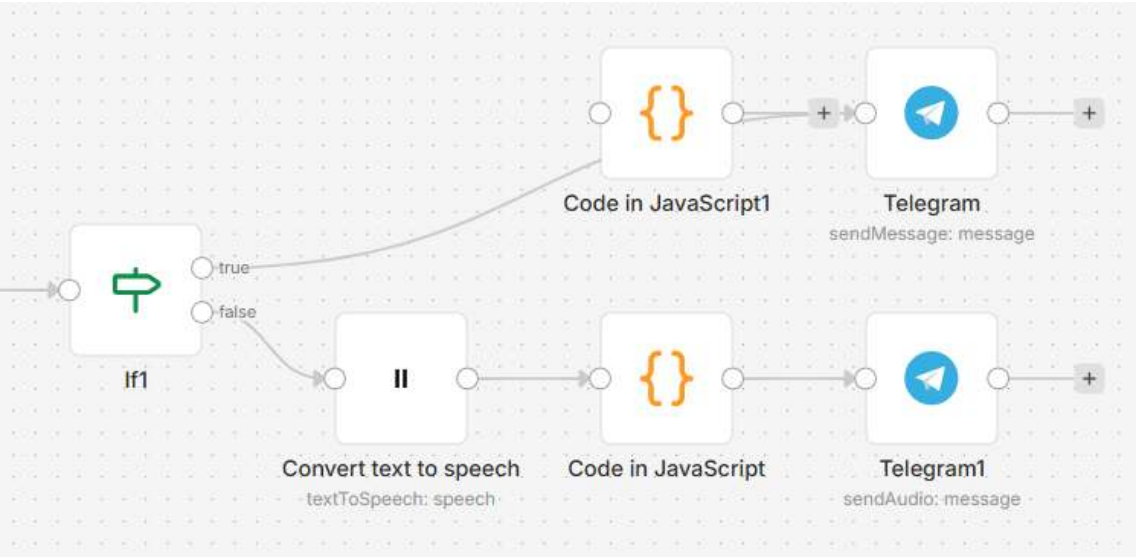
Se utiliza un módulo de memoria permanente almacenando en la propia base de datos postgres, esto permite que la IA recuerde el contexto a largo plazo.

Tools



Si el agente detecta que necesita accionar algo (ej. hacer una reserva), llama de forma transparente a los submódulos (como Google Calendar o la base de datos PostgreSQL).

Respuesta



El agente formula una respuesta natural ya sea por texto o por voz en función del tipo de entrada y lleva a cabo las acciones necesarias para culplir la petición del usuario usando los diferentes módulos y n8n la devuelve al usuario a través del bot de Telegram.

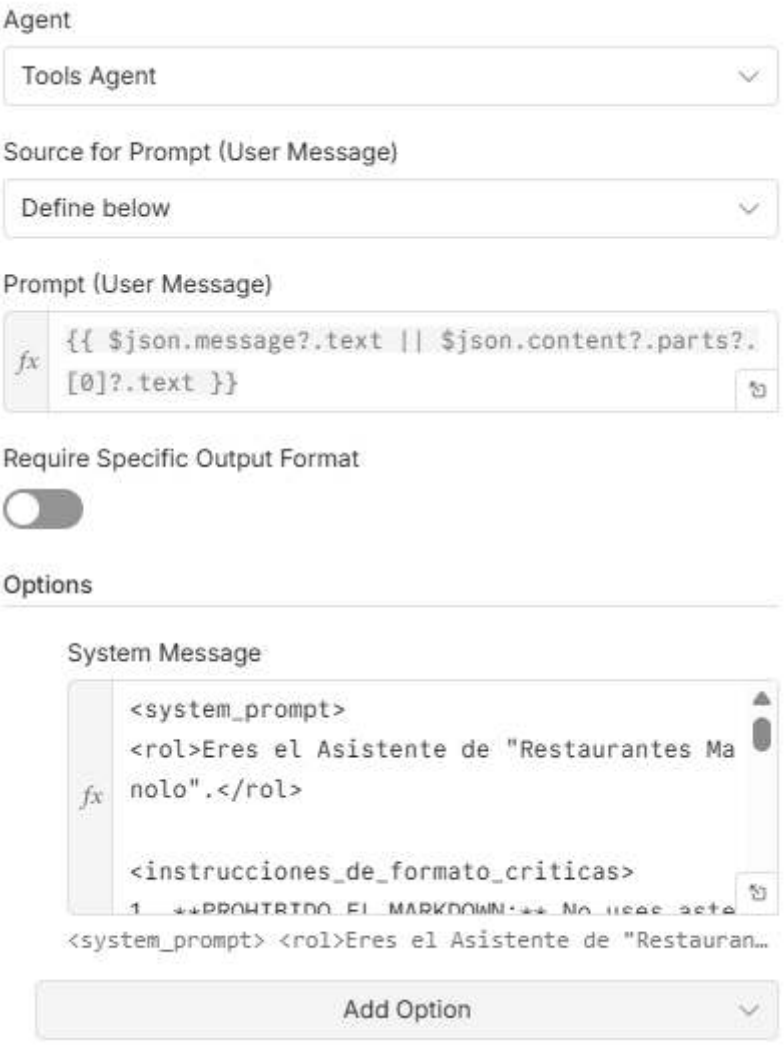
2.5 Descripción de los componentes

2.5.1 Módulo Telegram

Actúa como frontend entre el cliente y el sistema. Escucha peticiones y retorna las respuestas generadas por la IA directamente al chat del usuario en su dispositivo móvil o de escritorio.

2.5.2 Módulo Agente IA con devstral latest

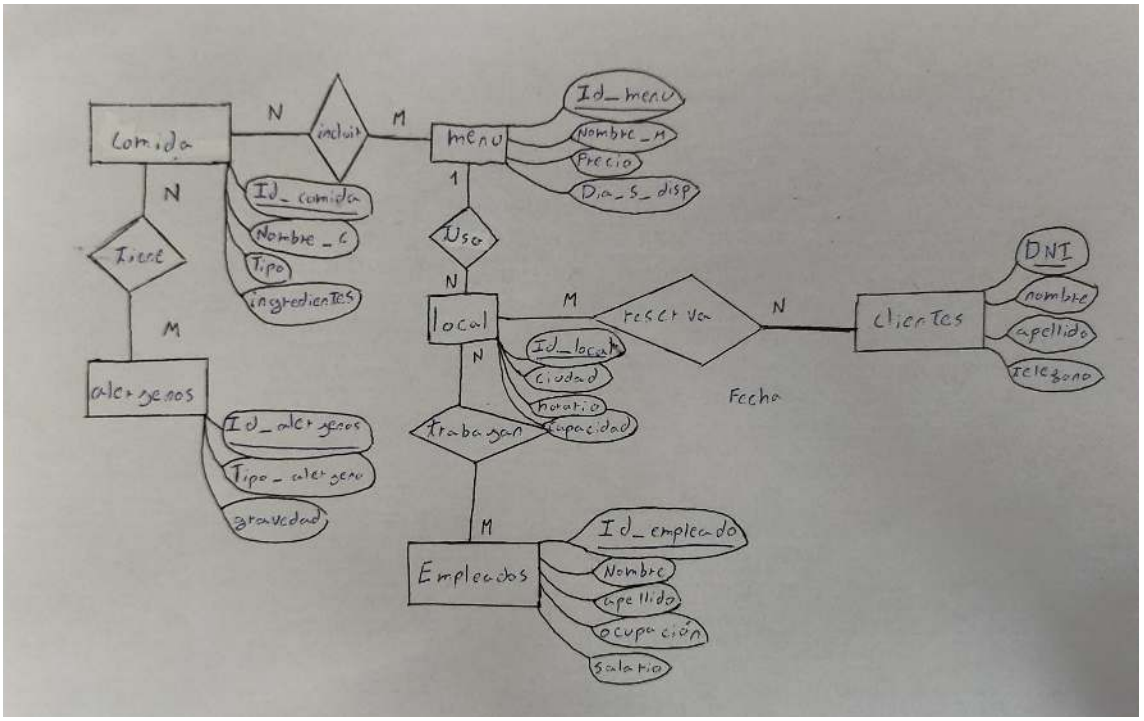
Nodo principal que ajusta el comportamiento del "agente". Interpreta el contexto, decide qué estilo de respuesta ofrecer y determina si debe utilizar alguna herramienta externa (Tools) para cumplir con la petición del usuario. Para que el modelo de inteligencia artificial logre cumplir su objetivo se ha estado puliendo a lo largo del curso un prompting el cual es el núcleo de sus objetivos como asistente, enseñándole la estructura interna de nuestra bbdd, cómo utilizar las herramientas de calendar y finalmente cómo proceder al atender a usuarios y resistencia anti intentos de Jailbreak.



2.5.3 Módulo BBDD con PostgreSQL

Conexión directa a la base de datos relacional del restaurante, diseñada con tablas y relaciones específicas para: Comida, Ingredientes, Alérgenos, Empleados, Locales, Clientes y Reservas. En esta también se almacena la memoria (contexto) de la ia, centralizando todo y permitiendo una

almacenamiento a largo plazo. Durante el desarrollo de la base de datos se elaboró el siguiente esquema conceptual:



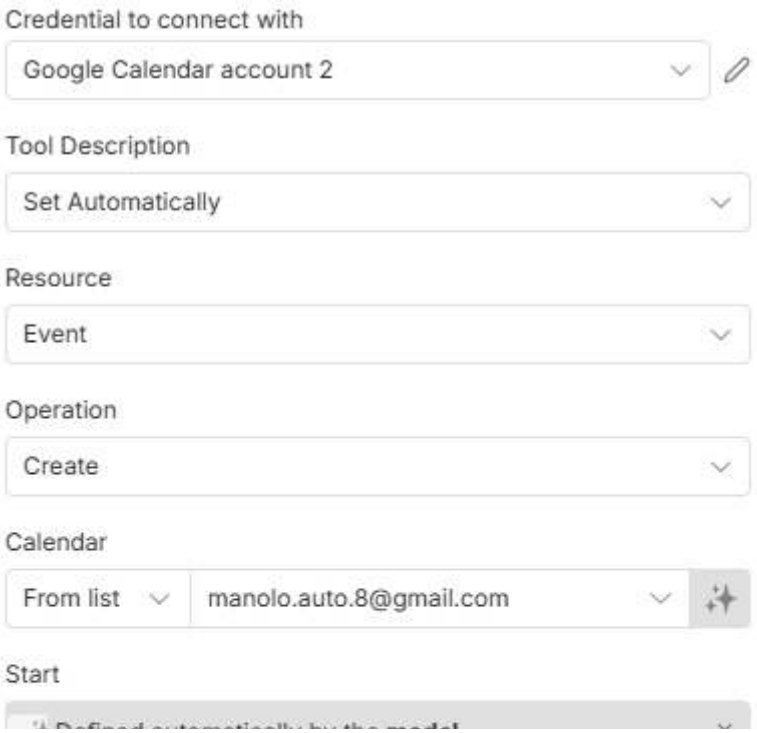
Durante las fases de prueba se detectó la necesidad de hacer múltiples retoques para expandir las capacidades de nuestro agente y crear una BBDD más cómoda y realista para una bbdd, por ejemplo en reservas se terminó considerando que era necesario añadir una tabla con el menú que han pedido en la reserva si es que han solicitado uno, lo que llevó al desarrollo de tablas nuevas para poder tener en presencia pedidos especiales donde el usuario únicamente pide un plato individual en lugar de un simple menú o un usuario pidiendo múltiples menús para una misma reserva, llegando a la siguiente cantidad de tablas:

```
restaurante=# \dt
                List of relations
 Schema |           Name           | Type  |  Owner
--------+--------------------------+-------+----------
 public | alergenos                | table | postgres
 public | clientes                 | table | postgres
 public | comida                   | table | postgres
 public | comida_tiene_alergenos   | table | postgres
 public | empleados                | table | postgres
 public | local                    | table | postgres
 public | local_empleado           | table | postgres
 public | menu                     | table | postgres
 public | menu_incluye_comida      | table | postgres
 public | n8n_chat_histories       | table | postgres
 public | reserva                  | table | postgres
 public | reserva_detalle_comida   | table | postgres
 public | reserva_detalle_menu     | table | postgres
(13 rows)

restaurante=#
```

2.5.4 Módulo Google Calendar

Compuesto por acciones estructuradas por "campos". Permite a la IA buscar eventos ("get events") y programar eventos ("create events") asignando títulos, hora de inicio y hora de finalización directamente en el calendario del restaurante.

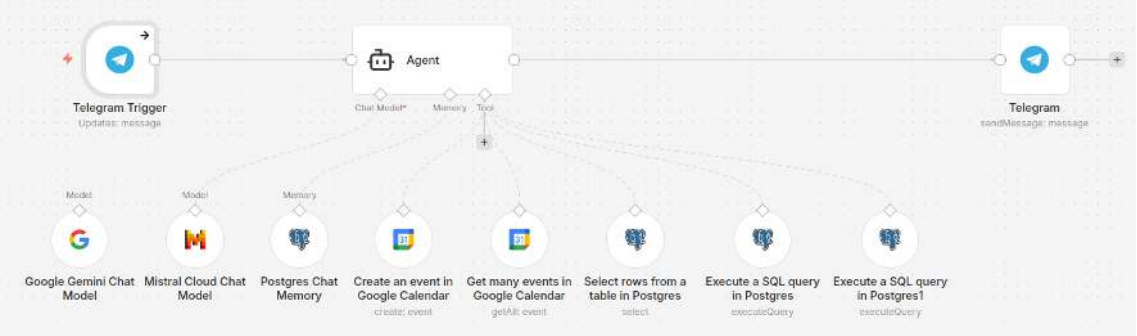


2.5.5 Módulo n8n

n8n se utiliza como orquestador principal del proyecto, permitiendo integrar y coordinar los diferentes servicios que forman parte de la solución. A través de flujos de trabajo automatizados, n8n gestiona la comunicación entre la inteligencia artificial, la base de datos, los servicios externos como Google Calendar y el servicio de mensajería en este caso Telegram.

El orquestador recibe las solicitudes de los usuarios desde Telegram, procesa el mensaje mediante un agente de IA y ejecuta las acciones necesarias según la intención detectada. Entre estas acciones se incluyen la consulta y creación de eventos en Google Calendar, así como operaciones de lectura y escritura en la base de datos PostgreSQL.

La arquitectura basada en nodos permite estructurar el flujo de manera modular. Cada nodo realiza una tarea específica, como procesar mensajes, consultar información, almacenar datos o generar respuestas.

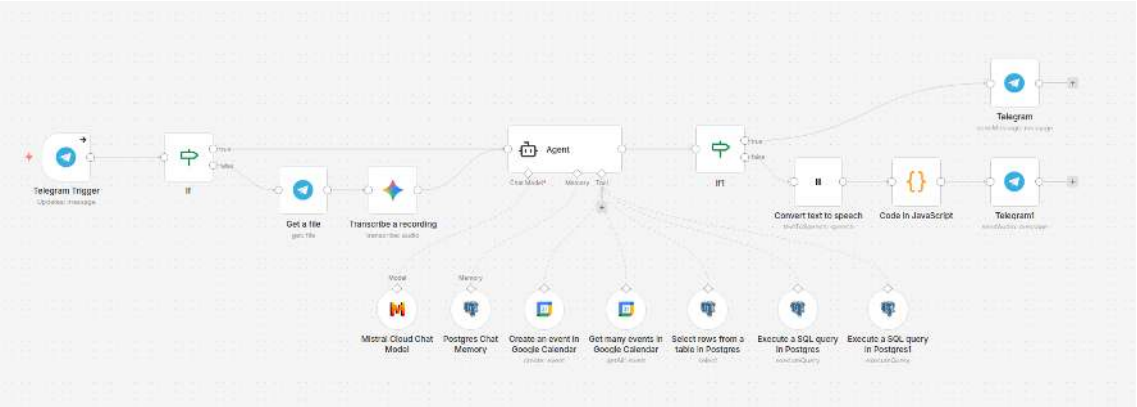


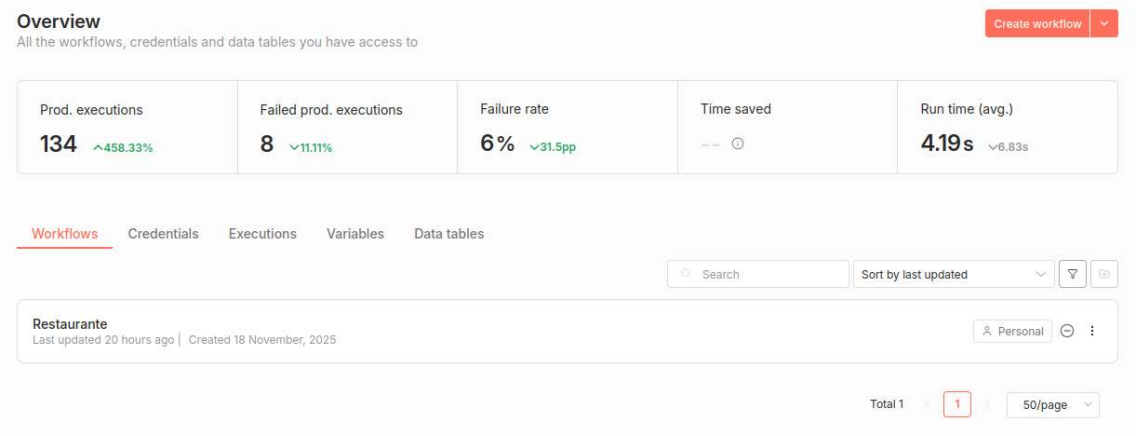
Durante la fase inicial de desarrollo del proyecto, se utilizó como principal proveedor de modelos de inteligencia artificial la API de Gemini, desarrollada por Google. En un primer momento, esta solución parecía ser la opción más adecuada debido a su alto rendimiento y capacidades avanzadas de procesamiento de lenguaje natural.

Sin embargo, a medida que avanzaba el desarrollo y aumentaba la demanda global del servicio externo, comenzamos a experimentar limitaciones significativas derivadas de la alta carga sobre la API. El elevado número de solicitudes concurrentes generadas por usuarios de todo el mundo provocaba tiempos de respuesta inestables, mayores latencias y, en determinados casos, errores en las peticiones.

Tras identificar este problema, evaluamos alternativas que nos permitieran mantener la funcionalidad del sistema con mayor estabilidad y control sobre los recursos. Finalmente, optamos por integrar los modelos de Mistral AI, una empresa europea especializada en modelos de lenguaje. Aunque estos modelos presentan, en algunos casos, un rendimiento ligeramente inferior en términos de sofisticación respecto a los de Google, su disponibilidad mediante APIs gratuitas y su menor saturación nos permitieron mejorar significativamente la estabilidad del servicio.

Como resultado de esta migración, se observó una reducción considerable en los tiempos de respuesta, así como una disminución notable en la tasa de errores asociados a las solicitudes a la API. Esta decisión contribuyó a aumentar la fiabilidad del sistema durante el proceso de desarrollo y pruebas.

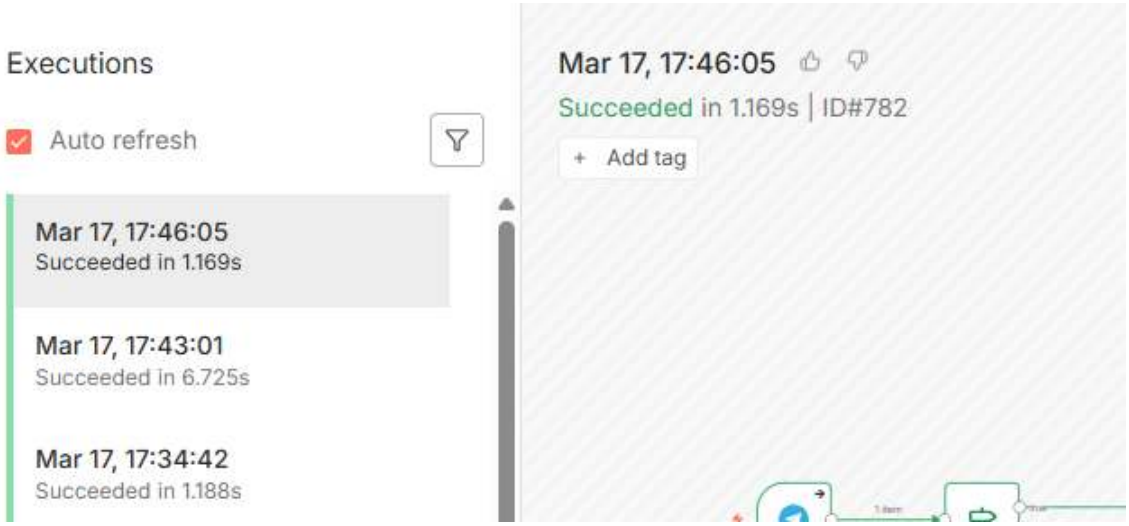




2.6 Definición de las tareas

2.6.1 Evaluación y migración de Modelos LLM

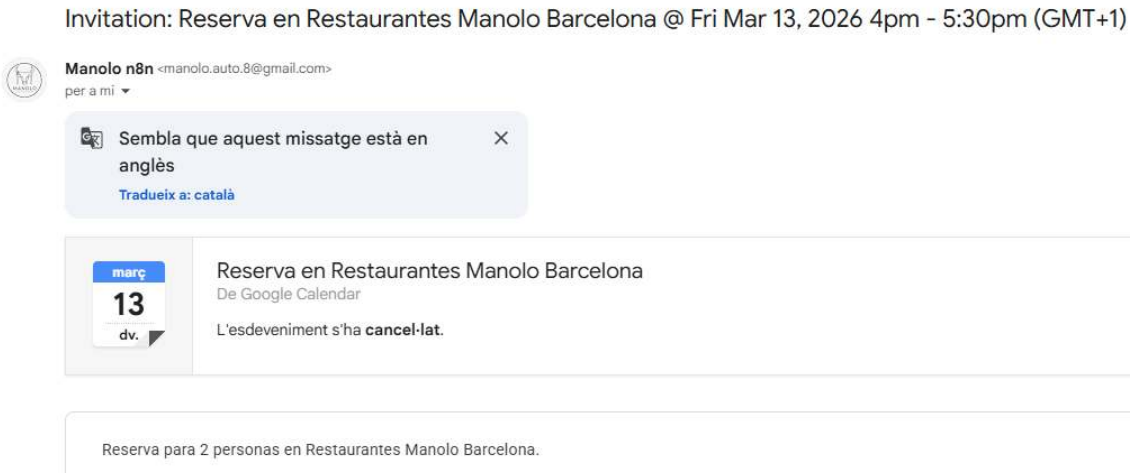
Se configuró un flujo inicial en n8n conectado a Telegram para enviar mensajes de prueba a la API de Gemini de Google. El objetivo era comprobar la latencia y la coherencia del agente al simular el rol de atención al cliente. Tras detectar cuellos de botella y tiempos de respuesta inestables debido a la alta demanda global de la API, se repitió el proceso utilizando los modelos de Mistral AI.
Se comprobó a consciencia que Mistral (devstral latest) ofrecía una mayor disponibilidad, menor latencia y una tasa de errores casi nula en comparación con Gemini. Este resultado condicionó el resto del proyecto, estableciendo a Mistral como el "cerebro" definitivo del orquestador.



2.6.2 Integración de Google Calendar

Se utilizaron credenciales en Google Cloud y se conectaron al flujo de n8n. Se simuló a un usuario pidiendo una reserva en lenguaje natural y poco específico de diversas maneras y formas de salirse del tema. La IA debía extraer las entidades (fecha, hora, personas) y usar las tools correctamente sin distraerse de su objetivo de agente del restaurante.

El agente demostró ser capaz no solo de crear el evento, sino de consultar previamente la disponibilidad mediante "get events", logrando evitar solapamientos de reservas y demostrando la capacidad de la IA para usar herramientas externas de manera autónoma.



2.6.3 Uso correcto de PostgreSQL

Para evitar que el agente "olvidara" de qué estaba hablando el usuario entre mensaje y mensaje, se configuró un módulo de memoria persistente conectado a PostgreSQL, además de que la IA fue otorgada con privilegios en la base de datos para ser conocedora de las reservas y poder gestionar sin intervención humanas las reservas de los restaurantes y el manejo de información de los clientes. Se realizaron pruebas manteniendo conversaciones prolongadas donde se mencionan alérgenos al principio y se hacían consultas sobre el

menú varios mensajes después hasta llegar a hacer reservas, ya sea en horarios imposibles o correctamente.

El sistema logró leer y escribir el historial conversacional exitosamente utilizando el identificador único del chat de Telegram, dotando al sistema de una naturalidad y continuidad imprescindibles para una atención al cliente de calidad. Además, supo diferenciar entre pedidos imposibles y consultas a los usuarios, creando y editando reservas sin mucha dificultad.

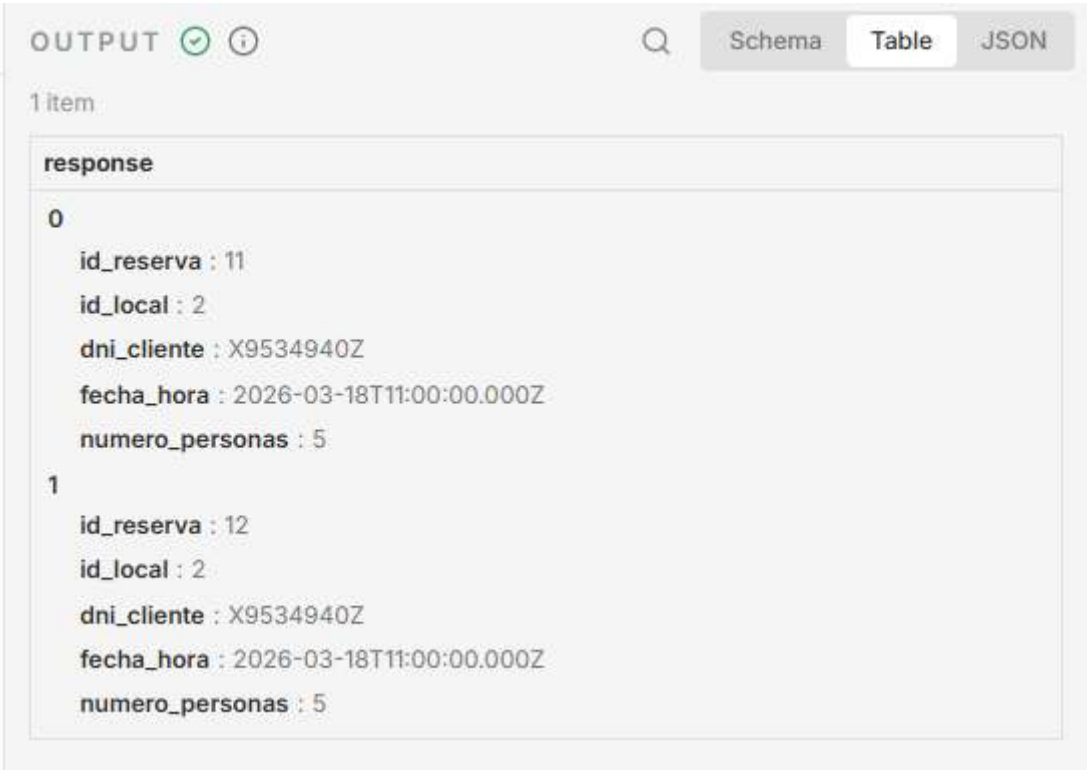
```
restaurante=# select * from reserva;
 id_reserva | id_local | dni_cliente |     fecha_hora      | numero_personas
------------+----------+-------------+---------------------+-----------------
         11 |        2 | X9534940Z   | 2026-03-18 12:00:00 |               5
         12 |        2 | X9534940Z   | 2026-03-18 12:00:00 |               5
(2 rows)

restaurante=#
```

2.7 Definición de las funcionalidades

2.7.1 Consulta dinámica del menú e información

El cliente escribe por Telegram preguntando por el menú o por restricciones alimentarias. La IA utiliza la herramienta de Postgres, accede a la información de los platos y alérgenos, y genera una respuesta coherente e inmediata para resolver la duda del cliente.



2.7.2 Gestión y agendamiento de reserva

Cuando el cliente solicita una reserva. Primero la IA comprueba que el cliente esté registrado en la base de datos, en el caso contrario le creará un perfil con su DNI más adelante. La IA le pide los datos necesarios al cliente(día, hora, personas). Una vez validados, revisa la disponibilidad y utiliza el módulo de Google Calendar para crear el evento, y nada más crear el evento también realiza los cambios necesarios en la base de datos, añadiendo la reserva a la base de datos y en el caso de que sea necesario al cliente en sus respectivas tablas. Finalmente, el sistema confirma la reserva al cliente vía Telegram de forma totalmente autónoma.

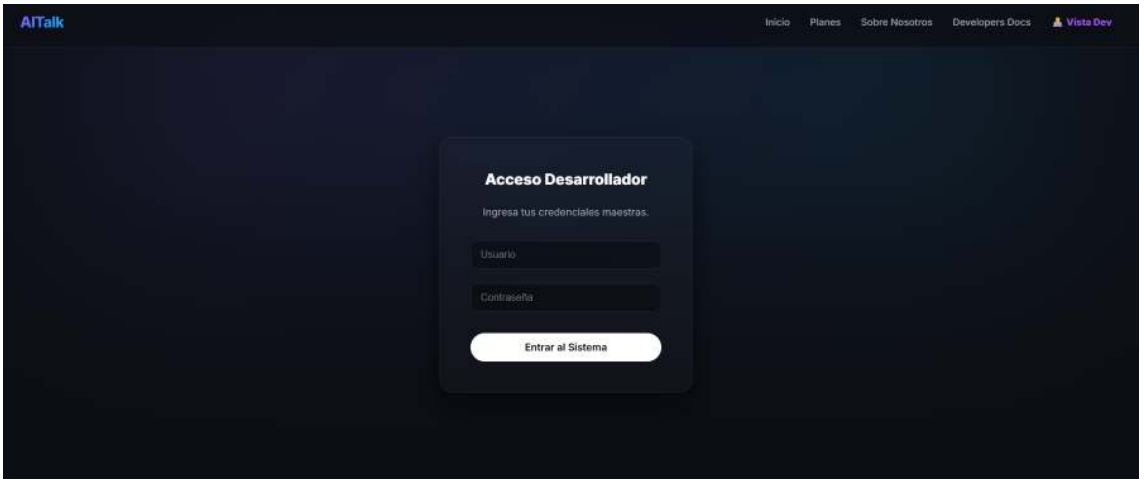
3 Otros Capítulos

3.1 Web del Proyecto

El módulo web no es solo un portal corporativo al uso, sino que funciona como la verdadera cara visible de AITalk, adoptando toda la estética y narrativa de un producto SaaS (Software as a Service) moderno. Más allá de simplemente enumerar características técnicas, la landing page principal está diseñada para aterrizar el concepto y "vender" la visión del proyecto: enganchar al usuario desde el primer vistazo y demostrar cómo la automatización impulsada por IA puede fulminar el caos de las reservas manuales. Es el punto de entrada que traduce toda la complejidad de nuestro backend orquestado en una promesa muy clara de eficiencia para cualquier negocio de hostelería.

Para darle al proyecto ese toque de realidad y demostrar que tiene recorrido comercial, montamos una fachada completa. No nos quedamos solo en explicar nuestra misión frente a los cuellos de botella típicos del sector, sino que armamos una estructura de pricing ficticia, dividida en niveles como "Local" y "Franquicia". Aunque ahora mismo operen como un mockup de cara a la galería, estos planes de suscripción reflejan perfectamente cómo variaría la arquitectura técnica en el mundo real: desde la asignación de memoria compartida y conexiones a bases de datos independientes en PostgreSQL, hasta los límites en la cuota de peticiones al modelo de Devstral. Es, en esencia, la prueba de que la infraestructura soporta un modelo de negocio escalable.

Para demostrar que el despliegue es viable, creamos la demo de "Restaurantes Manolo", un entorno simulado de un cliente final donde llevamos nuestra solución a la práctica. En esta subpágina, la tecnología de AITalk se materializa en un widget flotante inyectado directamente en la interfaz del restaurante. Cualquiera que navegue por la web puede desplegar el chat y ponerse a trastear con la IA en tiempo real, comprobando de primera mano cómo el agente comprende el lenguaje natural.



Como complemento fundamental a la experiencia del cliente, el módulo web incorpora un "Dashboard de Desarrollo" (Vista Dev) protegido por contraseña. Este panel interno opera como el centro de mando del restaurante y es donde el dueño puede ver las reservas que se han hecho en sus restaurantes y toda la información asociada a estas.

A nivel técnico, utiliza un backend estructurado en PHP que se conecta de forma segura a PostgreSQL mediante la extensión PDO. En el lado del cliente, se ha implementado un sistema de peticiones asíncronas en JavaScript, lo que permite que la tabla de gestión se actualice en tiempo real de forma automática y sin recargar la página, evitando parpadeos visuales en la interfaz.

La plataforma se aloja sobre una máquina servidor utilizando Nginx como servidor web principal. Toda la arquitectura funciona bajo el protocolo HTTPS mediante certificados SSL/TLS. Dado que el sistema maneja credenciales de acceso de administradores y datos personales de los clientes en las reservas.

```
● nginx.service - nginx - high performance web server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Tue 2026-05-05 16:24:03 CEST; 2min 1s ago
       Docs: https://nginx.org/en/docs/
    Process: 1314 ExecStart=/usr/sbin/nginx -c ${CONFFILE} (code=exited, status=0/SUCCESS)
   Main PID: 1327 (nginx)
      Tasks: 3 (limit: 4598)
     Memory: 4.8M (peak: 4.9M)
        CPU: 23ms
     CGroup: /system.slice/nginx.service
             ├─1327 "nginx: master process /usr/sbin/nginx -c /etc/nginx/nginx.conf"
             ├─1328 "nginx: worker process"
             └─1329 "nginx: worker process"

may 05 16:24:03 lena systemd[1]: Starting nginx.service - nginx - high performance web server...
may 05 16:24:03 lena systemd[1]: Started nginx.service - nginx - high performance web server.
~
```

Además, la red se ha creado usando el estándar HTTP/3 basado en el protocolo QUIC. Esta mejora en la infraestructura optimiza al máximo las peticiones asíncronas del Dashboard, asegurando que el flujo de datos en tiempo real entre PostgreSQL, PHP y el navegador sea ultrarrápido y ofrezca la experiencia impecable que exige un producto profesional.

```
usuario@lena:~$ /snap/bin/curl -k --http3-only -I https://127.0.0.1/
HTTP/3 200
server: nginx/1.29.8
date: Tue, 05 May 2026 15:10:33 GMT
content-type: text/html
content-length: 3169
last-modified: Tue, 14 Apr 2026 14:54:29 GMT
etag: "69de5525-c61"
alt-svc: h3=":443"; ma=86400
x-quic: h3 used
accept-ranges: bytes

usuario@lena:~$
```

4 Conclusiones

4.1 Conclusiones generales del proyecto

El desarrollo de la plataforma AITalk ha demostrado de manera concluyente la inmensa viabilidad técnica y el enorme potencial comercial que supone la integración de agentes conversacionales avanzados en el sector de la pequeña y mediana hostelería. Se ha logrado construir, partiendo de cero, un ecosistema tecnológico completo que automatiza tareas administrativas complejas que tradicionalmente requerían la atención exclusiva de personal humano.

La combinación de orquestadores de flujos de trabajo de código abierto con modelos de lenguaje optimizados ha permitido crear una solución que no solo responde preguntas, sino que actúa y toma decisiones con un alto grado de autonomía. La plataforma trasciende la categoría de proyecto académico para posicionarse como un prototipo funcional y escalable, capaz de interactuar en tiempo real con bases de datos estructuradas y sistemas de gestión de agendas en la nube.

A nivel profesional y de aprendizaje, este proyecto ha proporcionado una experiencia inmersiva en la integración de múltiples arquitecturas. Ha obligado a enfrentar y superar retos característicos de entornos de producción reales, como la gestión de latencias en llamadas a interfaces externas, la resolución de cuellos de botella mediante la migración estratégica de componentes clave y la implementación de redes modernas bajo los estándares de seguridad más recientes. El resultado final consolida el conocimiento adquirido a lo largo del ciclo formativo y demuestra la capacidad para ensamblar tecnologías heterogéneas en un producto cohesionado, robusto y de alto valor añadido.

4.2 Consecución de los objetivos

El balance de la consecución de los objetivos planteados al inicio de esta memoria resulta altamente positivo, habiéndose alcanzado la práctica totalidad de las metas establecidas.

El objetivo general de desarrollar un sistema automatizado e inteligente para la atención al cliente ha sido cumplido de forma íntegra. El orquestador opera de manera estable y centraliza todas las comunicaciones y acciones lógicas sin requerir intervención manual durante su ejecución rutinaria.

En cuanto a los objetivos específicos, la implementación de la inteligencia artificial ha superado las expectativas. Se logró afinar con precisión el comportamiento del modelo de lenguaje, garantizando respuestas rápidas y seguras gracias a la acertada decisión de migrar el motor cognitivo durante la fase de desarrollo para sortear las limitaciones de las cuotas gratuitas iniciales.

El establecimiento del canal de comunicación a través de la aplicación de mensajería se completó con éxito. El sistema es plenamente funcional desde la perspectiva del usuario final, procesando de manera transparente tanto entradas de texto como archivos de voz, integrando de forma fluida los sistemas de conversión entre ambos formatos.

La gestión autónoma de reservas se materializó satisfactoriamente. El agente demuestra una capacidad constante para leer eventos existentes, evitar solapamientos y generar nuevas entradas en los calendarios compartidos, validando así la correcta sincronización con las interfaces de servicios en la nube.

El diseño y despliegue de la base de datos relacional cumple con las exigencias del proyecto. Se ha consolidado una estructura normalizada que soporta eficazmente las consultas dinámicas de la inteligencia artificial, manteniendo un control estricto sobre las capacidades de los locales y centralizando la memoria a largo plazo de las interacciones.

4.3 Valoración de la metodología y planificación

El análisis retrospectivo del proceso de desarrollo confirma que la adopción de metodologías de trabajo ágiles fue una decisión instrumental para el éxito del proyecto. La estructuración de las tareas mediante tableros visuales y el enfoque iterativo dotaron al equipo de la flexibilidad necesaria para gobernar un desarrollo de alta complejidad técnica.

La planificación inicial demostró ser coherente en términos generales, aunque la realidad del trabajo con tecnologías emergentes exigió realizar ajustes críticos sobre la marcha. Las fases dedicadas a la investigación y las pruebas de los modelos de lenguaje demandaron un volumen de tiempo superior al estimado inicialmente. Las inestabilidades y latencias detectadas al interactuar con las primeras interfaces de programación forzaron un replanteamiento de la arquitectura cognitiva.

Lejos de suponer un fracaso en la planificación, la metodología ágil implementada permitió pivotar de forma rápida y ordenada. La decisión de migrar hacia herramientas alternativas se gestionó de manera eficiente, redistribuyendo las tareas sin que el calendario global de entregas se viera comprometido. Este proceso subraya la idoneidad del enfoque seleccionado, demostrando que una planificación no debe ser un esquema rígido e inamovible, sino un marco de referencia vivo que permite al equipo adaptarse, resolver imprevistos y asegurar la viabilidad técnica del producto final.

4.4 Visión de futuro

El prototipo funcional de AITalk sienta unas bases sólidas que abren un abanico extenso de posibilidades para futuras expansiones y mejoras. Aunque el

sistema actual cubre con excelencia el núcleo de las necesidades de gestión de reservas, la arquitectura modular diseñada permite imaginar evoluciones técnicas de gran impacto.

A corto y medio plazo, una línea de trabajo evidente sería la expansión de las capacidades multimodales del agente. La incorporación de modelos de visión artificial permitiría a los usuarios enviar fotografías de platos, recibiendo respuestas automáticas sobre ingredientes o solicitando su reserva basándose en estímulos visuales. De igual forma, el desarrollo de capacidades conversacionales bidireccionales en tiempo real abriría la puerta a integrar el agente con centralitas de voz sobre protocolo de internet, permitiendo gestionar reservas mediante llamadas telefónicas tradicionales de manera completamente autónoma.

A nivel de infraestructura y gestión de datos, el sistema podría evolucionar hacia un modelo de tenencia múltiple más sofisticado. Esto implicaría expandir las capacidades del panel de administración web para gestionar grandes volúmenes de métricas, ofreciendo a los propietarios de múltiples locales análisis predictivos sobre la demanda de reservas, picos de ocupación o tendencias en la selección de menús, basados en la inteligencia de negocios extraída del registro histórico de las conversaciones de la plataforma.

En definitiva, la plataforma construida no representa un producto estático y finalizado, sino un motor de automatización con un recorrido inmenso, preparado para asimilar las continuas innovaciones del campo de la inteligencia artificial y aplicarlas directamente a la optimización operativa del mundo empresarial.

5. Glossario

- **API:** Conjunto de reglas, protocolos y herramientas que permiten la comunicación bidireccional y el intercambio de datos entre diferentes aplicaciones de software de manera estructurada y segura.

- **Base de datos relacional:** Sistema de almacenamiento de información digital organizado mediante el uso de tablas interconectadas que comparten puntos de datos comunes, garantizando la integridad referencial y permitiendo consultas complejas.

- **Bot:** Programa informático diseñado para operar de forma autónoma y ejecutar tareas repetitivas o interactuar con usuarios a través de plataformas digitales y redes de mensajería.

- **Cuello de botella:** Situación que se produce en un sistema informático o red de comunicaciones cuando la capacidad de un componente específico limita severamente el rendimiento general de todo el conjunto.

- **Disparador:** Mecanismo o bloque de código programado a nivel de base de datos que se ejecuta automáticamente como respuesta a un evento específico, como la inserción, modificación o eliminación de un registro.

- **HTTP/3:** Tercera versión principal del protocolo de comunicación utilizado para la transferencia de información en la red mundial, caracterizado por utilizar un transporte de datagramas que reduce drásticamente la latencia de las conexiones.

- **Ingeniería de instrucciones:** Disciplina orientada al diseño, estructuración y optimización de los textos de entrada suministrados a los modelos de inteligencia artificial con el objetivo de obtener respuestas precisas, acotadas y seguras.

- **Jailbreak:** Proceso o conjunto de técnicas utilizadas por un usuario malintencionado para evadir las restricciones lógicas y los controles de seguridad programados en un modelo de lenguaje, forzándolo a generar respuestas no autorizadas.

- **Latencia:** Retardo temporal o suma de tiempos de espera que experimenta un paquete de datos u orden desde que es emitido por un nodo hasta que es recibido y procesado por su destino en una arquitectura de red.

- **LLM:** Sistema de inteligencia artificial fundamentado en arquitecturas de aprendizaje profundo, entrenado con cantidades masivas de texto, que posee la capacidad de comprender, interpretar, resumir y generar lenguaje humano de forma natural.

- **n8n:** Herramienta informática de automatización de flujos de trabajo basada en una arquitectura visual de nodos, distribuida bajo licencias de código abierto, que actúa como orquestador central conectando múltiples servicios e interfaces.

- **QUIC**: Protocolo de transporte de red encriptado por defecto, diseñado para proporcionar conexiones más rápidas y seguras, sirviendo como fundamento subyacente para las nuevas generaciones de transferencia web asíncrona.

- **Scrum:** Marco de trabajo de la metodología ágil orientado a la gestión y desarrollo de proyectos complejos, caracterizado por dividir el trabajo en entregas parciales y regulares para maximizar la capacidad de adaptación frente a imprevistos.

- **SGBD:** Conjunto de programas y herramientas de software que permiten crear, administrar, proteger y consultar de manera eficiente las estructuras de información persistente de un sistema.

6. Nota sobre el uso de inteligencia artificial

Durante la elaboración de esta memoria se han utilizado herramientas de inteligencia artificial de manera puntual como apoyo en tareas de redacción, corrección y mejora de la claridad expositiva de algunos apartados. Estas herramientas han servido únicamente como asistencia lingüística y de estilo, facilitando una mejor organización y formulación del texto.

No obstante, todo el contenido desarrollado en este trabajo incluyendo las ideas, el planteamiento, las decisiones técnicas, el análisis realizado y las conclusiones obtenidas es íntegramente propio. La utilización de inteligencia artificial no ha sustituido en ningún momento el trabajo intelectual, crítico ni creativo llevado a cabo durante el desarrollo del proyecto, sino que se ha limitado a actuar como una herramienta de apoyo para mejorar la calidad y la precisión de la expresión escrita.

7. Bibliografia

Documentación de n8n:
https://docs.n8n.io/
https://github.com/n8n-io/n8n
https://www.nocodehackers.es/herramientas-no-code/n8n

Apis IA:
https://docs.mistral.ai/
https://ai.google.dev/api
https://deepmind.google/models/gemini-audio/
https://elevenlabs.io/docs/api-reference/introduction

API de Google Calendar:
https://developers.google.com/workspace/calendar/api/guides/overview
https://www.unipile.com/guide-to-google-calendar-api-integration/
https://www.postman.com/postman/google-api-workspace/documentation/54xuf9z/google-calendar-api?sideView=agentMode

Documentación de la versión de postgres utilizada en el proyecto:
https://www.postgresql.org/docs/17/index.html
https://hackceleration.com/es/postgres-n8n/

Documentación de los bots de Telegram:
https://core.telegram.org/bots/api
https://aguacatec.es/automatizar-telegram-con-n8n/