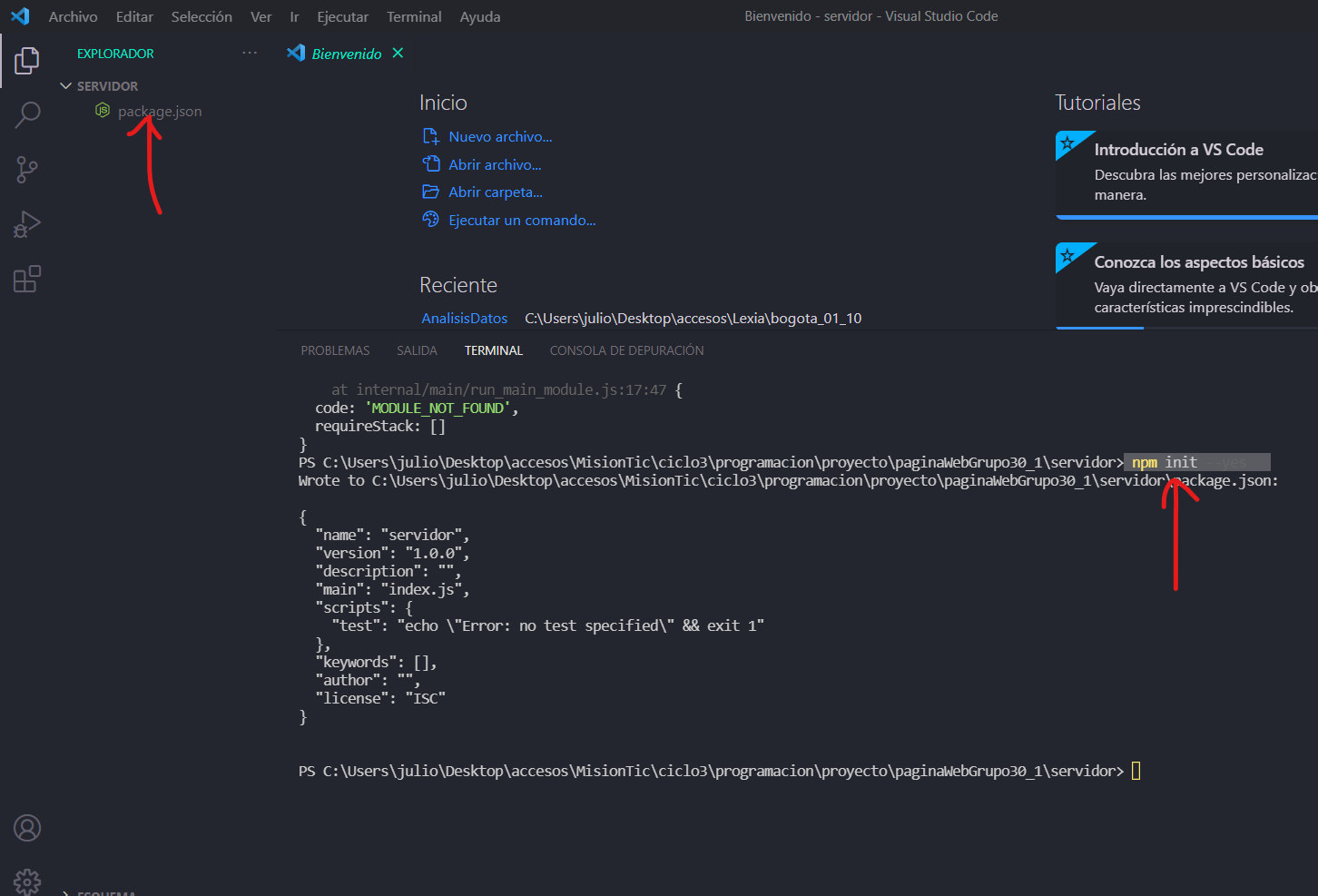
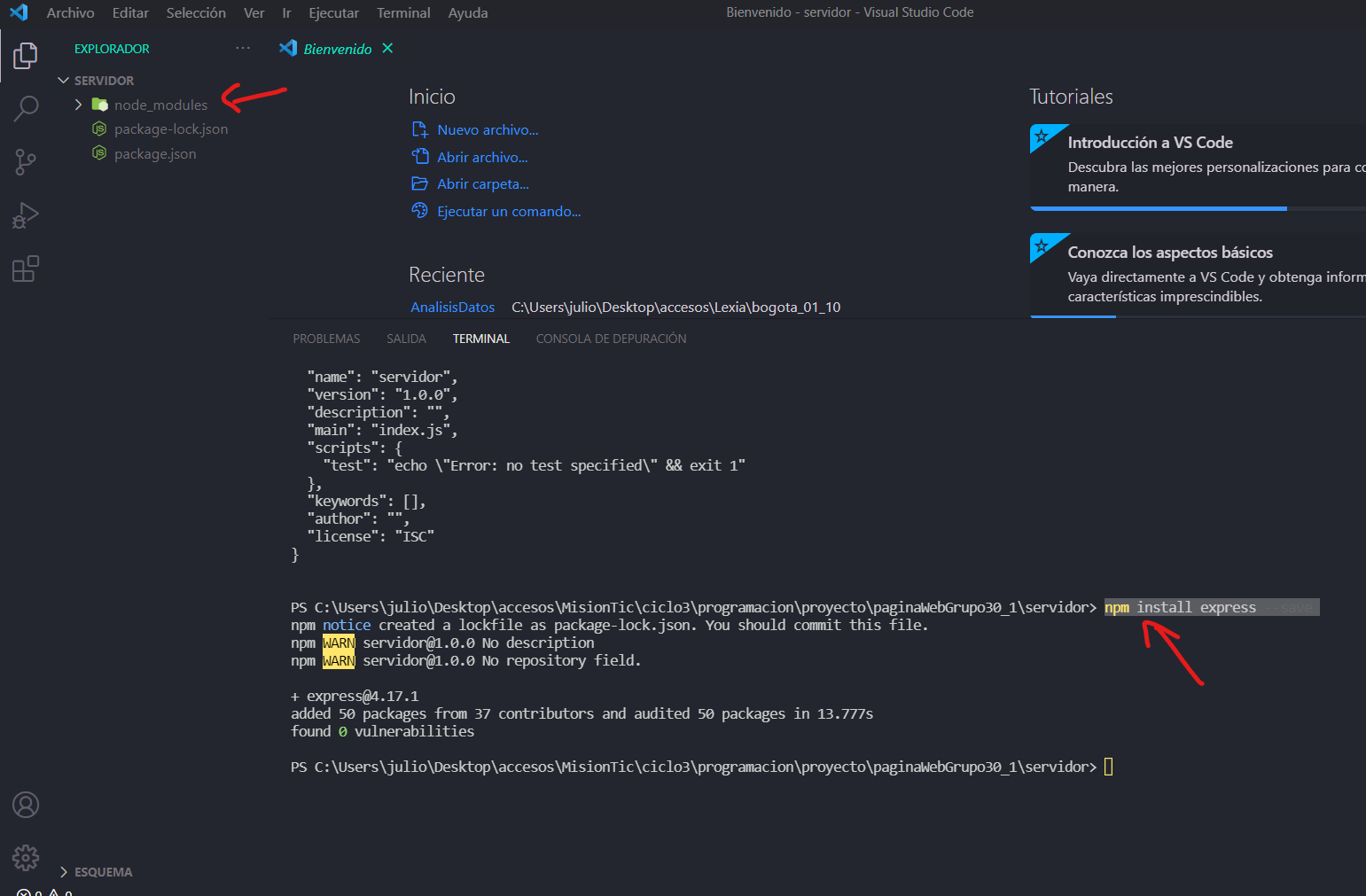
Creación del servidor

* Tener instalado Node.js
* En una carpeta se crea nuevo proyecto en node con el comando npm init –yes el cual genera el archivo package.json.

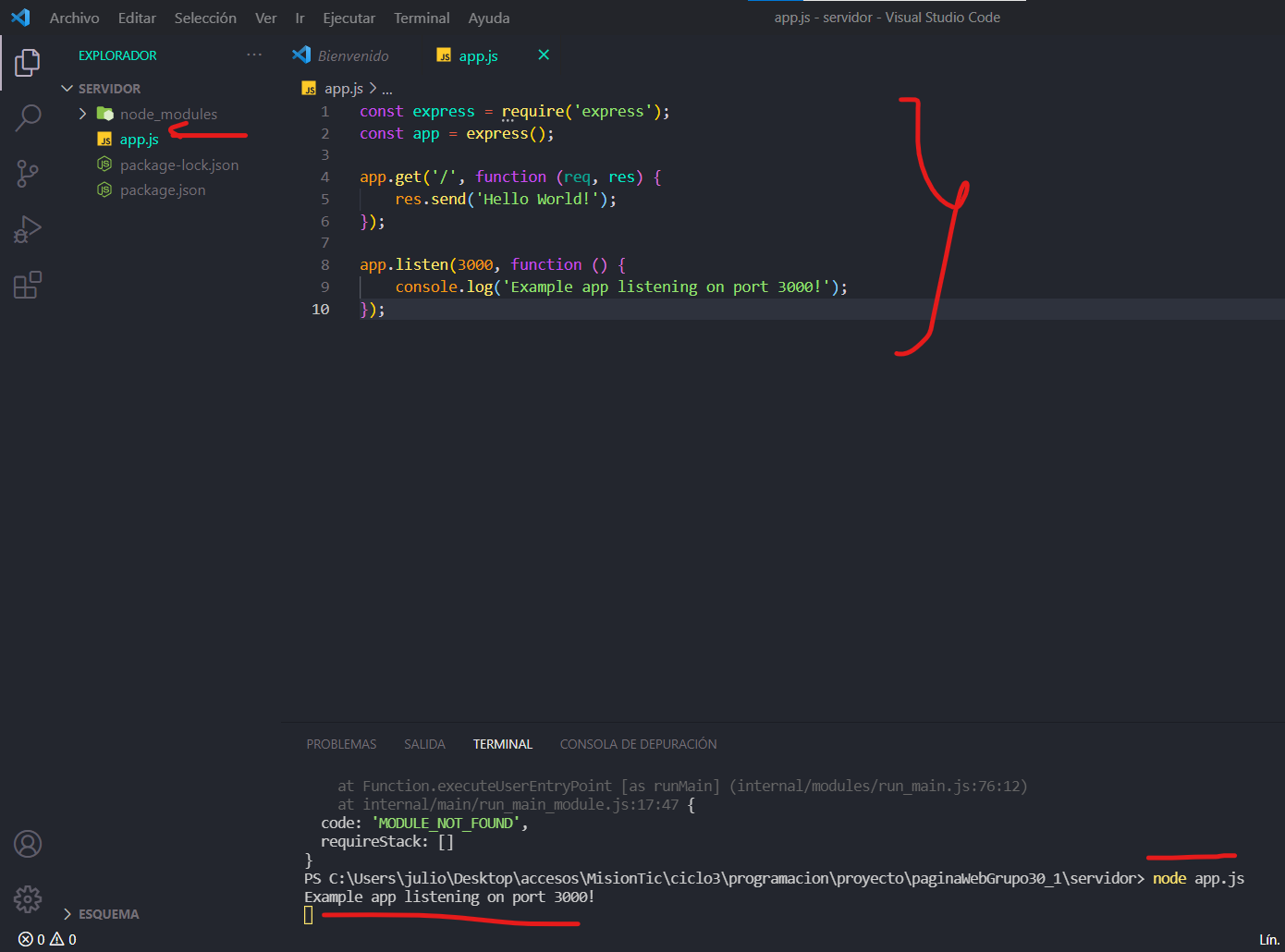


* Se instala express con el comando npm install express --save el cual crea nuevas carpetas y la dependencia de express que permite la conexión a la base de datos

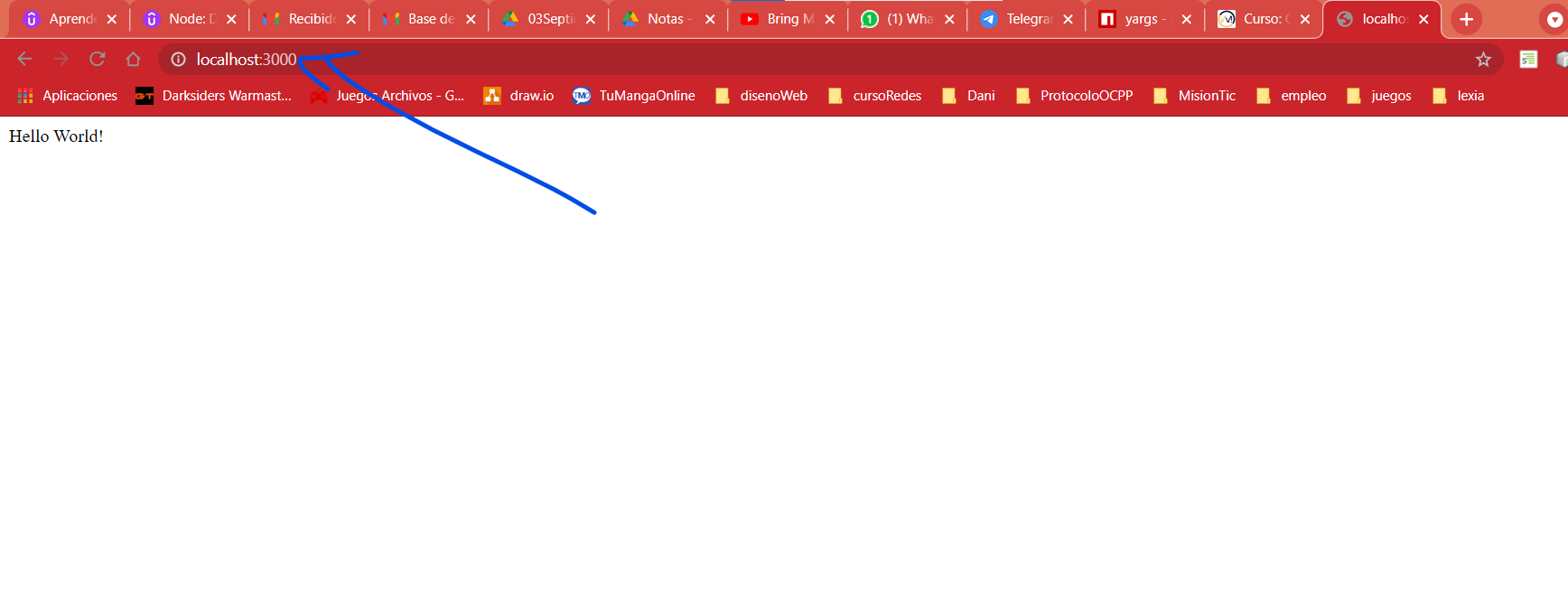


* Se crea un archivo donde se van a agregar las funciones para la conexión con la base de datos y la obtención de las rutas que se creen, se puede ejecutar esto con node app.js

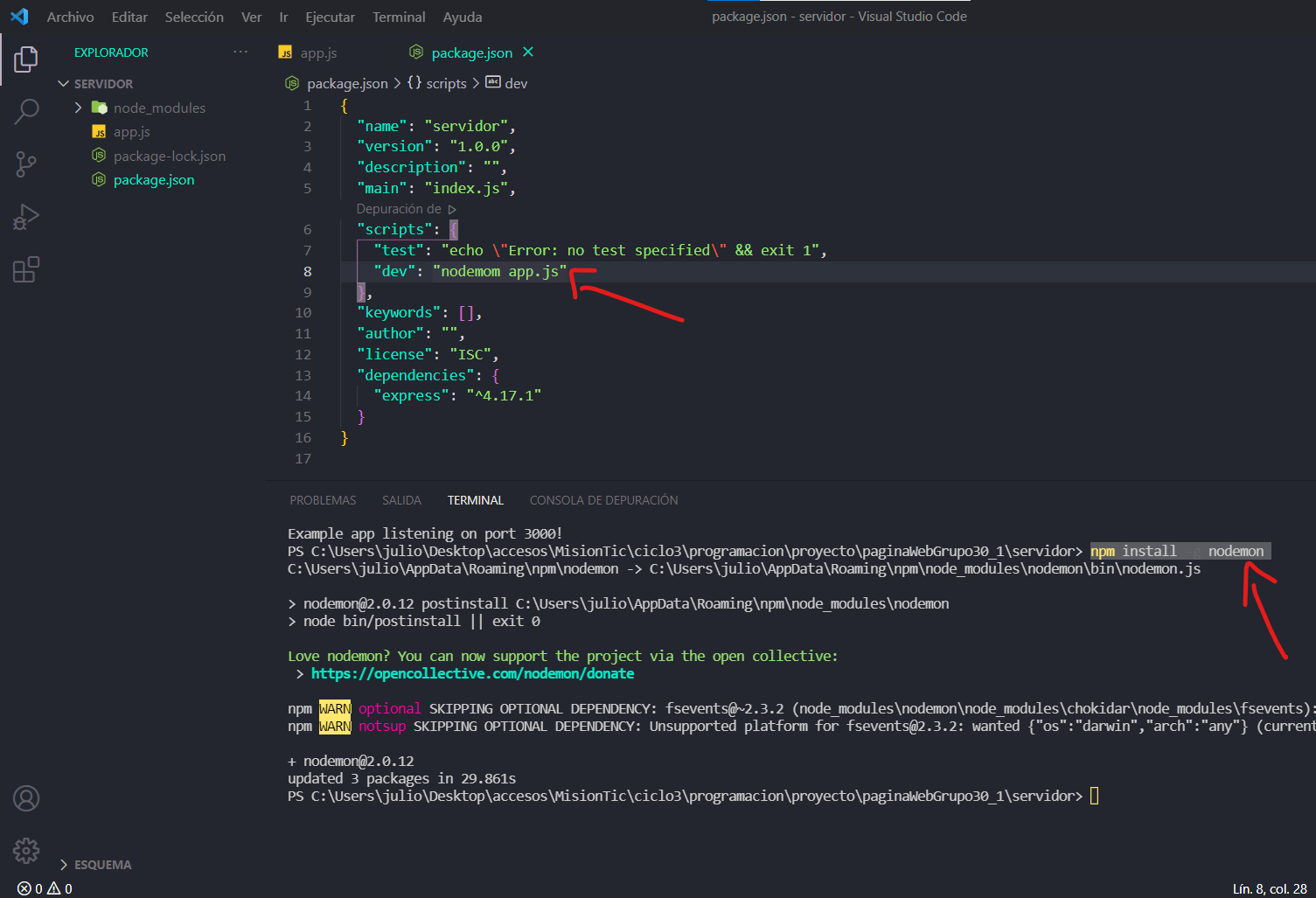
Para probar esto primero se debe importar la dependencia de express y luego configurar una función para que escuche el puerto, luego se crea una función de prueba en una ruta raíz que mande un mensaje para ver si funciona



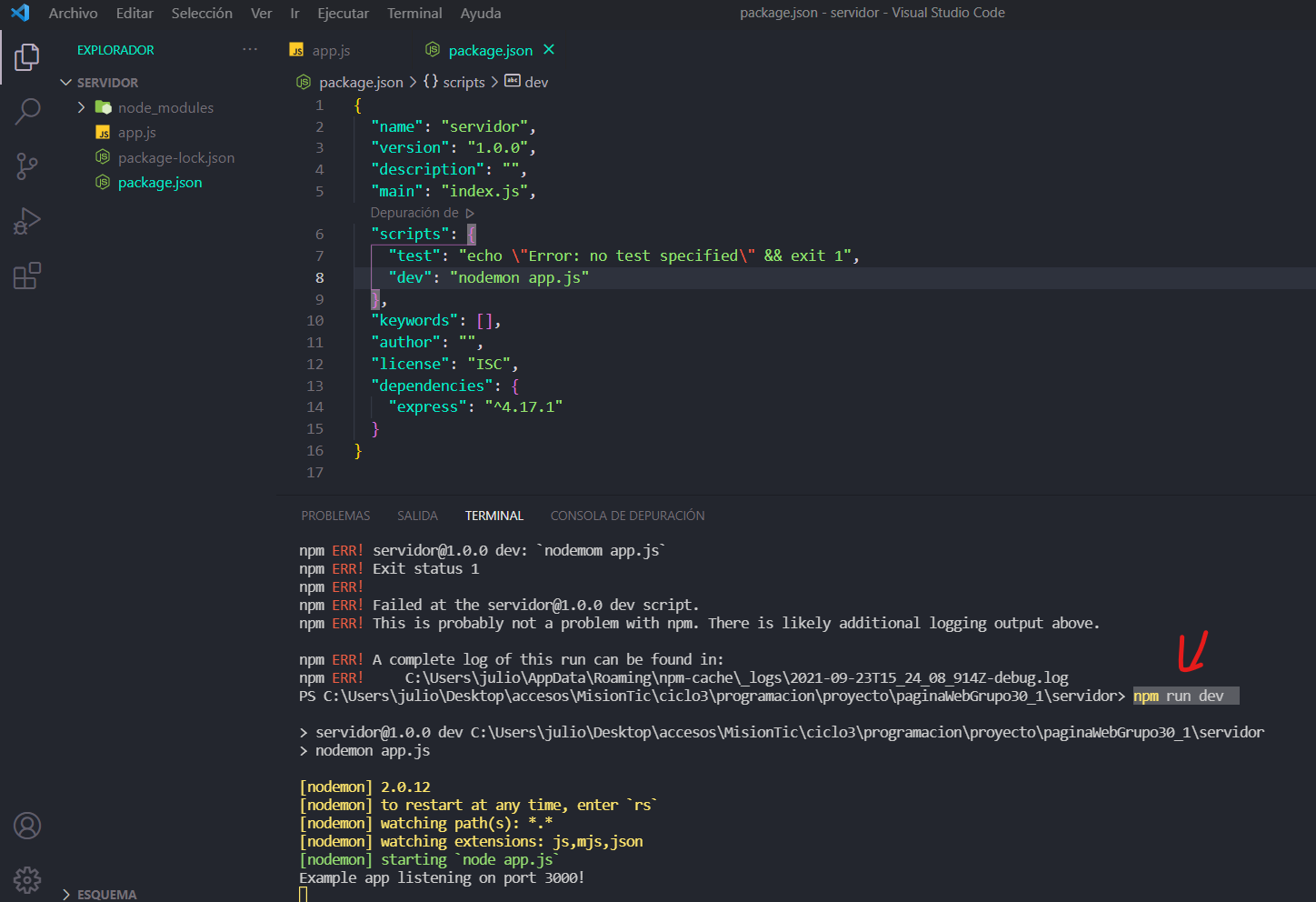
Y si vamos al navegador y entramos al localhost:3000 podemos ver el mensaje



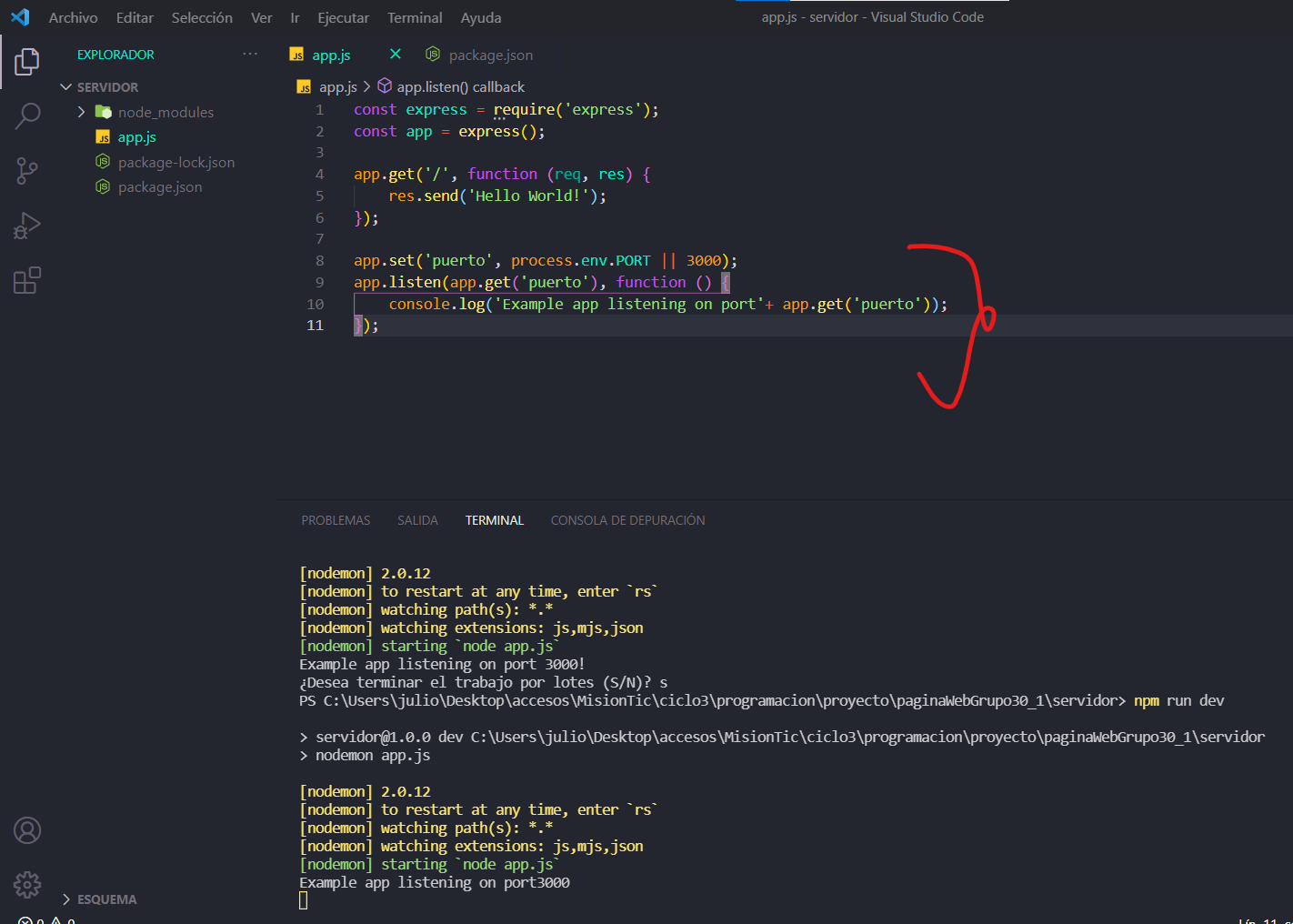
* Luego de poner el servidor en un puerto es necesario instalar nodemon con el comando npm install -g nodemon para que se recarguen lo cambios automáticamente y se debe agregar al script de app.js “dev”: “nodemon app.js” para que se corra el desarrollo el comando.



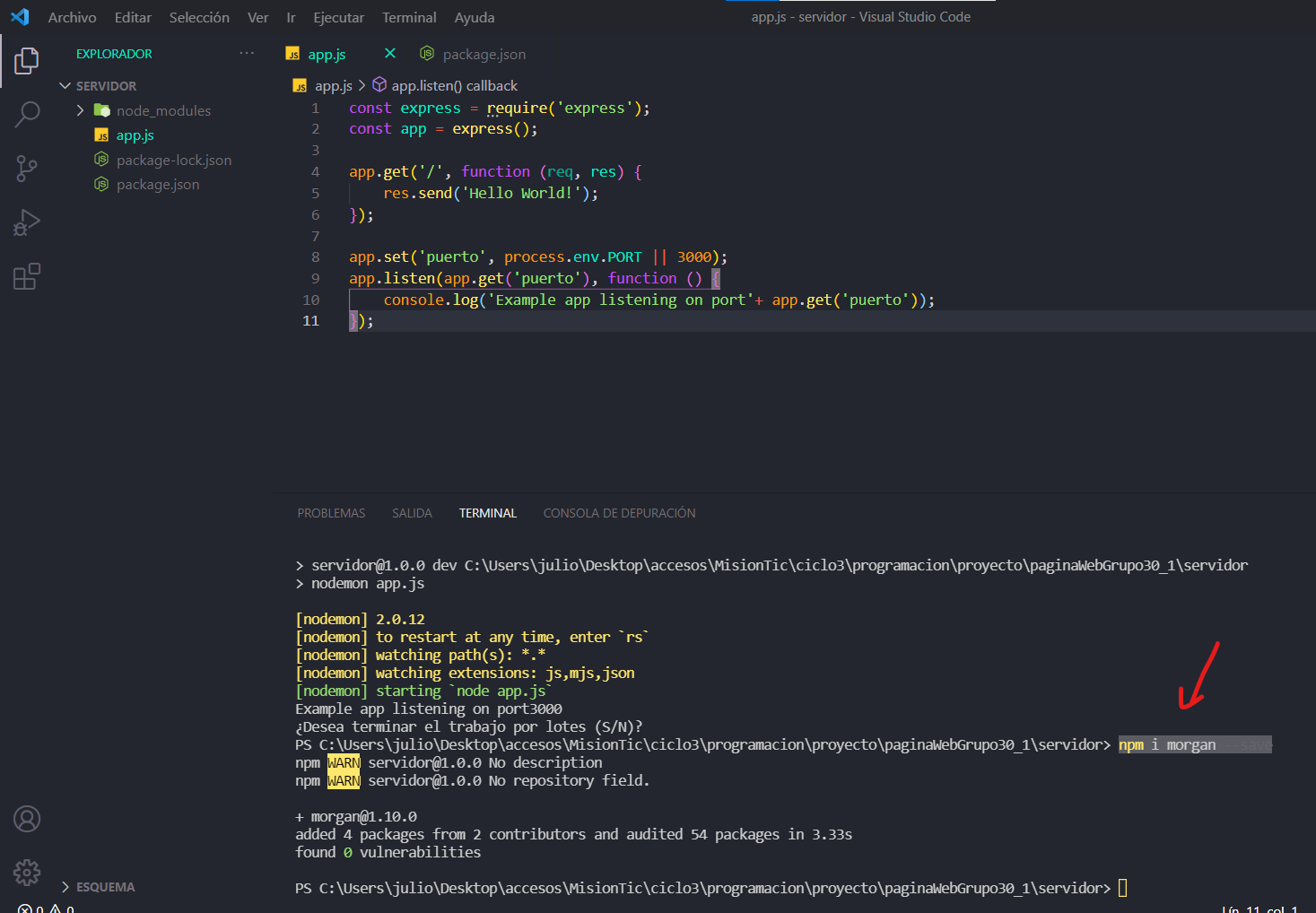
* Se agrega esta dependencia en el package.json y se da el comando npm run dev con el cual corre el servidor



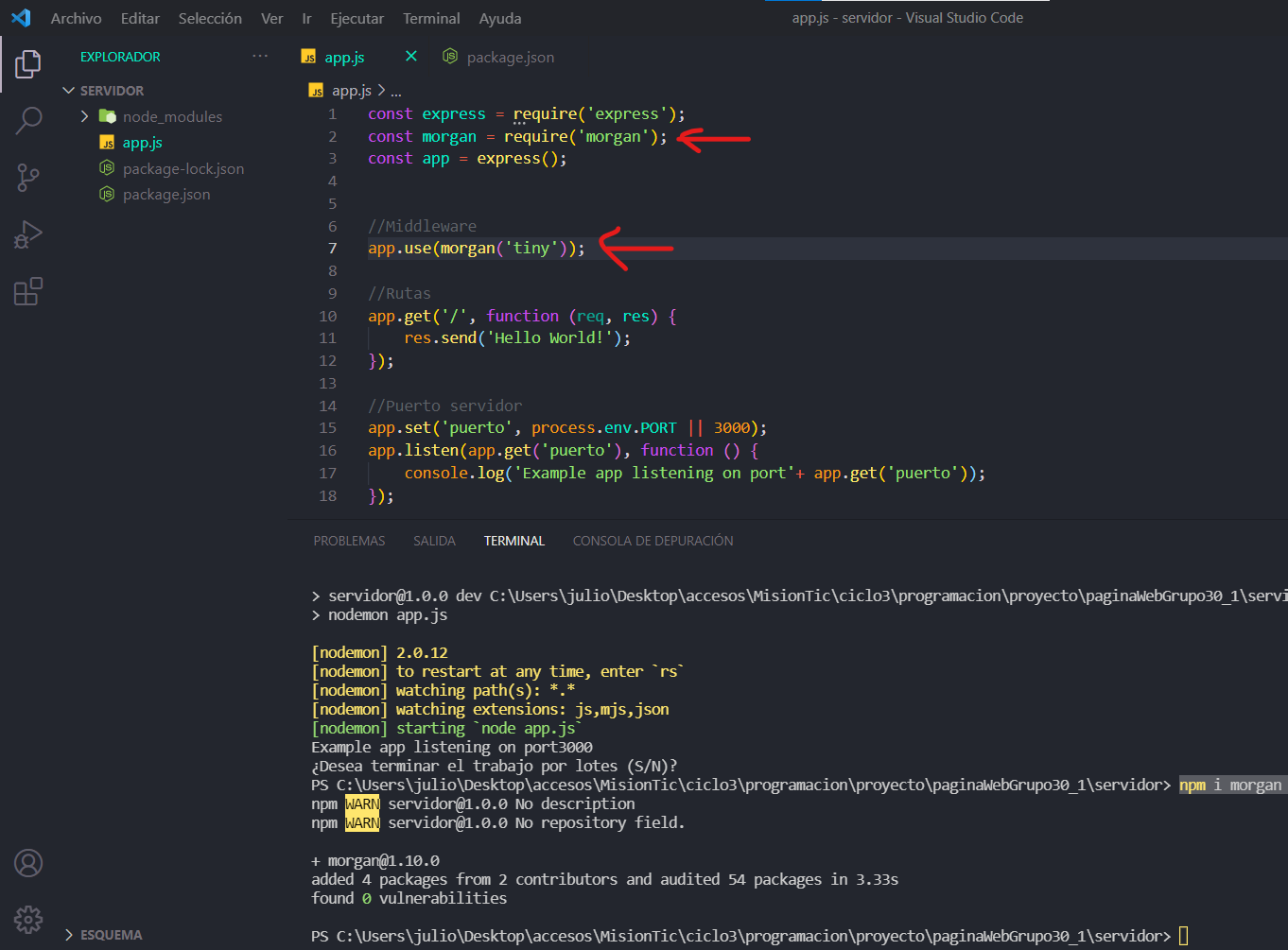
* Por otro lado, también se puede asignar el puerto de manera automática cambiando el código en app.js por el siguiente



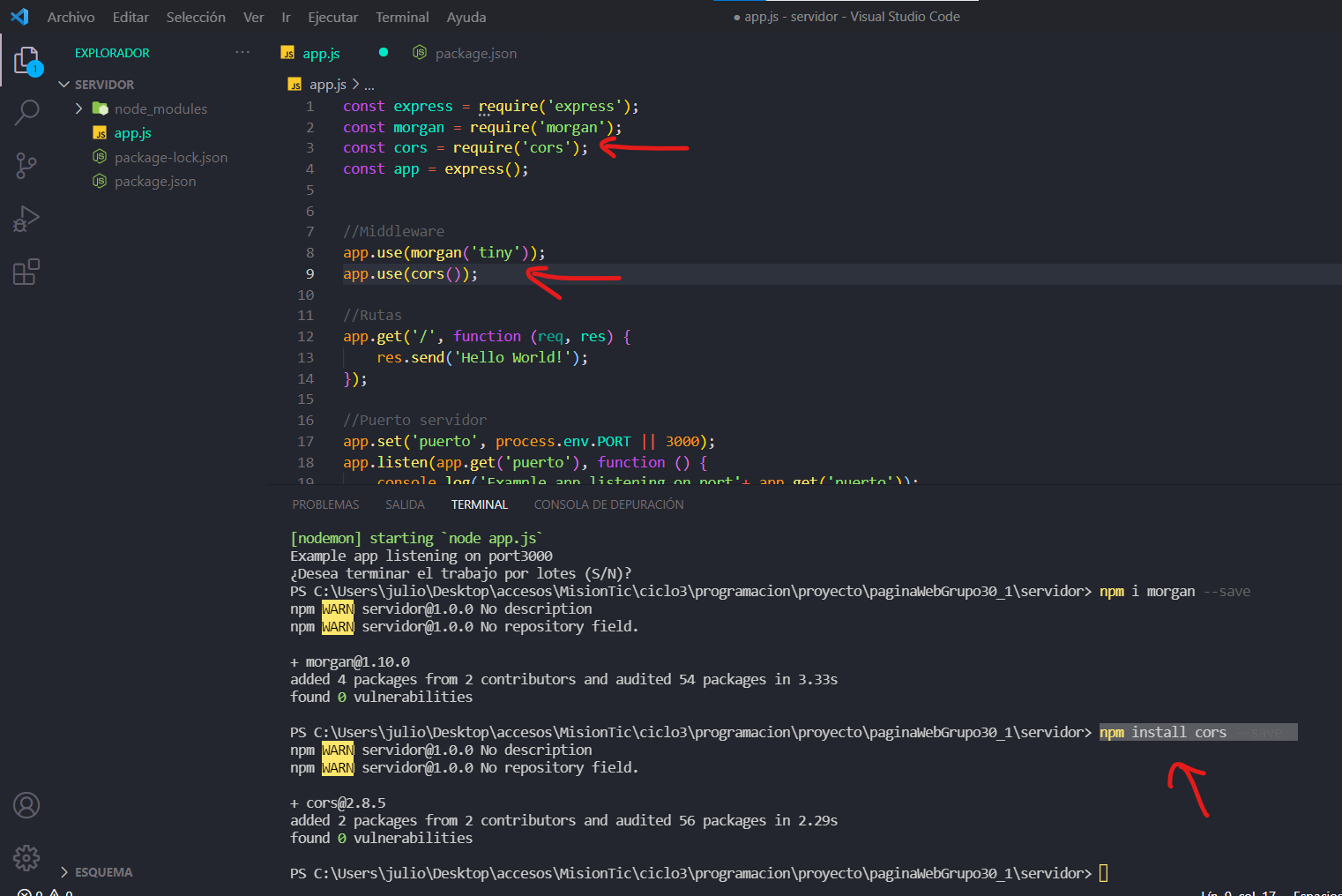
* Ahora se deben instalar algunos middlewares que permitan la conexión con base de datos y las peticiones http
* Primero instalamos Morgan que permite las peticiones http que se solicitan a la app



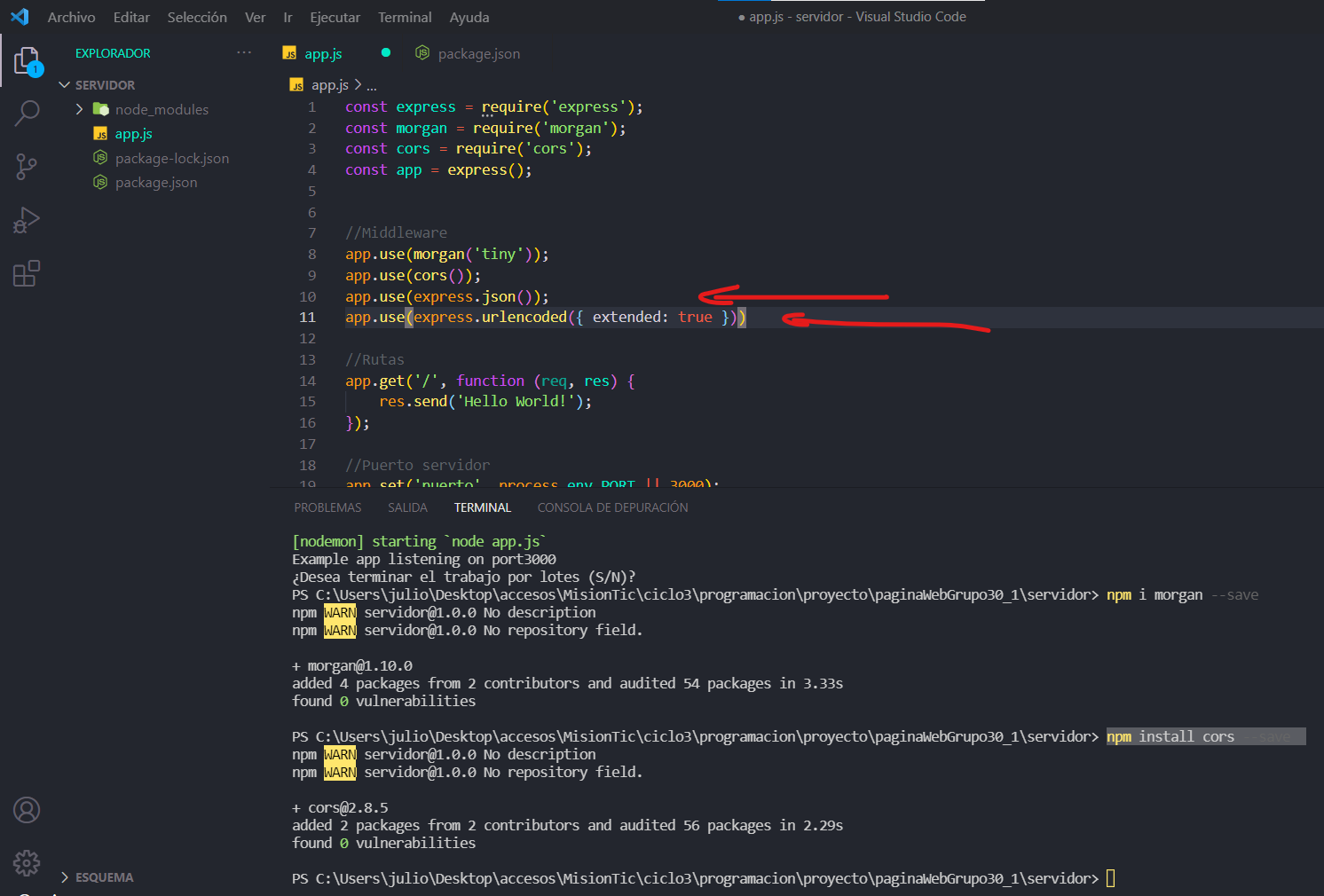
También se agregan a el documento app.js de la siguiente forma



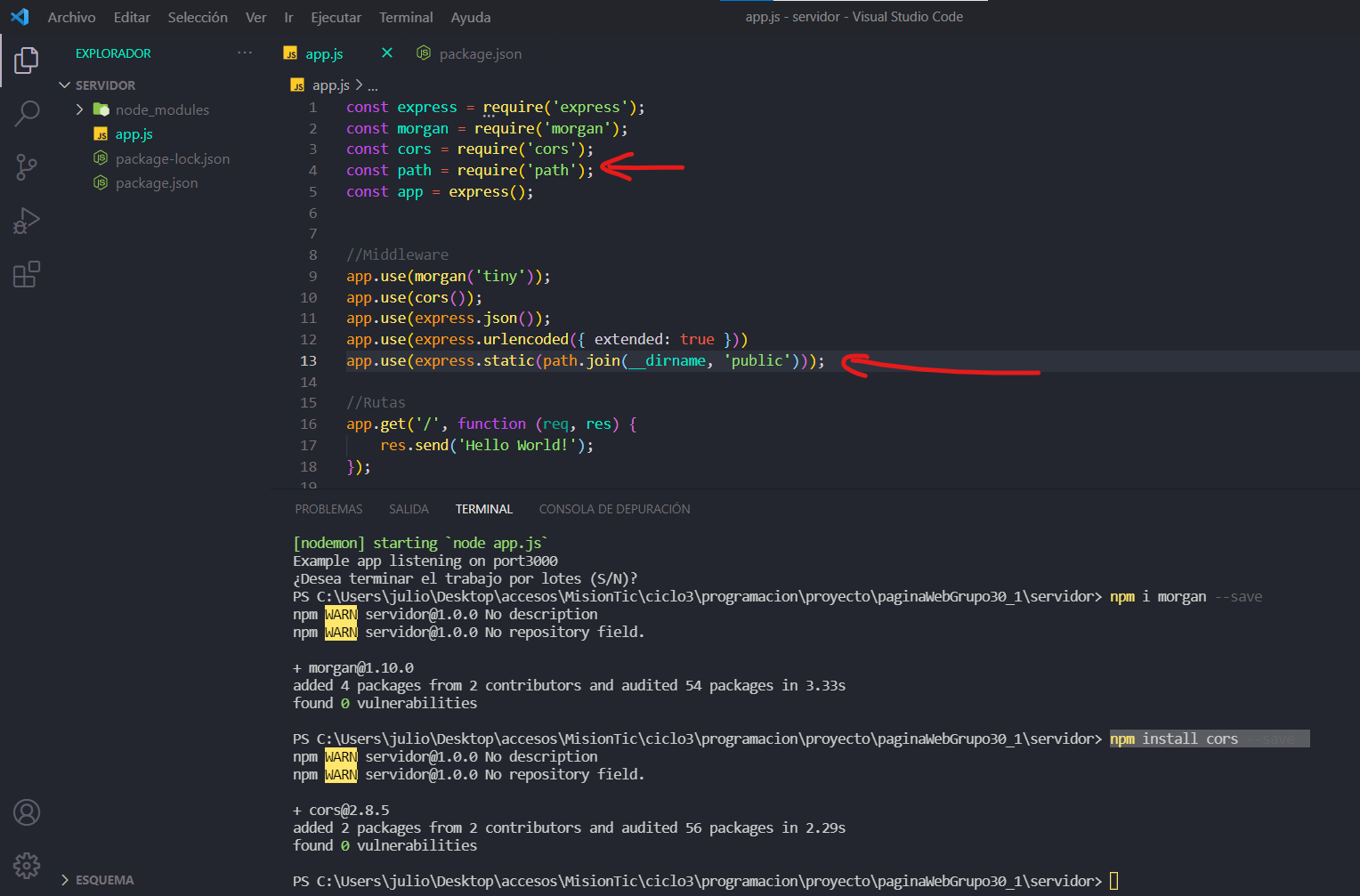
* Luego instalamos CORS que permite el acceso desde un servidor externo y que no haya bloqueos, el comando es npm install cors –sabe y se agrega y se usa la dependencia al archivo app.js



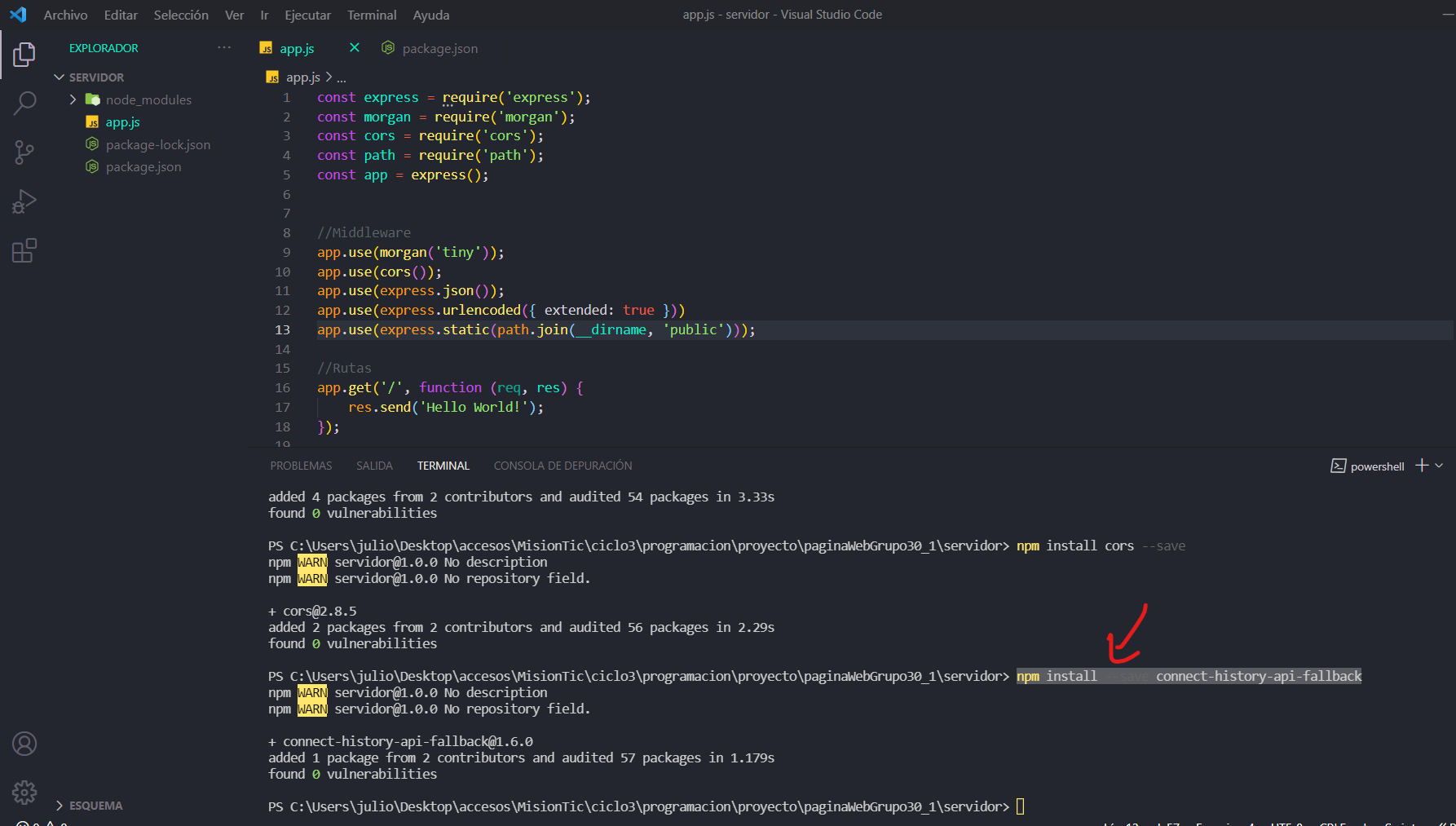
* Tambien se necesitan los middlewares para manejar json y los envios por formularios de tipo x-www-form-urlencoded, pero estos ya están incluidos en express por lo tanto solo deben ser llamados.



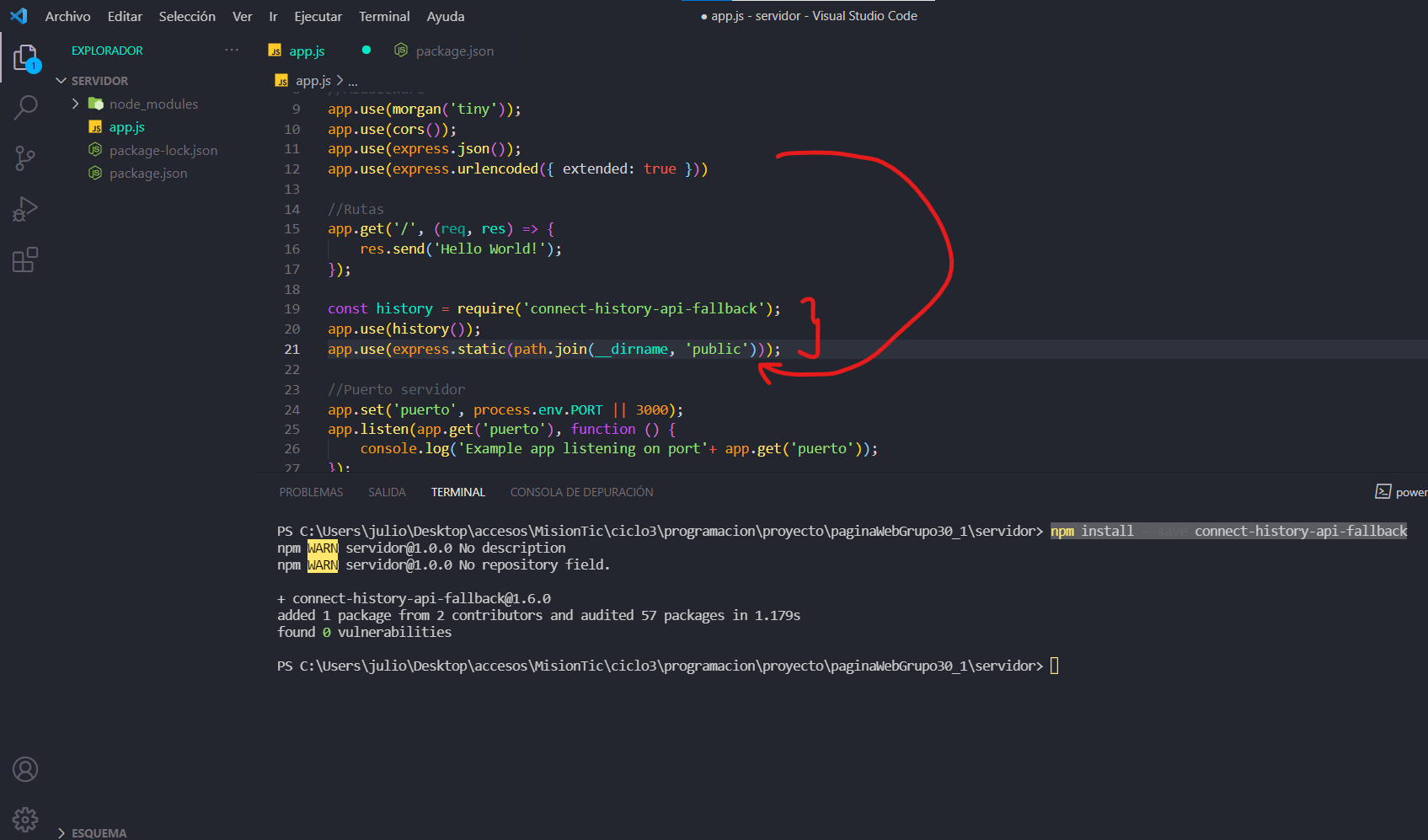
* Para poder acceder al path o directorio usamos otro middleware que no es necesario instalarlo, pero si debe ser incluido



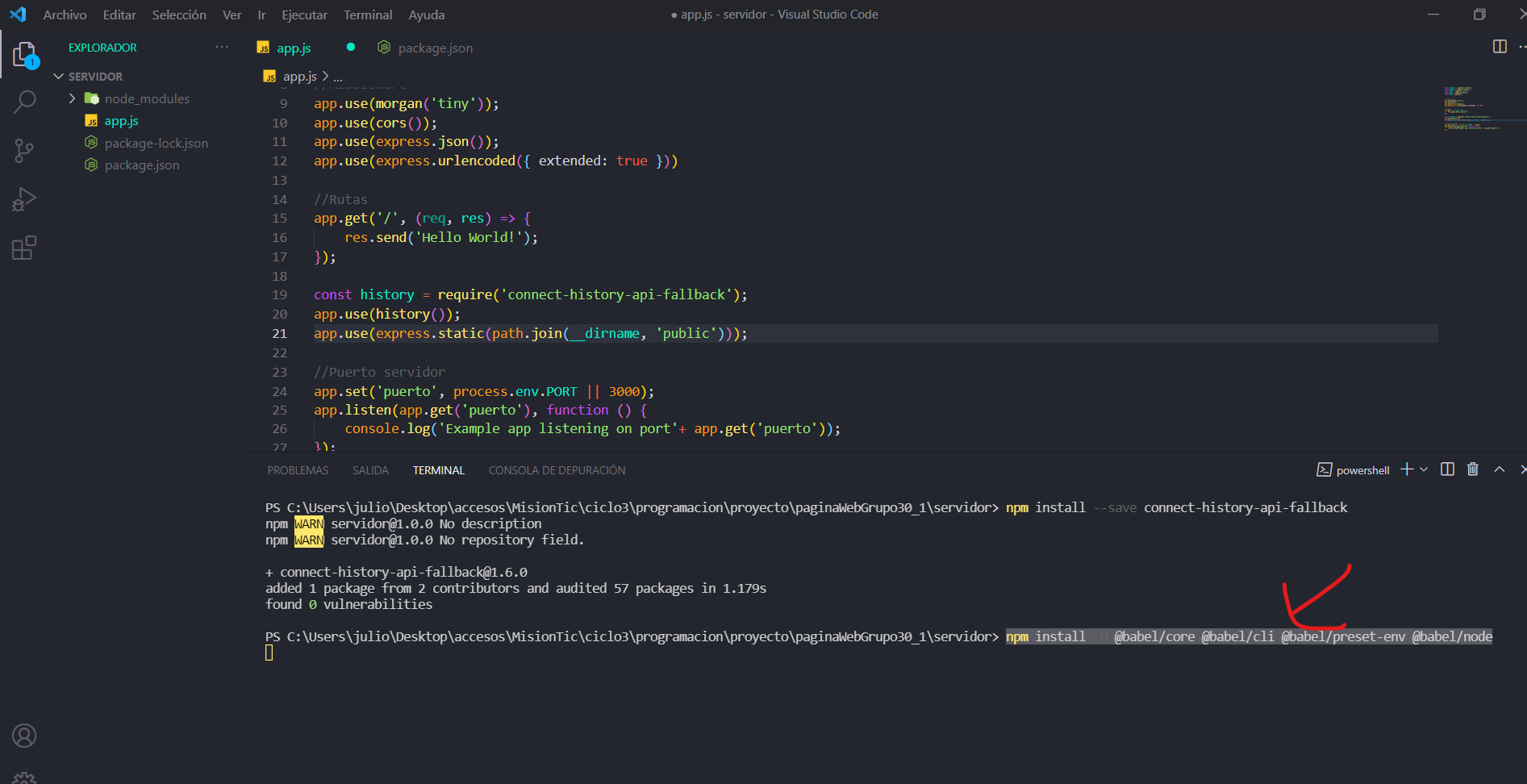
* Ahora para poder incluir las o simular las rutas de navegación del sitio web y que no haya problemas con los HTML agregamos otro middleware con el comando npm install --save connect-history-api-fallback



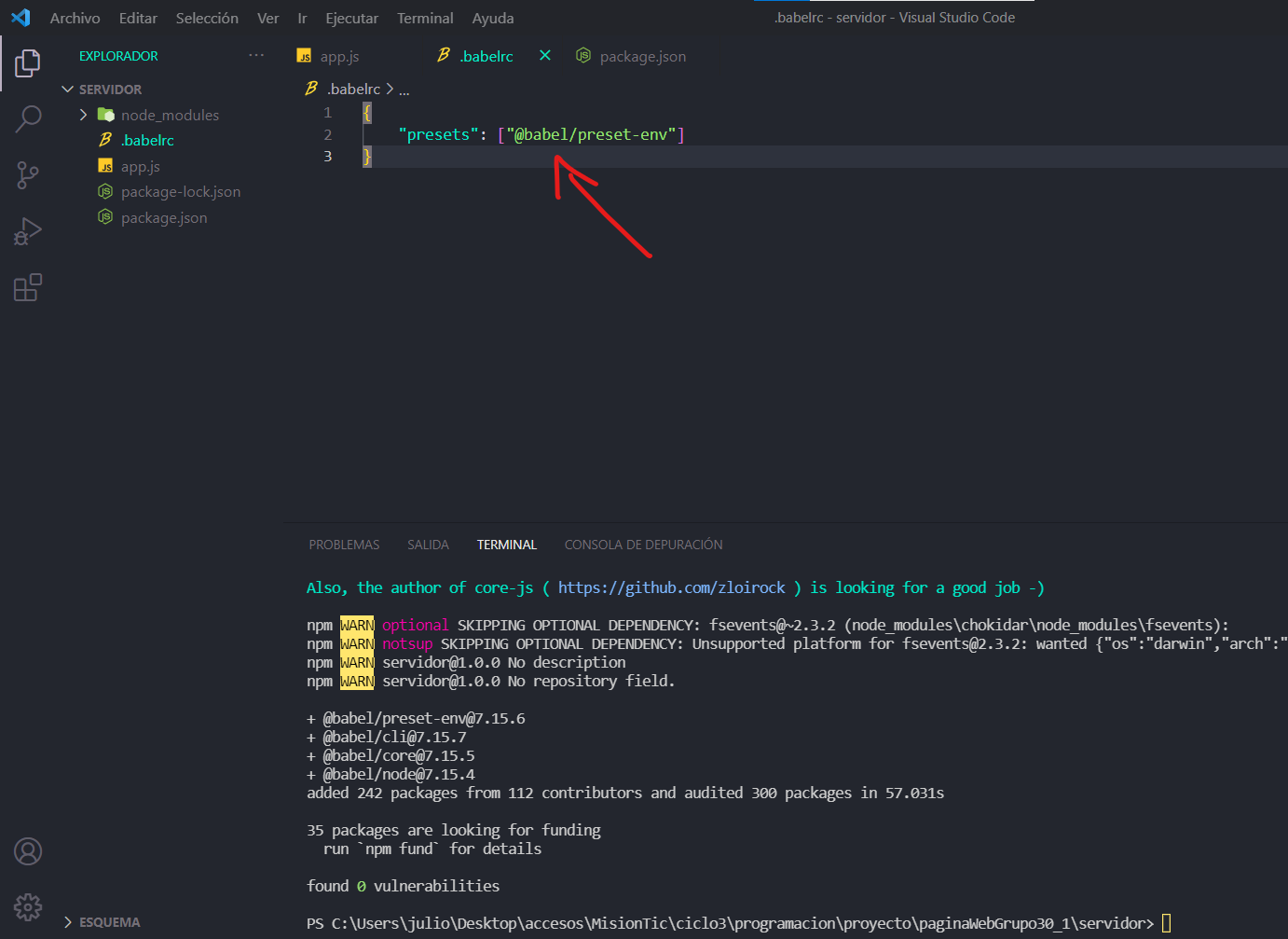
Luego se agrega el siguiente código y se debe dejar la configuración de las rutas estáticas debajo



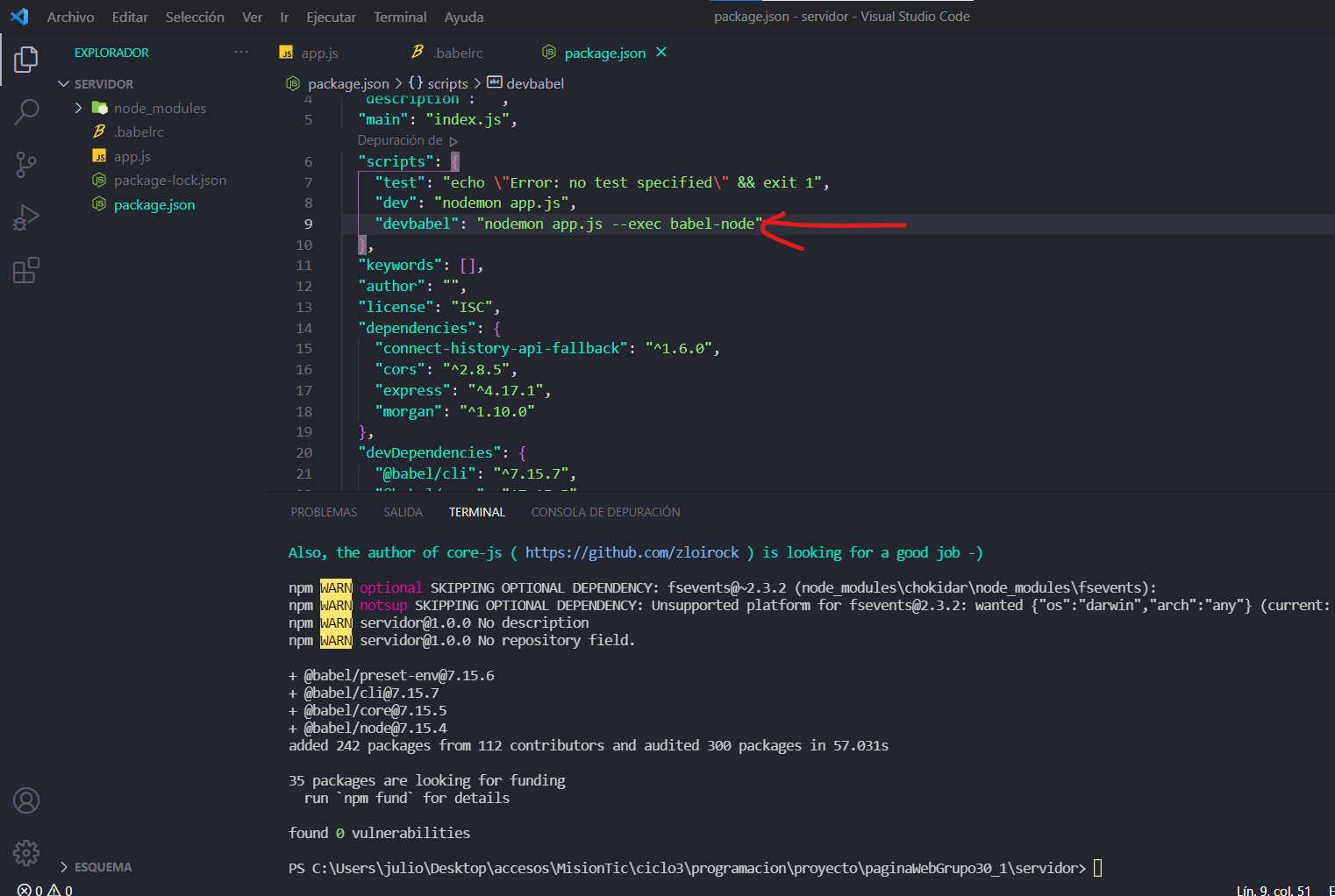
* Ahora se van hacer algunos cambios al proyecto ya que se instalará Babel el cual es un trascompilador de js que permite la compatibilidad con las diferentes versiones de js, para la instalación ingresamos el comando npm install -D @babel/core @babel/cli @babel/preset-env @babel/node



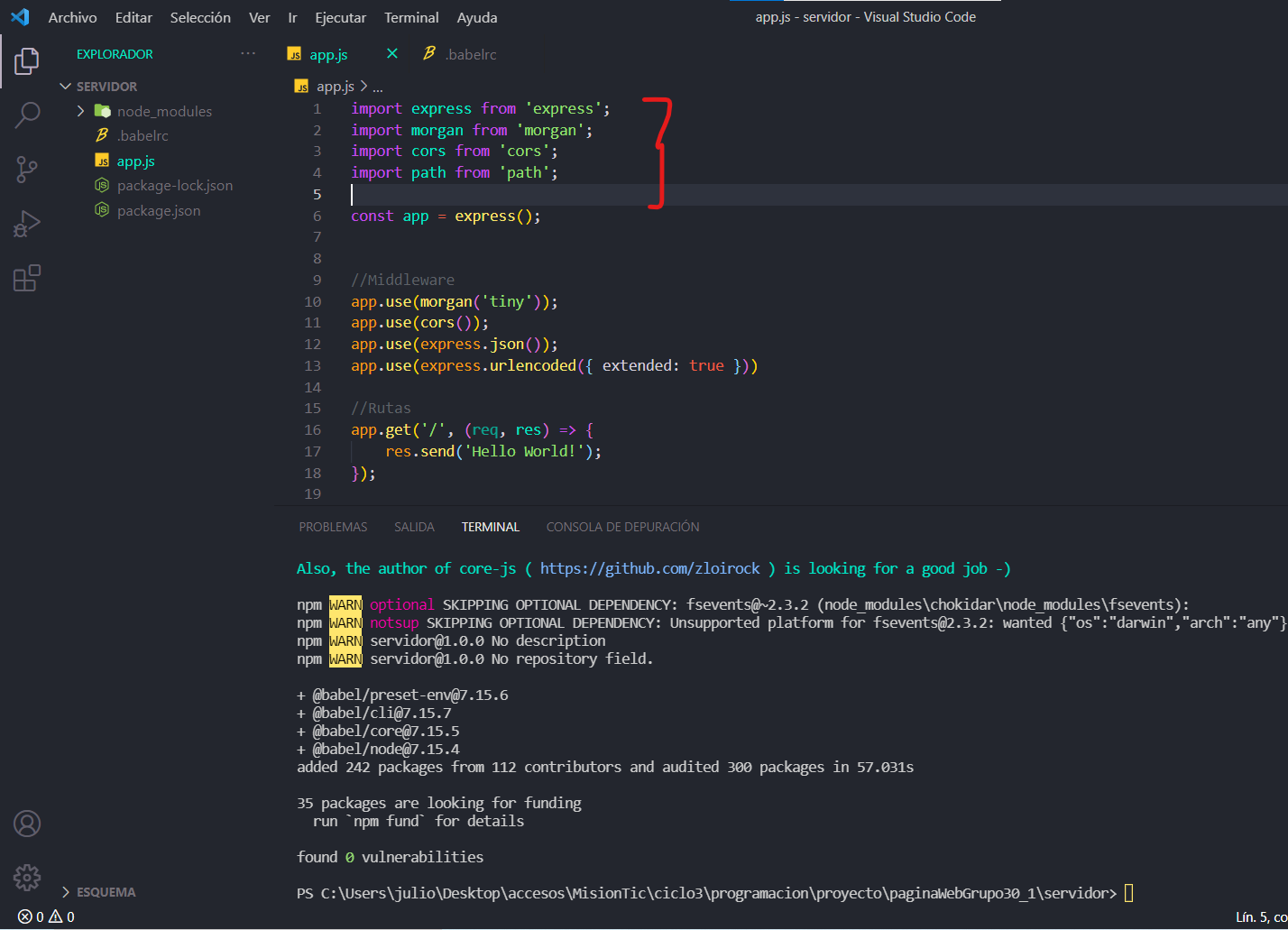
* Luego de la instalación de babel es necesario configurarlo, para lo cual se crea un archivo en la raíz llamada .babelrc y agregar lo siguiente



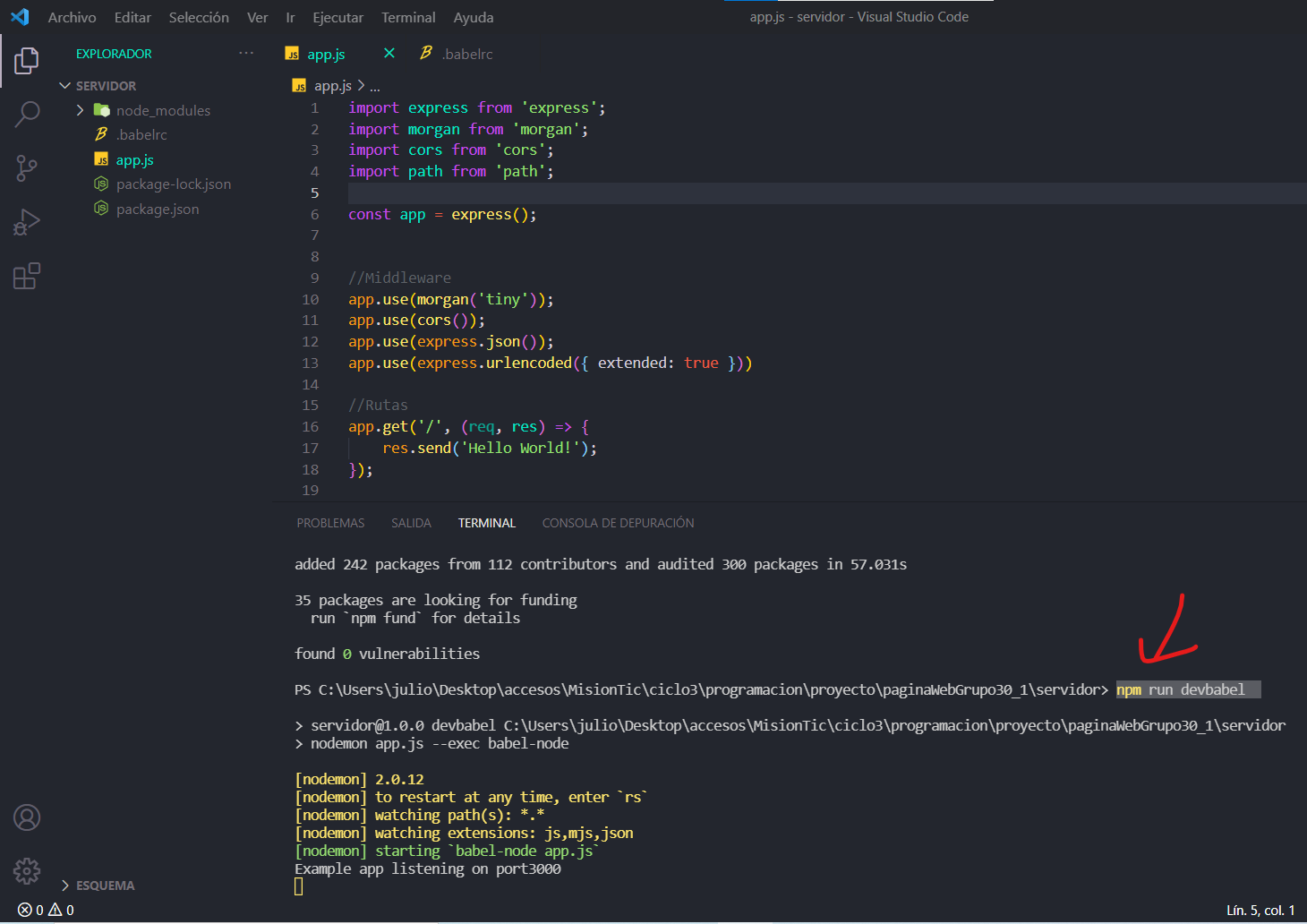
Luego se configura en el package.js en la sección de scripts



Después de esto el código pasa a ser ES6 por lo cual se pueden importar las dependencias de la siguiente manera

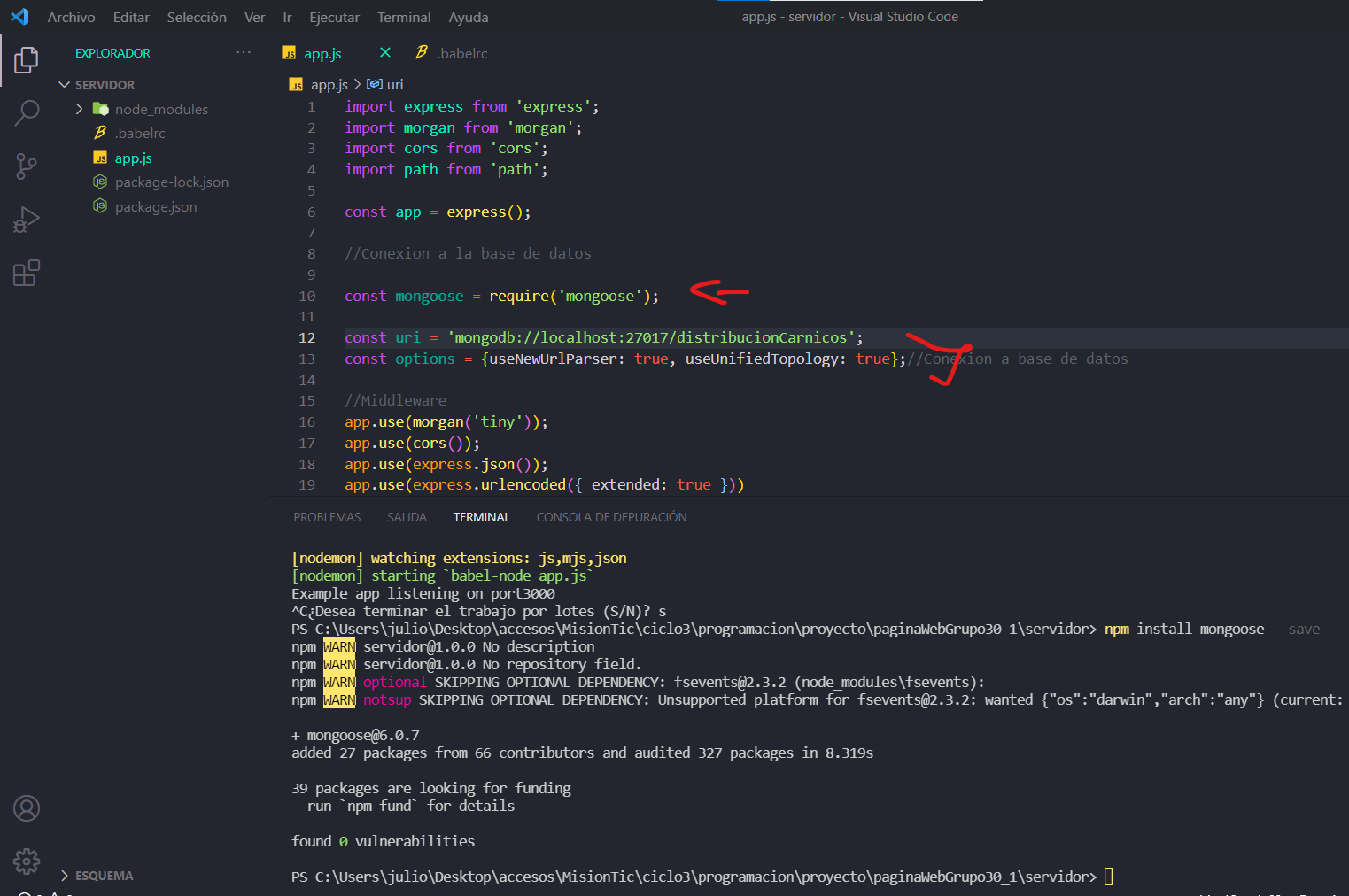


* Luego de todo lo anterior podemos comprobar si los cambios hechos son validos corriendo el servidor con npm run devbabel

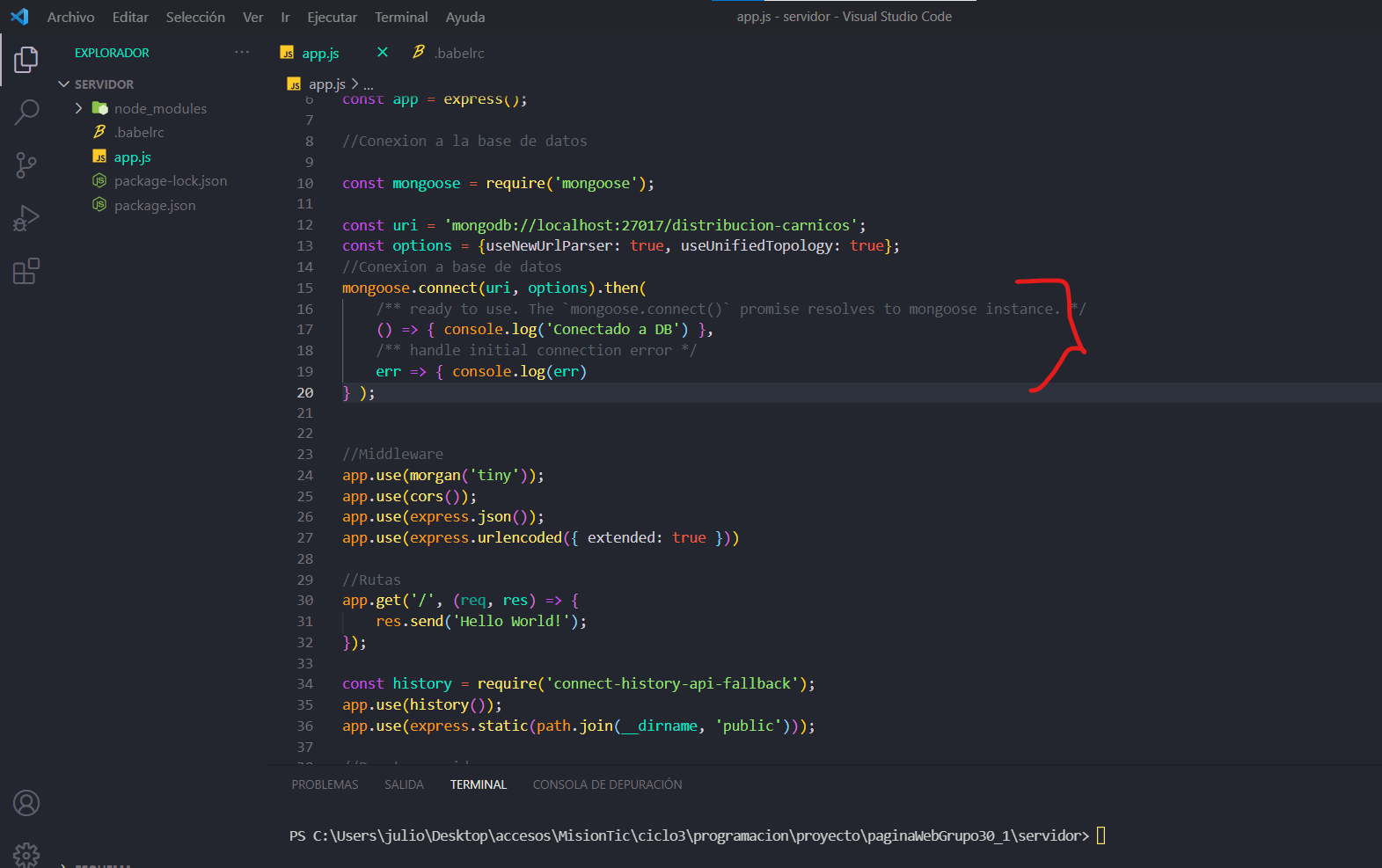


* Ahora con la creación del servidor es necesario añadir la conexión a la base de datos y las rutas para la creación del CRUD

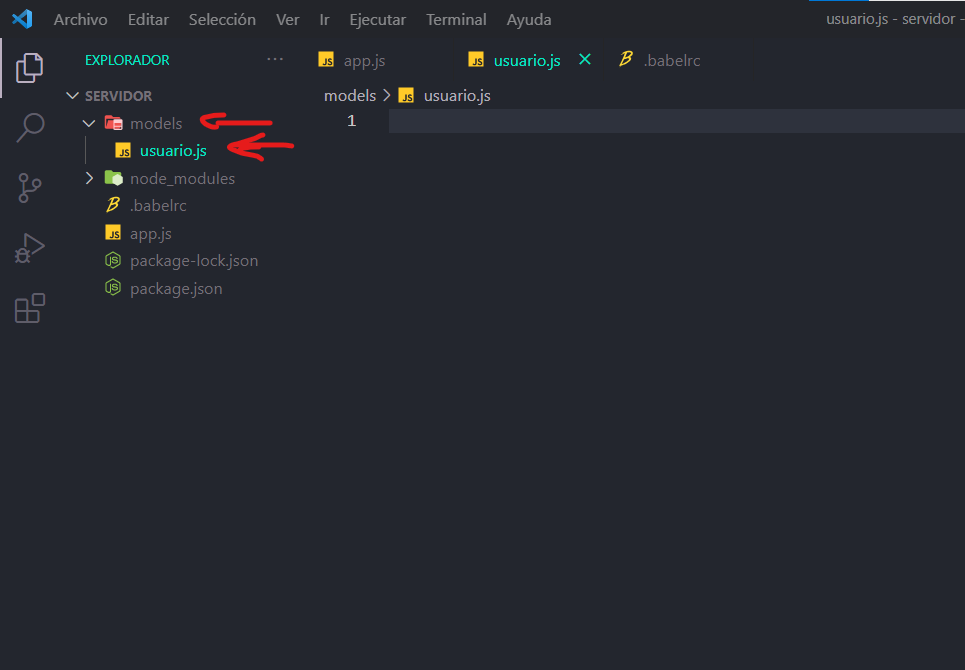
Para esto usamos mongoose y lo instalamos con el comando npm install mongoose –save, luego lo incluimos y hacemos la conexión con la base de datos como se muestra



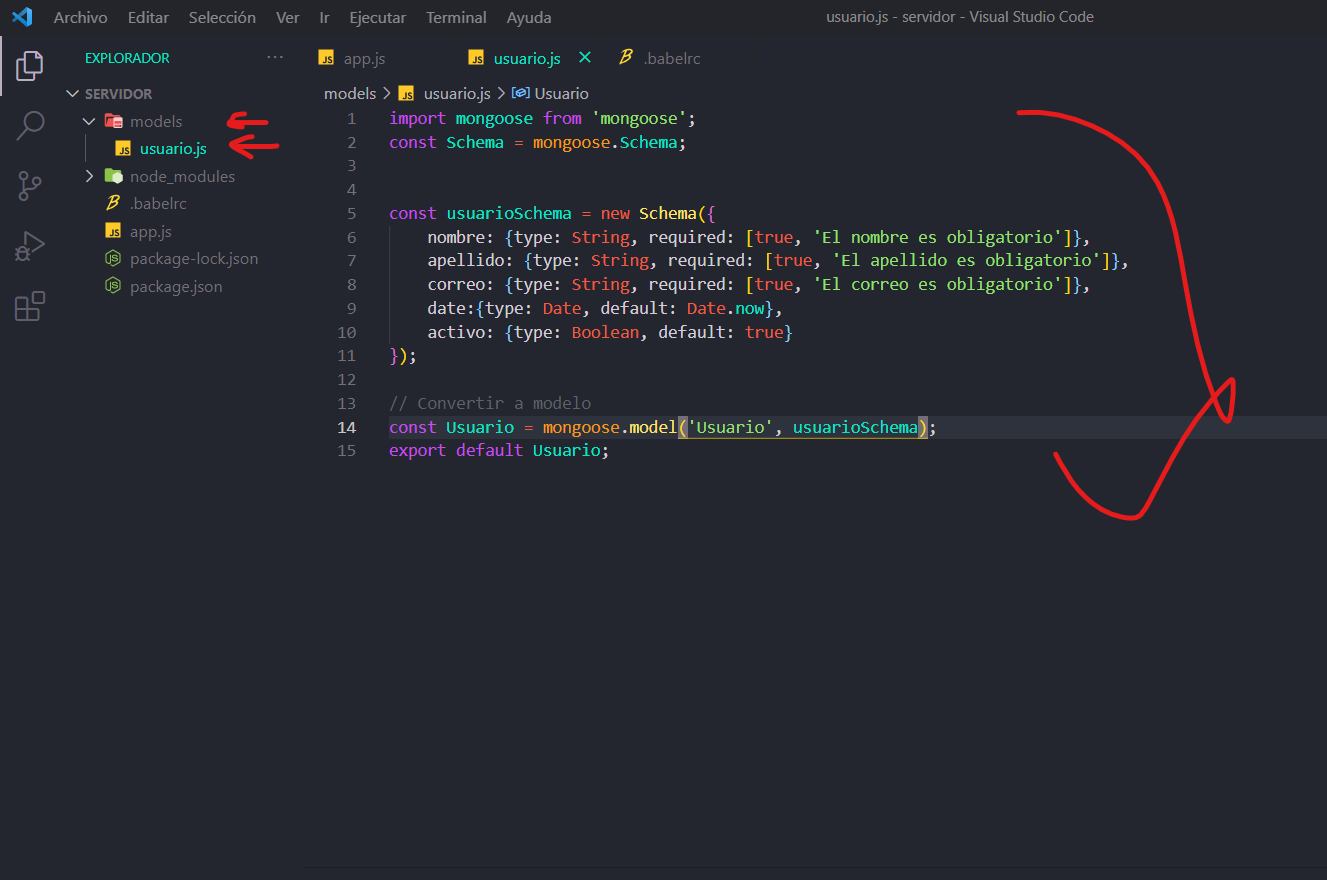
También se hace la conexión a través de una promesa



* Ahora es necesario crear los esquemas de la base de datos, es decir los elementos que irán en las colecciones, en este caso el modelo es para usuarios para esto creamos una carpeta llamada models y un archivo js llamado usuario.js

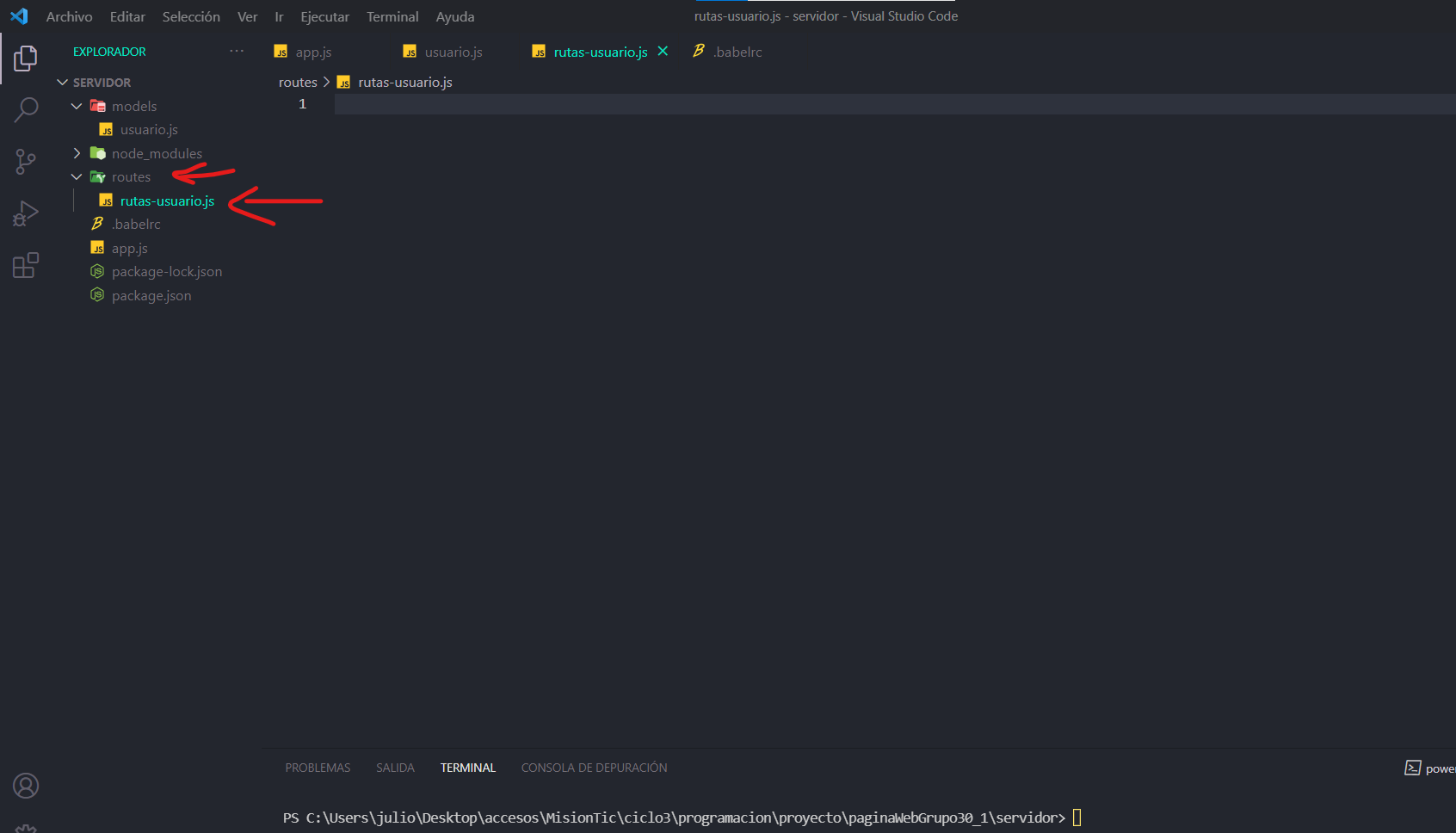


* Luego dentro de usuario.js se debe agregar moongose y se crea el esquema con los elementos que se manejaran de la siguiente forma

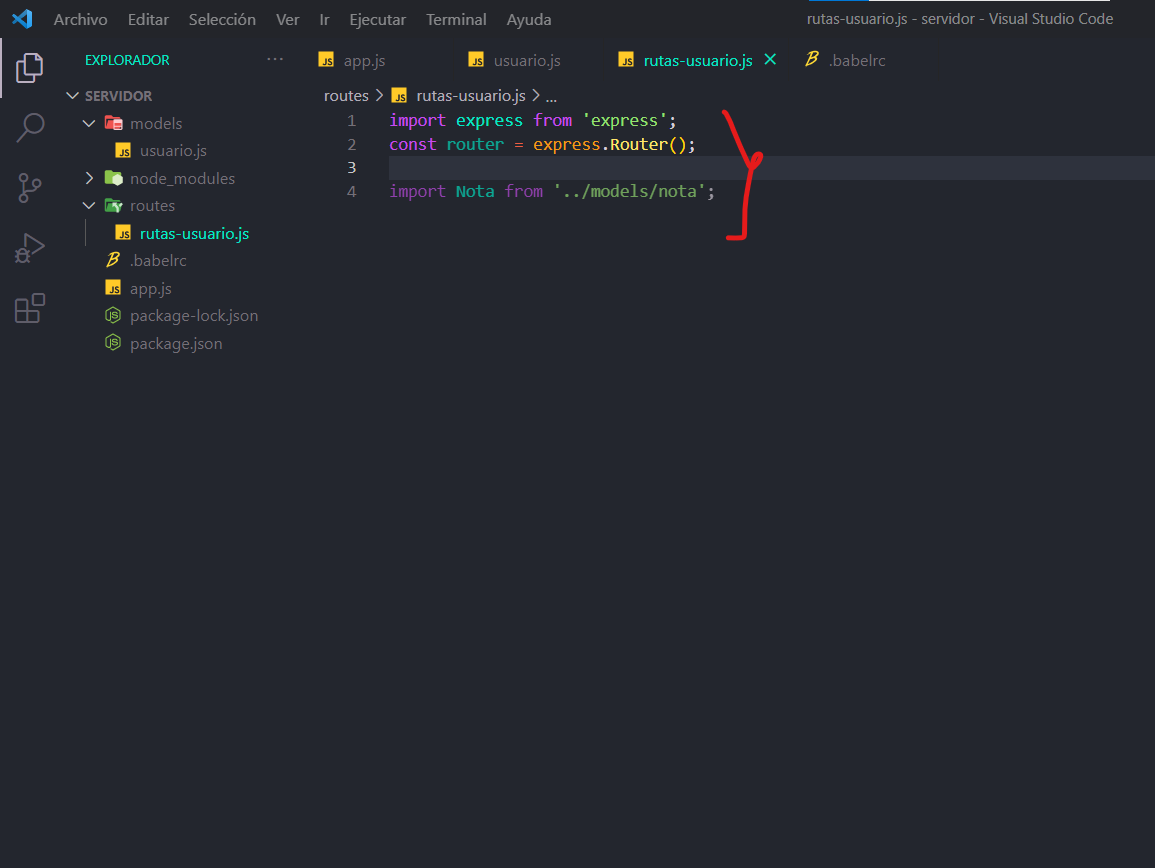


* Luego de la creación de el modelo es necesario crear las rutas por las cuales se accederá o se creará interacción con la base de datos.

Primero se debe crear una carpeta llamada routes y luego se agrega un archivo js con un nombre que haga referencia a el modelo como se muestra a continuación



Luego de la creación del archivo js es necesario usar express para el acceso a la base de datos y para indicar que este archivo tendrá las rutas que se usaran para hacer los cambios en los datos. Para esto se debe agregar el siguiente código

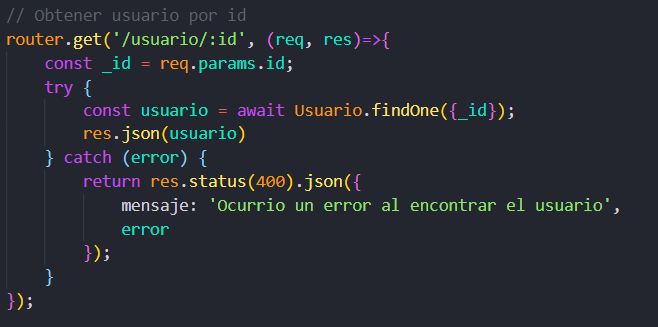


Luego es necesario crear las diferentes funciones para el CRUD con el nombre de sus respectivas rutas de la siguiente manera.

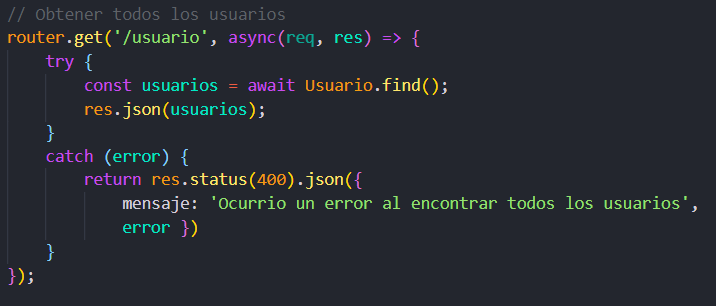
Para agregar nuevo usuario



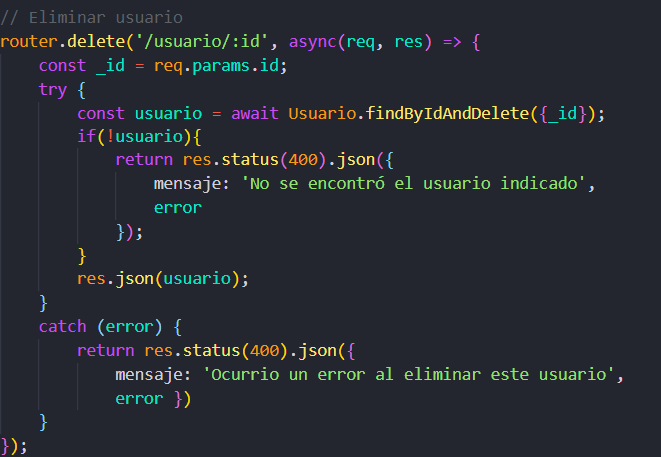
Encontrar usuario por id



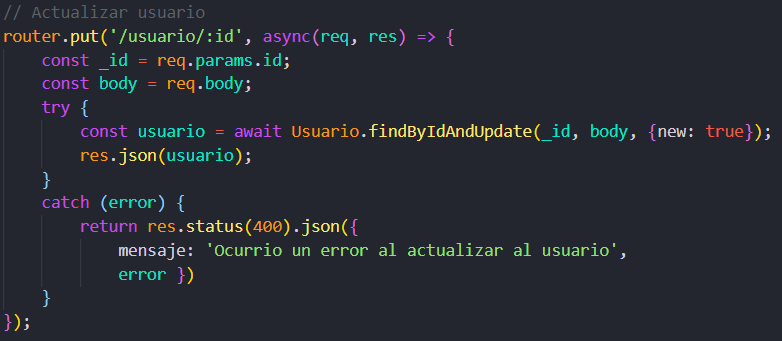
Encontrar todos los usuarios



Eliminar usuario



Actualizar usuario



* Por último, es necesario exportar estas rutas indicadas con express para lo cual se agrega esta línea al final del código

