

Api

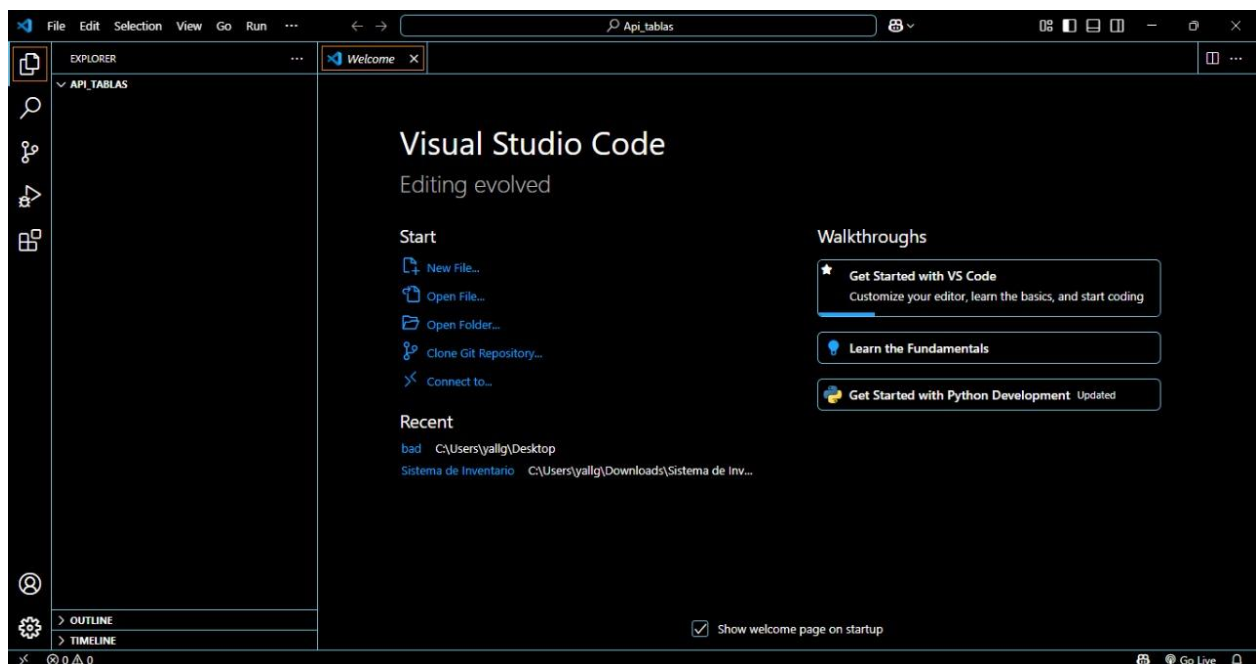
Angie Lorena Jiménez Porras

Ing. Sistemas, UNIMINUTO

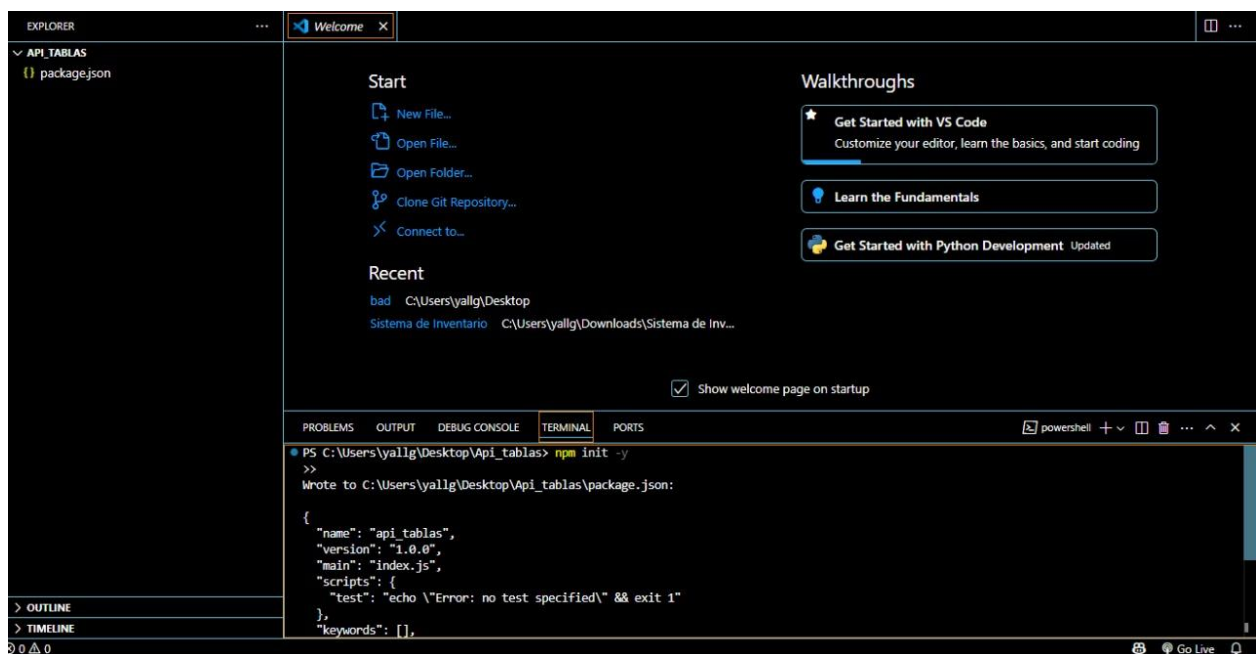
William Alexander Matallana Porras

Bases de datos masivas

11/04/2025



Inicamos creando un acarpeta y visualizamos con visual studio code, esta carpeta debe estar vacia



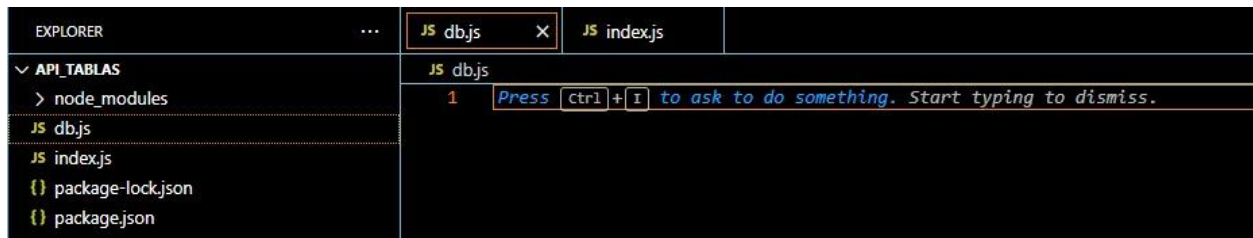
Abrimos la terminal y usamos el comando `npm init -y`: el cual creara un paquete Json el cual almacenará rápidamente archivos como el nombre del proyecto, paquetes y dependencias

```

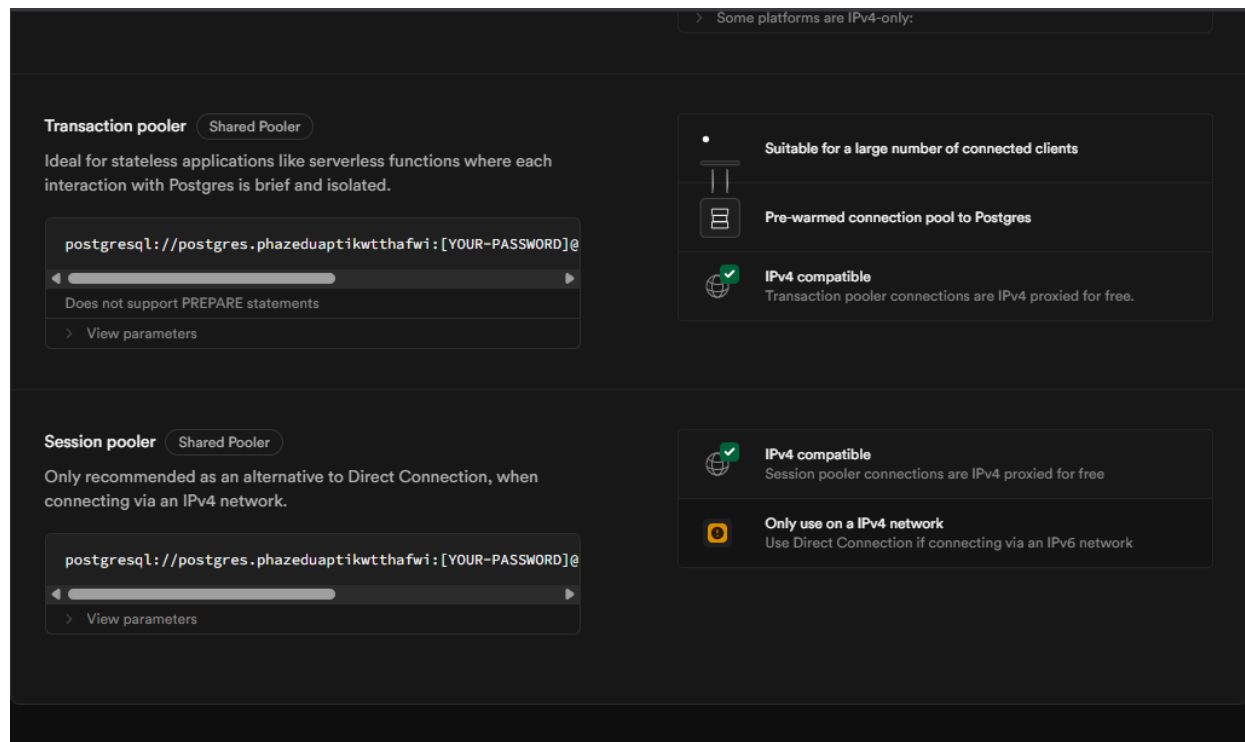
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\yallg\Desktop\Api_tablas> npm install express pg cors
>>
.

```

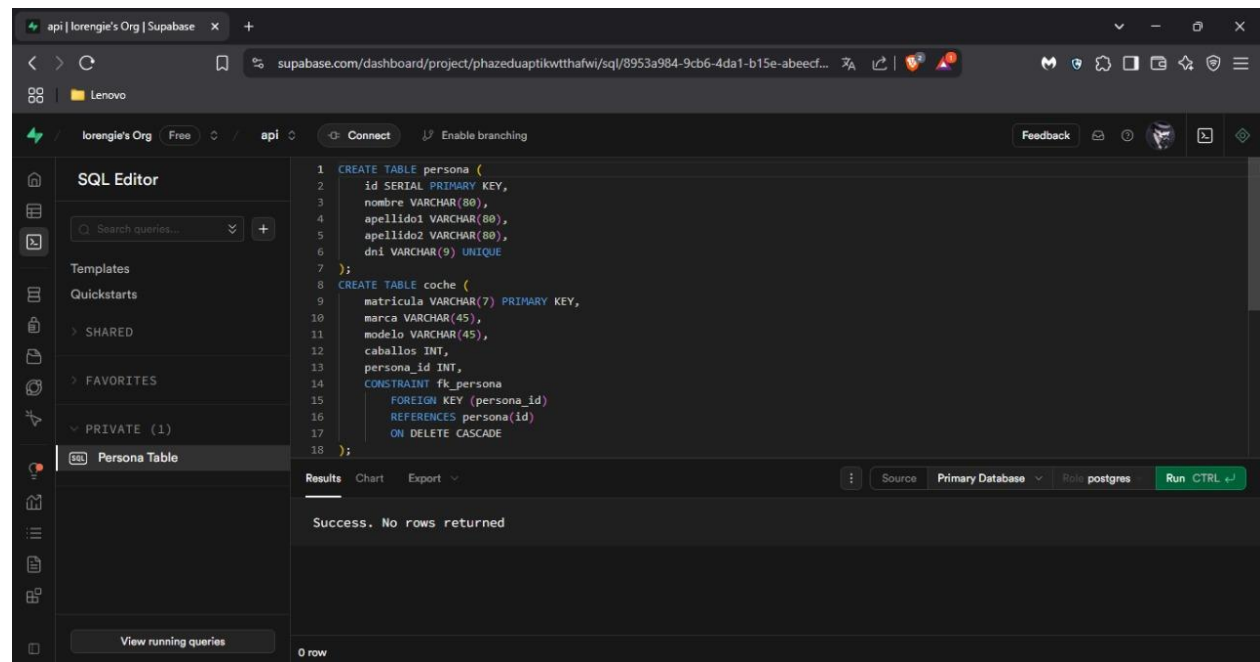
Con el comando nmp install express pg cors: se instalará los servicios de postrges, espress y cors: los cuales estaremos usando para el proyecto



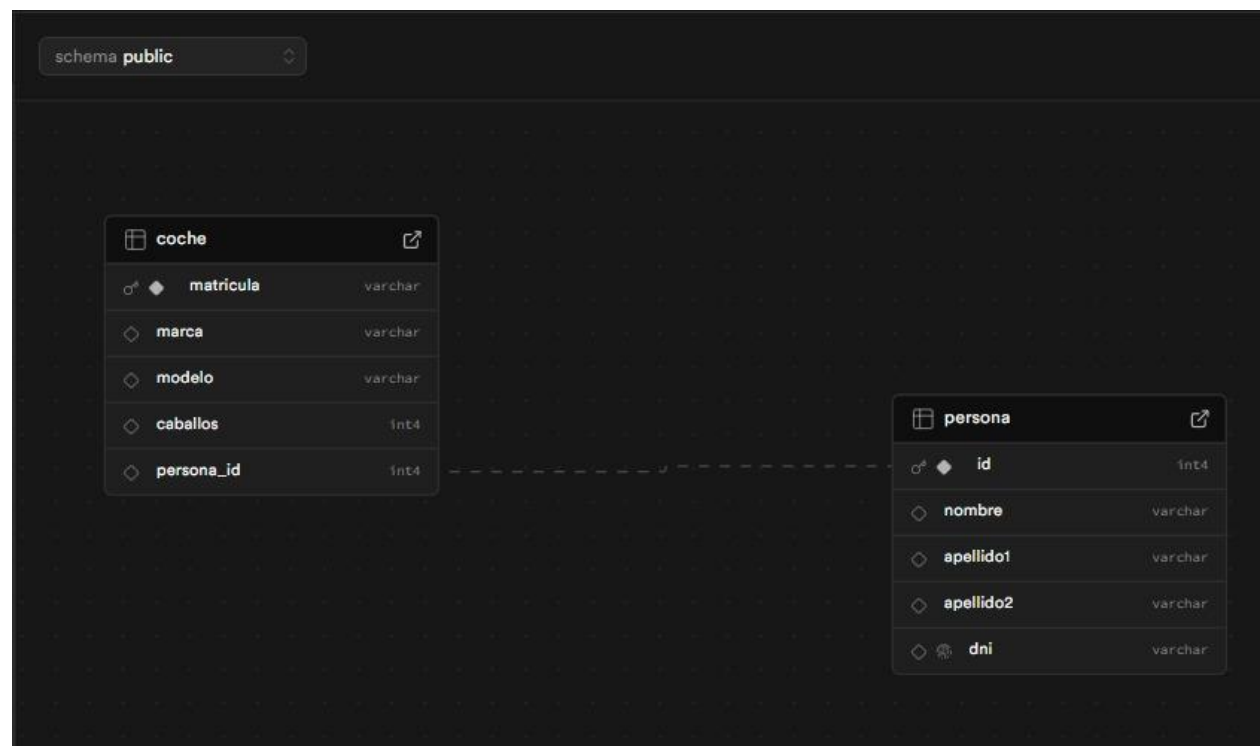
Ahora procedemos a crear 2 archivos, uno db.js es cual servirá para la conexión con supa base e index.js el cual va a tener las api y el puerto que se va a usar



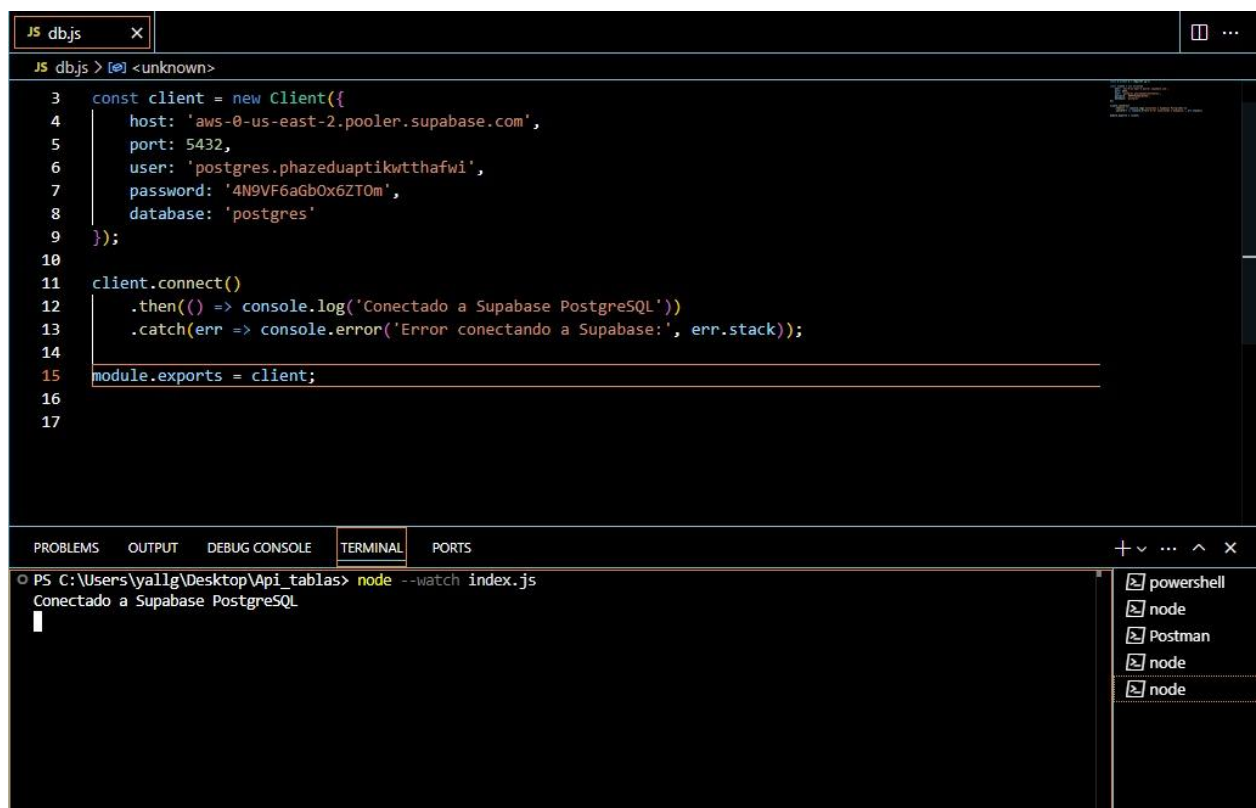
Para poder conectarnos a supa base: creamos un nuevo proyecto y le damos a Connect esto con el fin de poder hacer la conexión para el proyecto



Para crear las tablas y que estén conectadas, vamos a generarlas mediante el SQL Editor que ofrece supabase, teniendo en cuenta los parámetros y la conexión que hay entre las tablas



Una vez generado los comandos, vamos al visualizador de esquemas, lo cual nos genera las tablas con cada campo y la conexión



The image shows a Visual Studio Code editor window with a file named `db.js` open. The code in the editor is as follows:

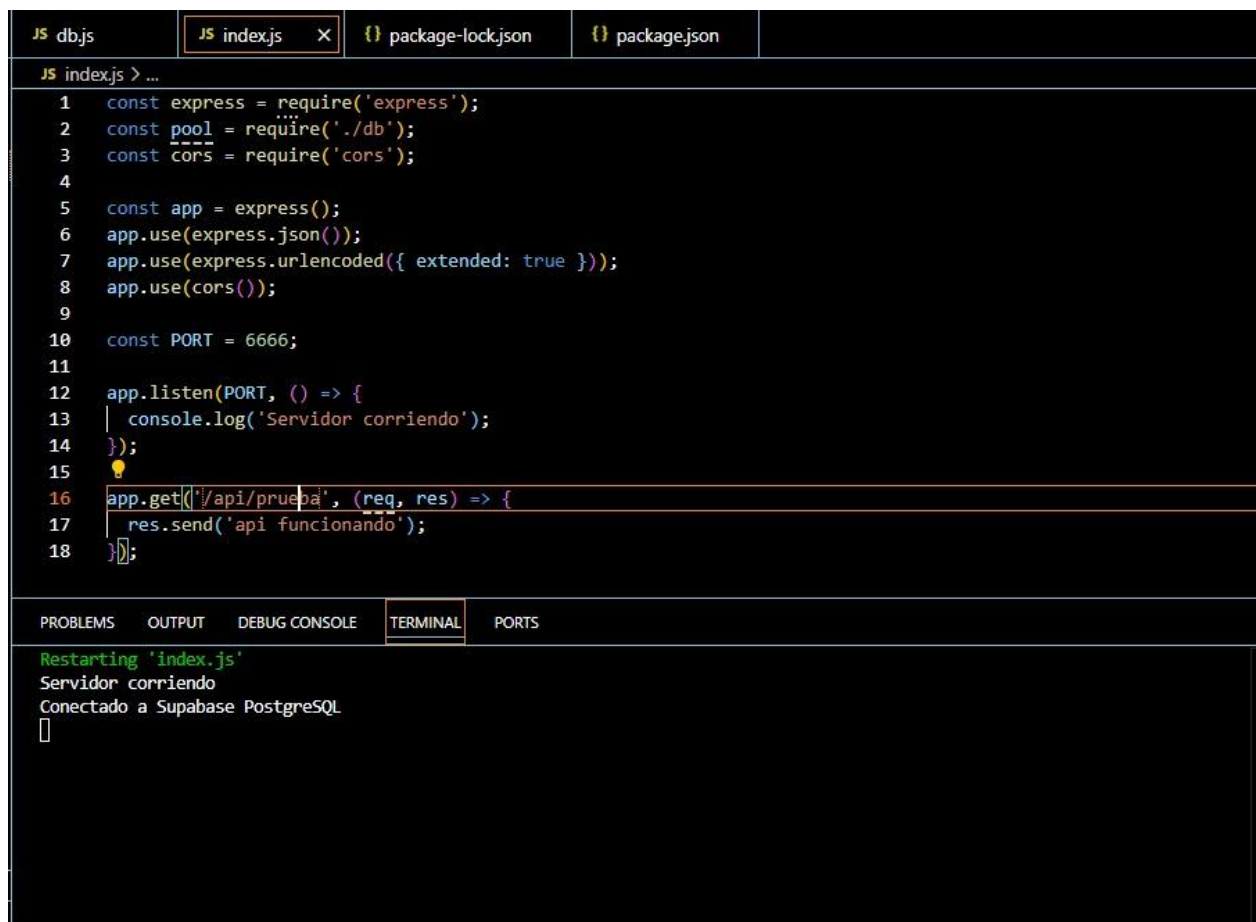
```
3 const client = new Client({
4   host: 'aws-0-us-east-2.pooler.supabase.com',
5   port: 5432,
6   user: 'postgres.phazeduaptikwtthafwi',
7   password: '4N9VF6aGb0x6ZT0m',
8   database: 'postgres'
9 });
10
11 client.connect()
12   .then(() => console.log('Conectado a Supabase PostgreSQL'))
13   .catch(err => console.error('Error conectando a Supabase:', err.stack));
14
15 module.exports = client;
16
17
```

Below the editor, the **TERMINAL** tab is active, showing the command `node --watch index.js` and its output:

```
PS C:\Users\yallg\Desktop\Api_tablas> node --watch index.js
Conectado a Supabase PostgreSQL
```

On the right side of the terminal, a sidebar shows a list of open files: `powershell`, `node`, `Postman`, `node`, and `node`.

el archivo de conexión queda de la siguiente manera con los datos que nos dio supabase para la conexión



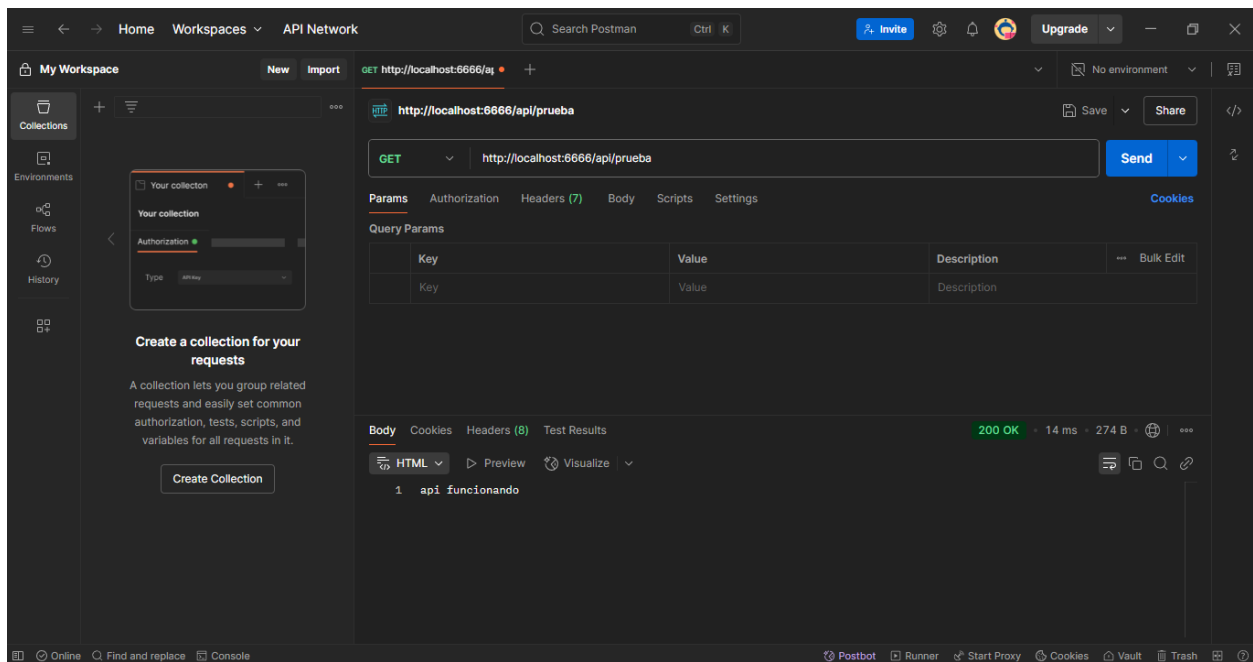
The image shows a VS Code editor with a dark theme. At the top, there are four tabs: 'JS db.js', 'JS index.js' (which is active and has a close button), 'package-lock.json', and 'package.json'. The 'JS index.js' tab contains the following code:

```
1 const express = require('express');
2 const pool = require('./db');
3 const cors = require('cors');
4
5 const app = express();
6 app.use(express.json());
7 app.use(express.urlencoded({ extended: true }));
8 app.use(cors());
9
10 const PORT = 6666;
11
12 app.listen(PORT, () => {
13   console.log('Servidor corriendo');
14 });
15
16 app.get('/api/prueba', (req, res) => {
17   res.send('api funcionando');
18 });
```

Below the code editor, there is a panel with four tabs: 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL' (which is active). The 'TERMINAL' tab shows the following output:

```
Restarting 'index.js'
Servidor corriendo
Conectado a Supabase PostgreSQL
█
```

En el index hacemos un api de prueba para mirar que todo esté funcionando de manera correcta



Mediante el uso de postman ponemos el puerto y la ruta que hayamos usado para crear el api de prueba lo cual nos arrojará un 200 que significa que está todo correcto