

**Parcial Api Con ExpressJS**

Cristhian Andrey Poveda Gaviria   
ID: 843183

Corporación Universitaria Minuto de Dios

Base de Datos Masivas

Ingeniería De Sistemas

2025

Contenido

[Introducción 3](#_Toc1900066378)

[Objetivos 3](#_Toc1472256638)

[Desarrollo 3](#_Toc2074800758)

[Conclusiones 8](#_Toc543985188)

[Referencias 8](#_Toc1449622821)

**Taller Api Con ExpressJS**

# Introducción

El presente taller presenta el desarrollo de una API REST utilizando el framework Express.js de Node.js. Esta API tiene como objetivo gestionar la información de una cadena de comidas rápidas, incluyendo datos sobre restaurantes, empleados, productos y demás.

Para lograr esto, utilizaremos varias herramientas como PostgreSQL para la gestión de la base de datos, Supabase como plataforma que nos facilita el uso de PostgreSQL, y Postman para realizar pruebas de la API, verificando la correcta implementación de las rutas y la integridad de los datos. Finalmente, utilizaremos GitHub para el posible control de versiones y la colaboración facilitando la entrega del proyecto.

# Objetivos

* Desarrollar una API REST funcional que permita realizar operaciones sobre las tablas de la base de datos.
* Implementar la conexión a la base de datos: Establecer una conexión exitosa y funcional con una base de datos PostgreSQL, utilizando Supabase.
* Ejecutar consultas de PostgreSQL, exponiendolas a través de la API las consultas en el taller, demostrando la capacidad de realizar operaciones directamente en la base de datos.
* Utilizar Postman para realizar pruebas, asegurando su correcto funcionamiento.
* Documentar la API y las consultas: Generar una documentación clara y completa de la API, incluyendo la descripción de las rutas, los parámetros de entrada/salida y el funcionamiento de las consultas nativas.

# Desarrollo

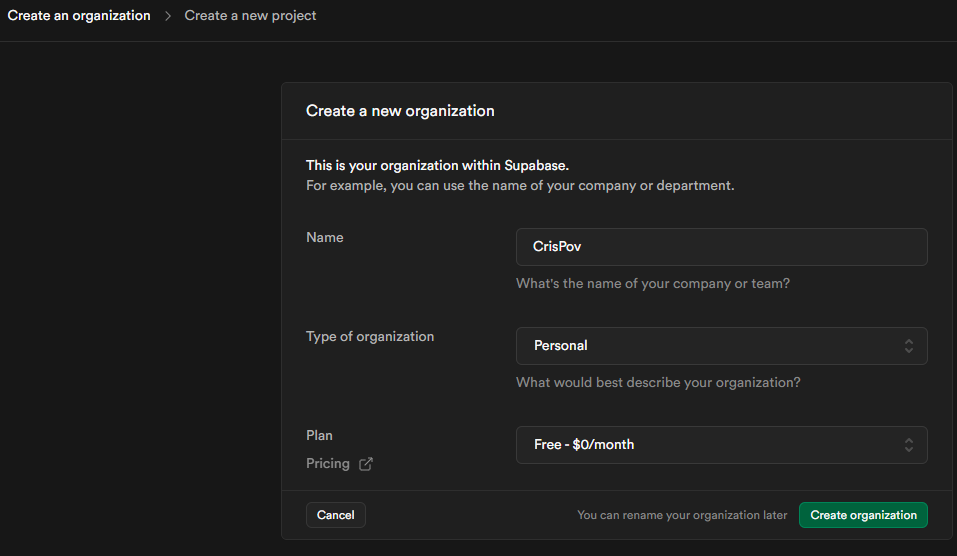
Express js es un framework para Node.js que permite desarrollar aplicaciones web y APIs de manera más optimizada y flexible, ofreciendo muchas funcionalidades facilitando la gestión de rutas, solicitudes y otros aspectos de las aplicaciones web.

Por otro lado, una API (Interfaz de Programación de Aplicaciones), es un conjunto de reglas y protocolos que permite que las aplicaciones se comuniquen entre ellas. Definen cómo las solicitudes y respuestas deben formatearse y qué operaciones están disponibles, como puede ser acceder a datos o funcionalidades de otro sistema.

Supabase es un es un gestor de proyectos de base de con la diferencia de que la base de datos PostgreSQL estará alojada en la nube, permitiendo una autenticación, un almacenamiento de archivos, una API autogenerada, y una interfaz web con muchas de las funciones de un gestor de base de datos.

Conectaremos Supabase con apis generadas usando Express para hacer peticiones y operaciones CRUD, para conectarlos, simplemente necesitamos las credenciales de conexión de la base de datos en Supabase (host, puerto, nombre de la base de datos, usuario y contraseña), que encontramos en el panel de configuración de Supabase.

Ingresaremos a *supabase.com* para registrarnos, usando nuestra cuenta de GitHub, luego configuramos nuestra “organización”, escogiendo el plan gratuito, y creamos un proyecto en supabase.

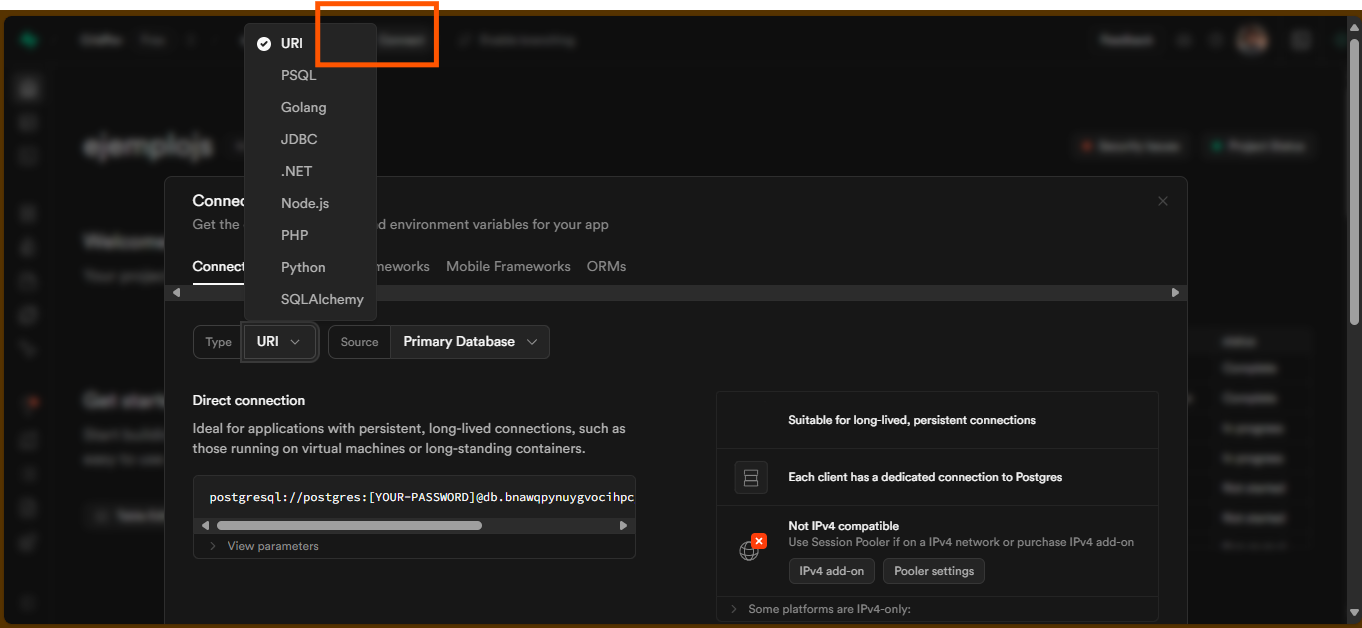
Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

* *BaseDatosMasivas*

Creamos una posible organización, luego asignamos un nombre al proyecto, y una contraseña para la base de datos.

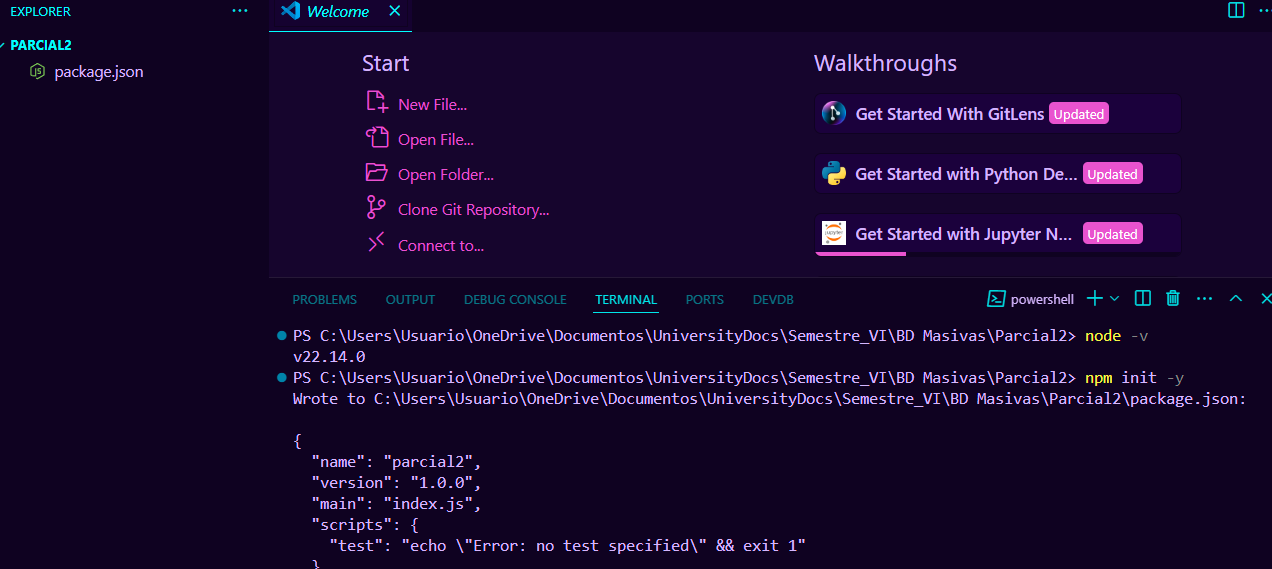
En pantalla, Vamos al apartado donde dice **“connect”,** para ver las direcciones y parámetros necesarias para conectar con esta base de datos en la nube.



* *postgresql://postgres:[YOUR-PASSWORD]@db.bnawqpynuygvocihpcjx.supabase.co:5432/postgres*

Pero antes, necesitamos tener un entorno configurado para crear las APIs y la conección. Usaremos Visual Studio.

Abrimos un folder vacío, Inicializamos el proyecto con node (*npm init -y*), y descargaremos las dependencias a usar, a la vez que verificamos el tener instalado el Node.js:



* **Express**: Framework para Node.js que nos facilita la creación de aplicaciones web y APIs.
* **PG**: Cliente para conectarse a bases de datos PostgreSQL desde Node.js.
* **Dotenv**: Permite gestionar las variables de entorno desde un archivo .env. Es útil para mantener datos sensibles fuera del código fuente principal.
* **CORS (Cross-Origin Resource Sharing)**: Es un *middleware* que nos habilita el intercambio de recursos entre diferentes dominios, lo necesitamos para que el servidor responda a solicitudes de otros sitios web.

Se nos habrán descargado y se nos mostrarán:

Texto

El contenido generado por IA puede ser incorrecto.

Crearemos dos archivos, el primero, que nos asegurará la conexión:

Texto

El contenido generado por IA puede ser incorrecto.

En este utilizaremos la URI que nos dio supabase, solo remplazando el apartado de contraseña.

El siguiente archivo será donde crearemos las apis.

Texto

El contenido generado por IA puede ser incorrecto.

Definimos e importamos las constantes que vamos a utilizar.

Antes de crear las apis, crearemos las tablas en la base de datos en la nube que tenemos en Supabase:

En el apartado SQL editor, en Supabase podesmo escribir el código para crearlas, aunque también está el apartado para crearlas visualmente.

Pantalla de computadora con letras

El contenido generado por IA puede ser incorrecto. Texto

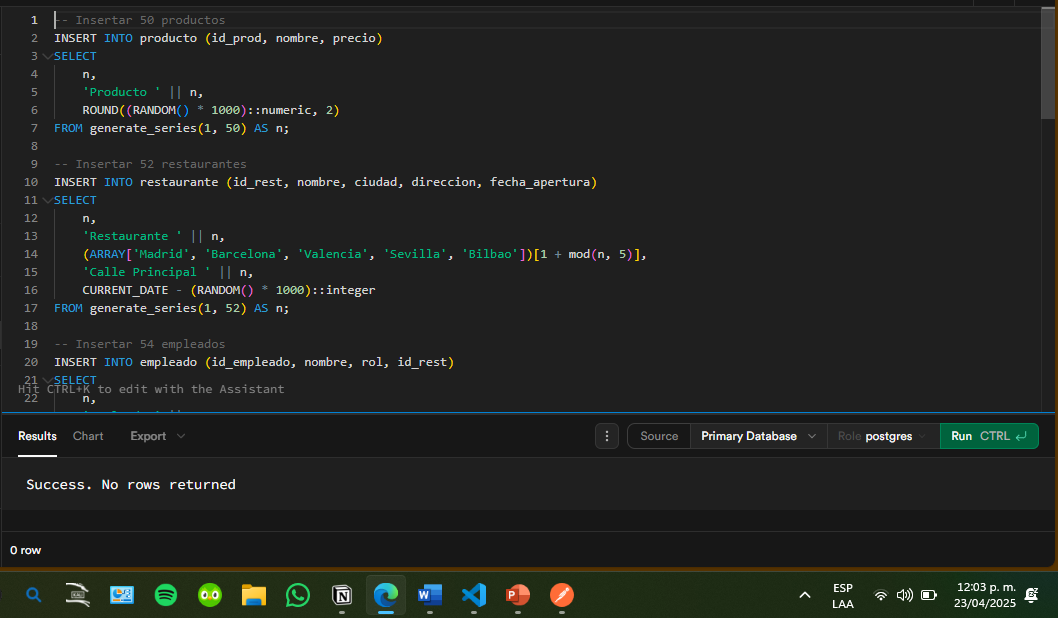
El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

En la visualización nos queda tal que así.

Añadimos los datos:



*-- Insertar 50 productos*

*INSERT INTO producto (id\_prod, nombre, precio)*

*SELECT*

*n,*

*'Producto ' || n,*

*ROUND((RANDOM() \* 1000)::numeric, 2)*

*FROM generate\_series(1, 50) AS n;*

*-- Insertar 52 restaurantes*

*INSERT INTO restaurante (id\_rest, nombre, ciudad, direccion, fecha\_apertura)*

*SELECT*

*n,*

*'Restaurante ' || n,*

*(ARRAY['Madrid', 'Barcelona', 'Valencia', 'Sevilla', 'Bilbao'])[1 + mod(n, 5)],*

*'Calle Principal ' || n,*

*CURRENT\_DATE - (RANDOM() \* 1000)::integer*

*FROM generate\_series(1, 52) AS n;*

*-- Insertar 54 empleados*

*INSERT INTO empleado (id\_empleado, nombre, rol, id\_rest)*

*SELECT*

*n,*

*'Empleado ' || n,*

*(ARRAY['Camarero', 'Cocinero', 'Gerente', 'Recepcionista', 'Limpieza'])[1 + mod(n, 5)],*

*1 + mod(n, 52) -- Para asegurarnos que id\_rest existe*

*FROM generate\_series(1, 54) AS n;*

*-- Insertar 56 pedidos*

*INSERT INTO pedido (id\_pedido, fecha, id\_rest, total)*

*SELECT*

*n,*

*CURRENT\_DATE - (RANDOM() \* 365)::integer,*

*1 + mod(n, 52), -- Para asegurarnos que id\_rest existe*

*0 -- El total será actualizado después*

*FROM generate\_series(1, 56) AS n;*

*-- Insertar 58 detalles de pedido*

*INSERT INTO detalle\_pedido (id\_detalle, id\_pedido, id\_prod, cantidad, subtotal)*

*SELECT*

*n,*

*1 + mod(n, 56), -- Para asegurarnos que id\_pedido existe*

*1 + mod(n, 50), -- Para asegurarnos que id\_prod existe*

*1 + (RANDOM() \* 5)::integer,*

*0 -- El subtotal será calculado después*

*FROM generate\_series(1, 58) AS n;*

*-- Actualizar los subtotales en detalle\_pedido*

*UPDATE detalle\_pedido*

*SET subtotal = dp.cantidad \* p.precio*

*FROM detalle\_pedido dp*

*JOIN producto p ON dp.id\_prod = p.id\_prod;*

*-- Actualizar los totales en pedido*

*UPDATE pedido p*

*SET total = (*

*SELECT SUM(subtotal)*

*FROM detalle\_pedido dp*

*WHERE dp.id\_pedido = p.id\_pedido*

*);*

* El anterior codigo para agregar los datos fue proporcionado por Claude.ia <https://claude.ai/chat>

Organizamos las carpetas del proyecto de la siguiente manera:

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Definiendo un archivo de controlador a cada tabla

# Conclusiones

ELa combinación de las herramientas representa un equilibrio interesante entre facilidad de uso (característica de servicios BaaS) y control detallado (propio de herramientas de escritorio como pgAdmin).

# Referencias

Aplyca. (2023, febrero 28). Supabase: una alternativa ágil de código abierto. Recuperado de <https://www.aplyca.com/blog/blog-supabase-una-alternativa-agil-de-codigo-abierto>

Generación de datos con: <https://claude.ai/chat/>