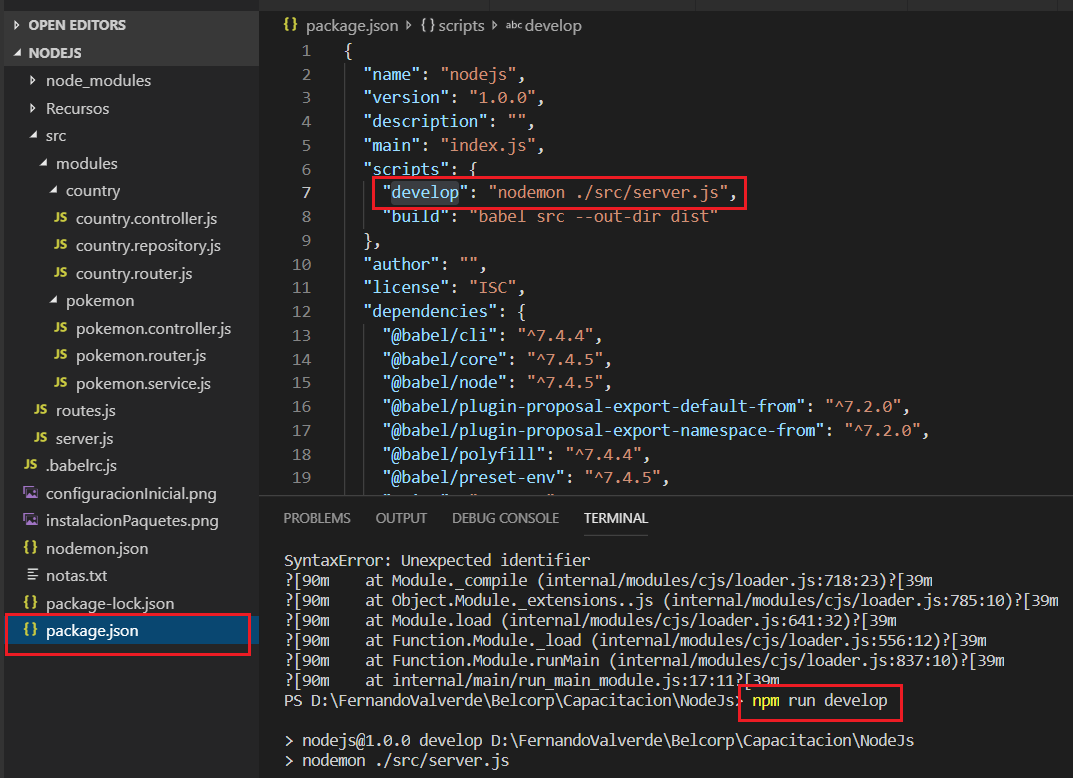
**Ejemplo práctico de servicios con NodeJs**

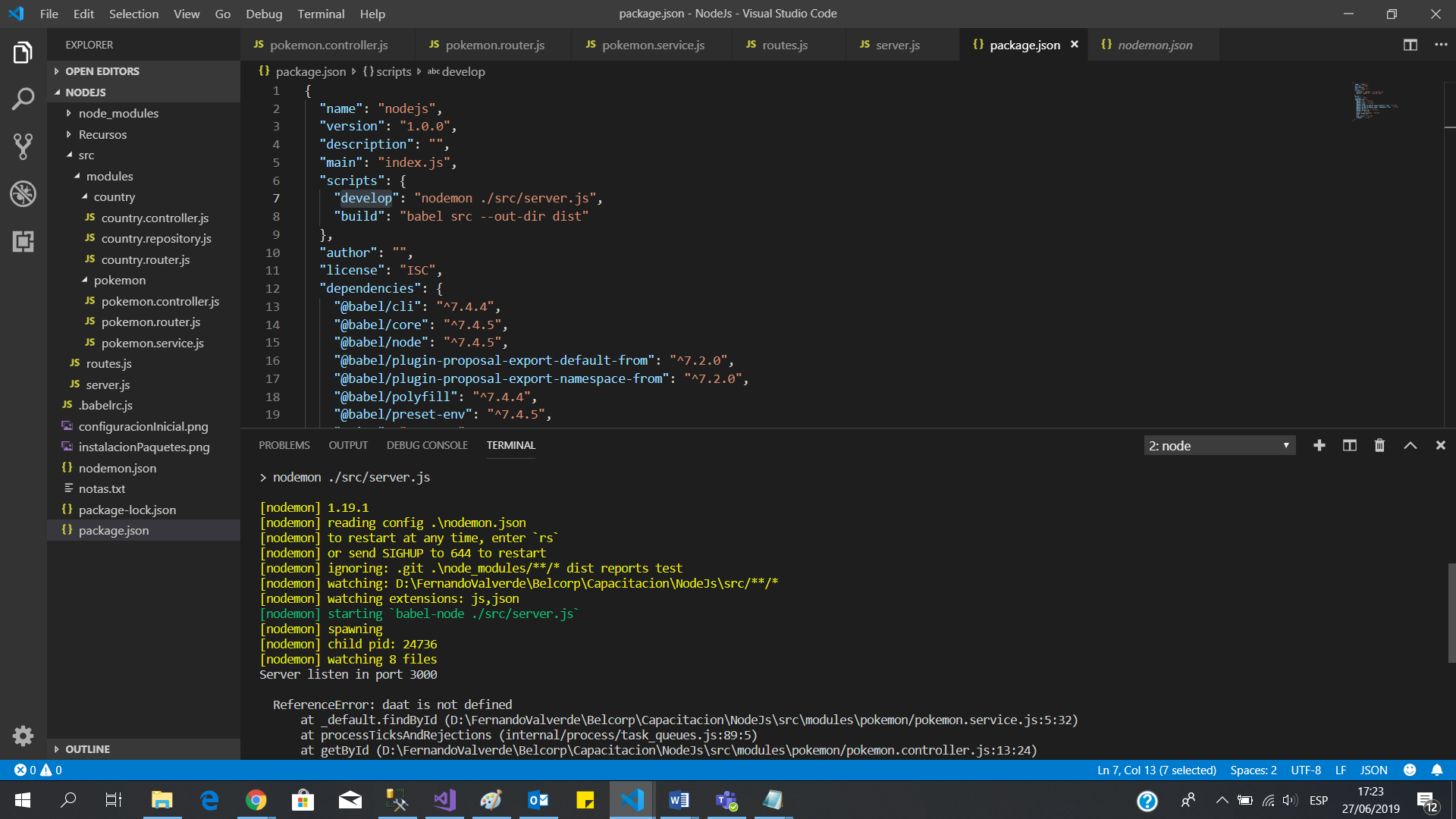


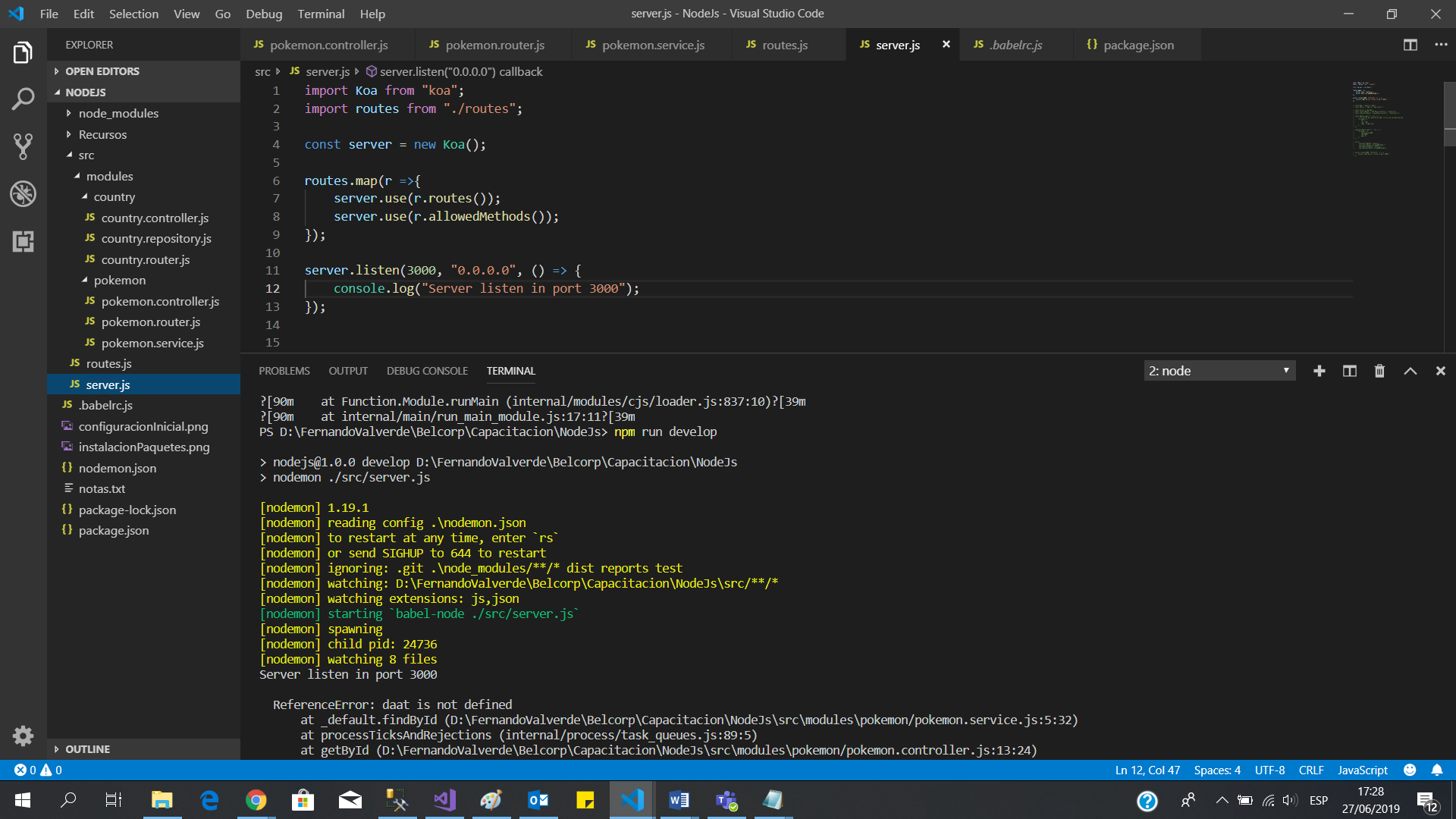
nodemon permite ejecutar comandos de distintas versiones de node.

El servidor se ejecutaría con el comando node ./src/server.js pero al tener configurado nodemon podemos guardar esa ruta en la variable develop y ejecutarlo mediante el comando:

**npm run** develop

El servicio inicia (se ejecuta archivo server.js) y muestra el mensaje configurado.



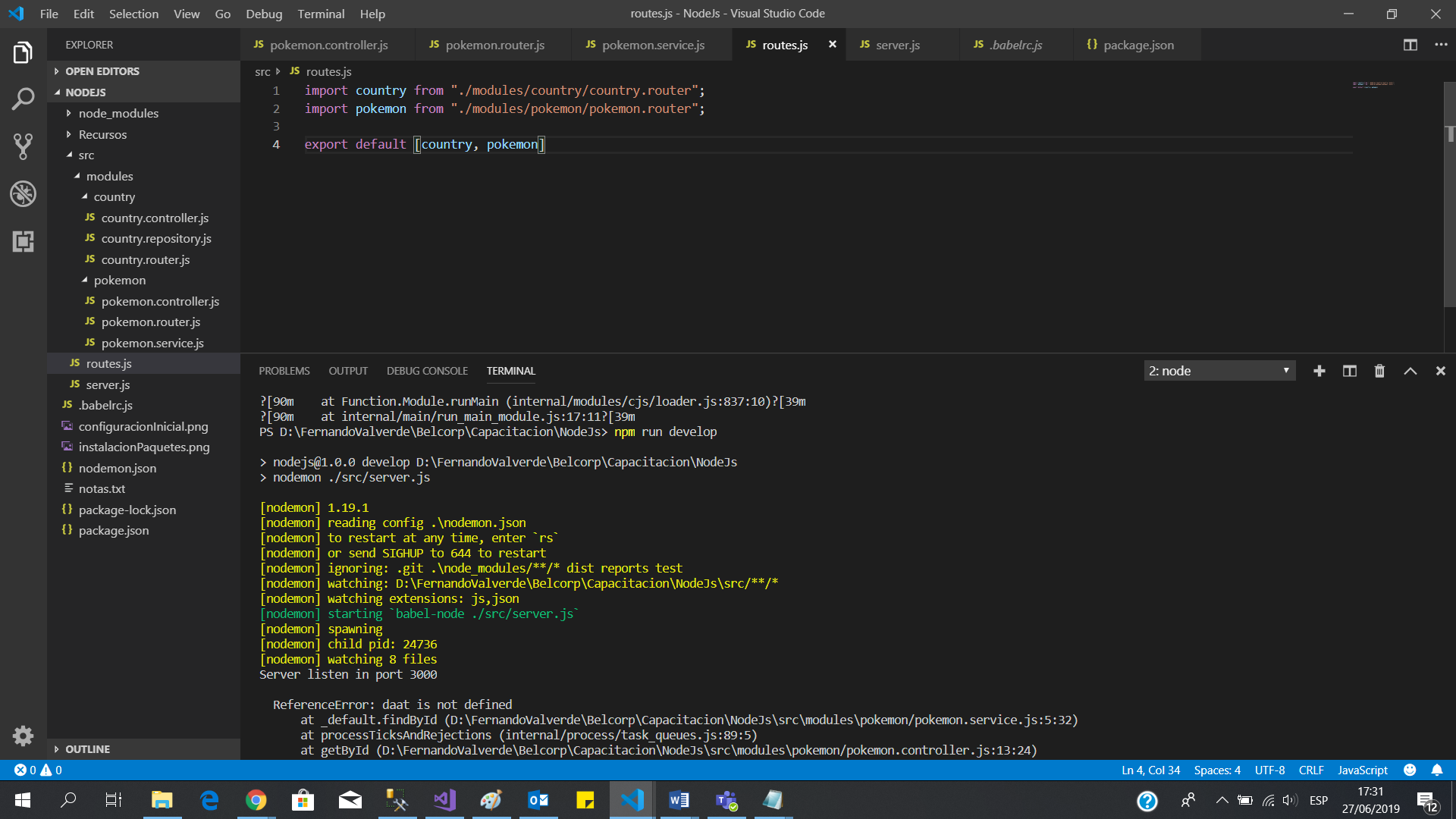
**Service.js**

Importa la configuración del archivo routes.js

Todas las rutas, todos los métodos.

Inicia el servicio en el localhost (realmente un broadcast) puerto 3000 mostrando un mensaje.

**routes.js**



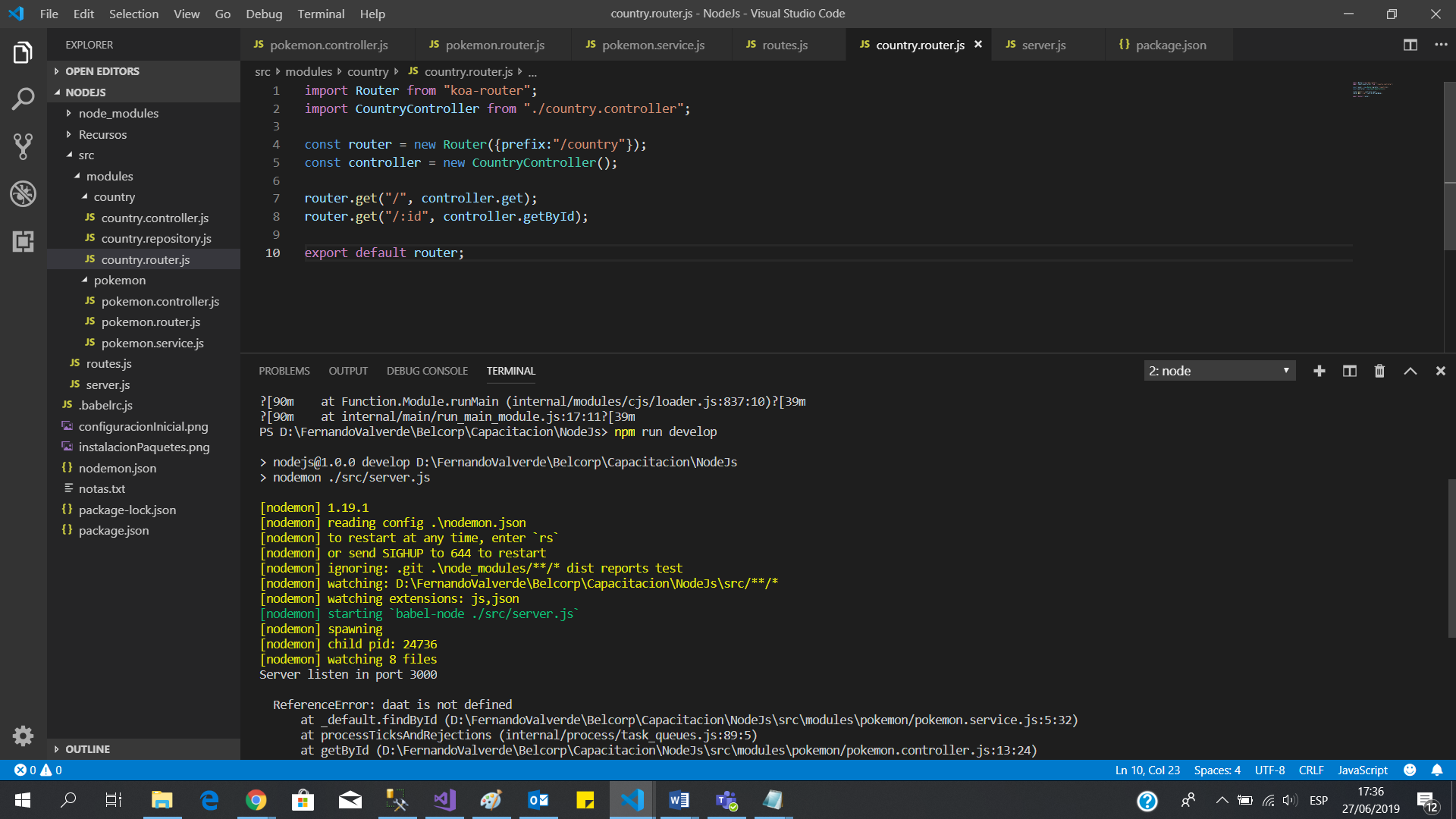
Importa la configuración de diferentes archivos. Son rutas para acceder a algún servicio. En este ejemplo, **country.router** simula un resultado de un servicio y **pokemon.router** llama a un servicio externo que provee datos de los pokemones.

service.js accede a estas rutas al importar routes.js

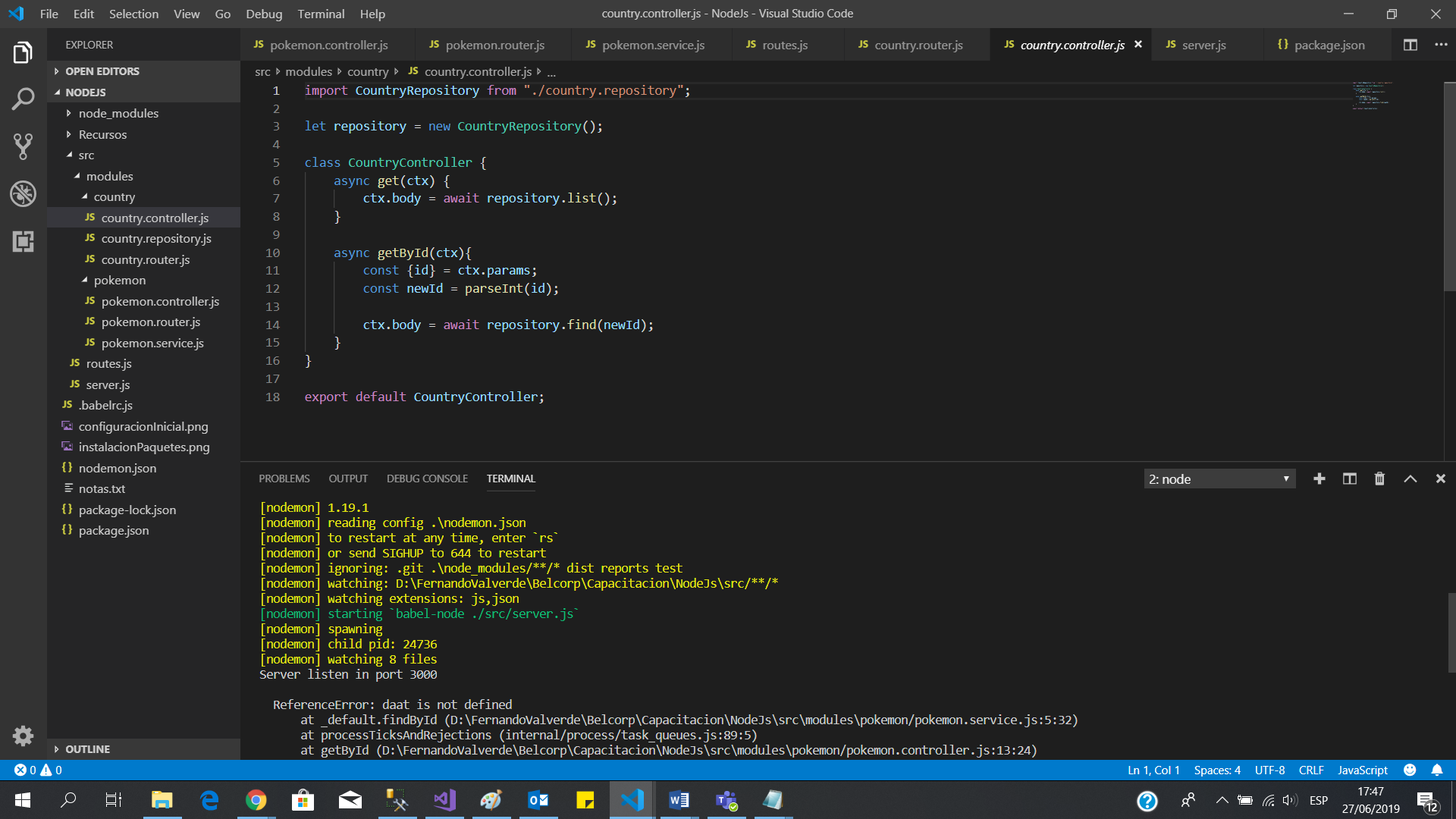
**Ejemplo Servicio Country**

Se simula el llamado a un servicio recuperando la data en duro desde country.repository.js

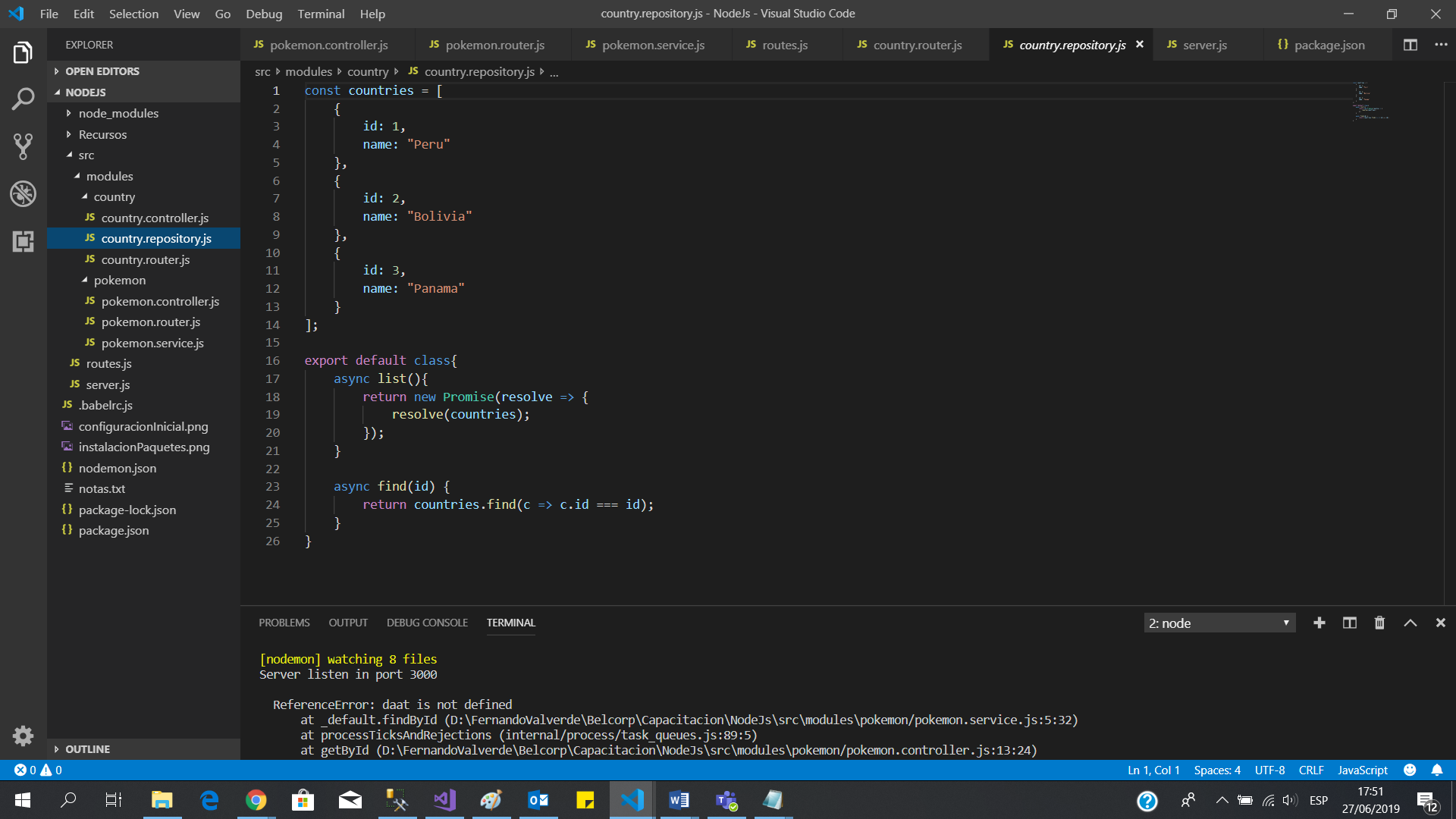
**country.router.js**

Se importa la configuración del **country.controller**, se define el prefijo para esta nueva ruta **/country** y las formas de acceder a la información mediante el método GET: **/** todos los elementos, **/:id** elemento por id.

**country.controller.js**



Importa la configuración del repositorio **country.repository.js** y define los métodos asíncronos para listar o buscar objetos.

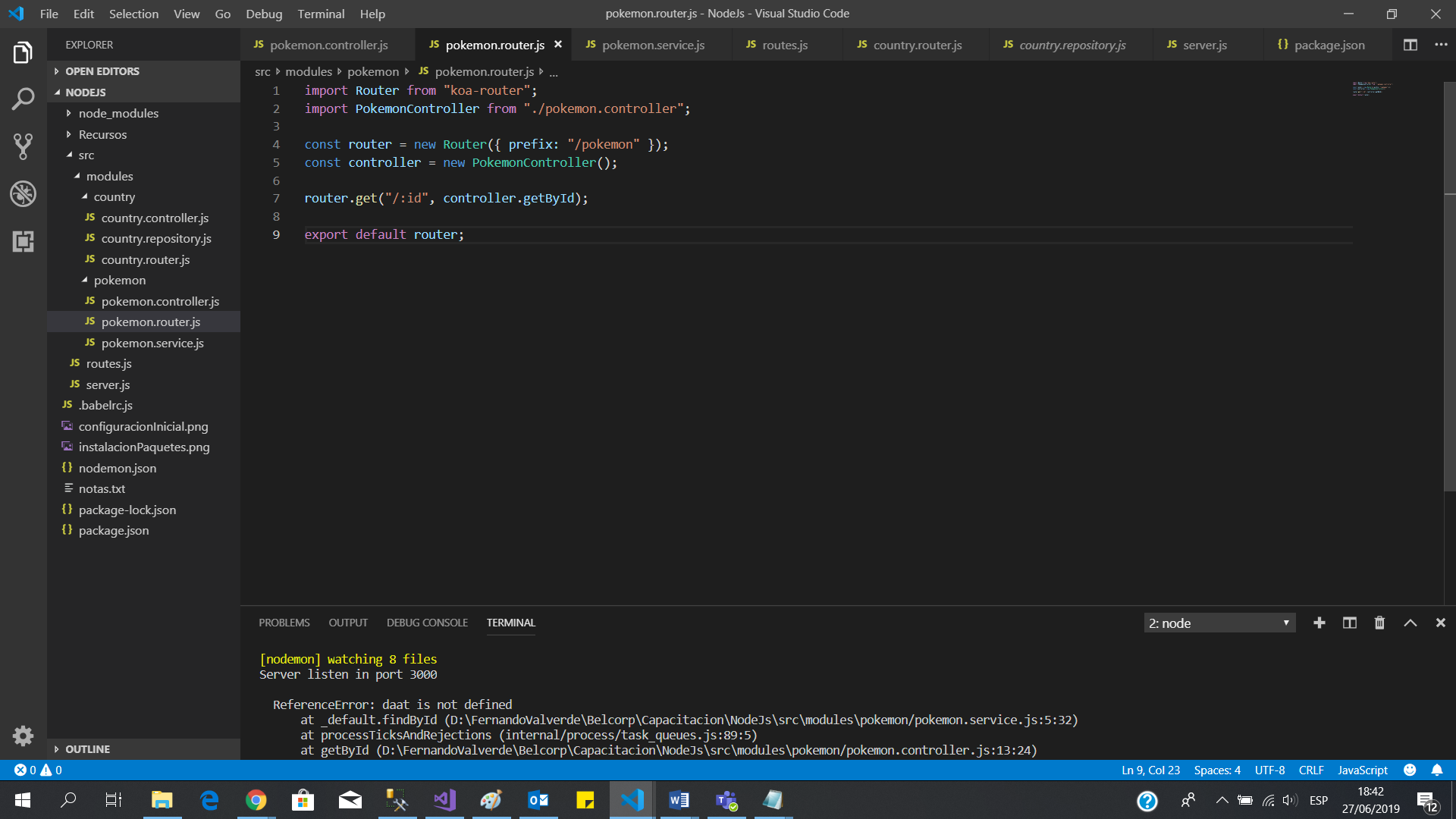
**country.repository.js**

se crea el arreglo de objetos **countries** que contiene los datos en duro de tres objetos para simular un servicio que devuelve información.

Exporta mediante una clase default el método para hacer el listado y el método para hacer la búsqueda por id.

**Ejemplo Servicio Pokemon**

Se hace la consulta a un servicio web real, recuperando la data en formato json.

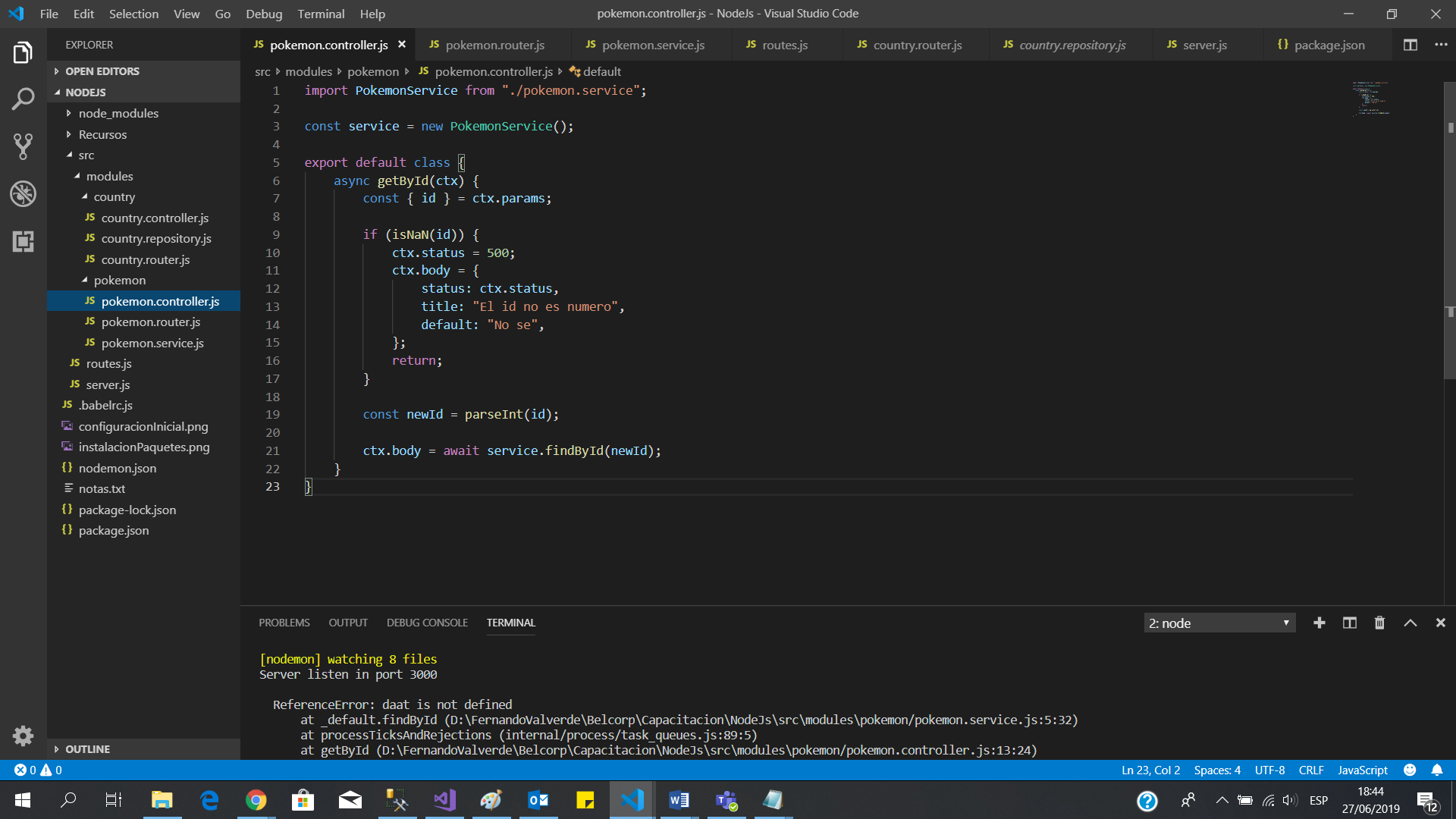
**pokemon.router.js**

Solo se recupera data por id.

Se importan los datos de controler.

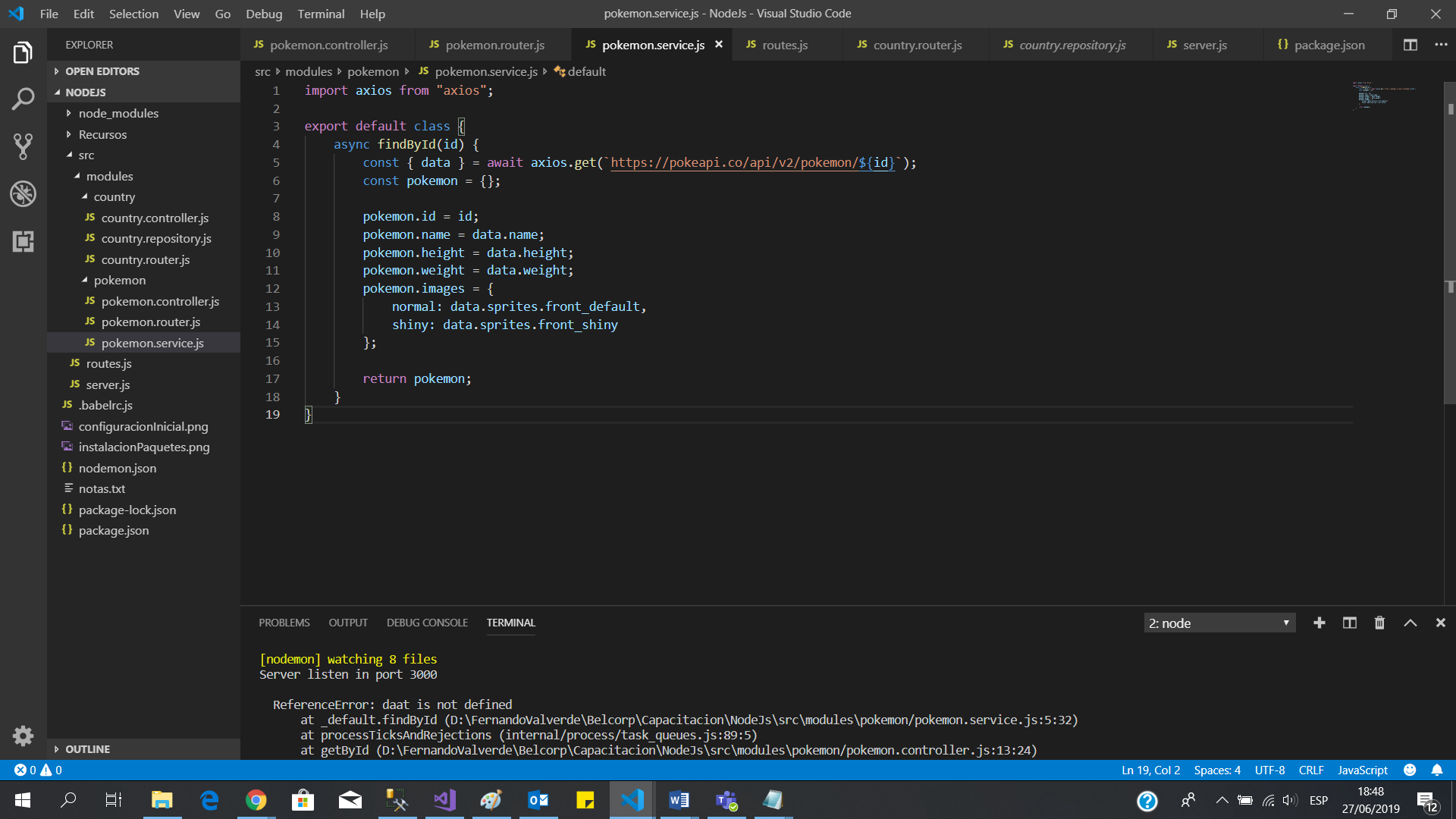
La ruta sería /pokemon/[id]

**pokemon.controller.js**

se importa la configuración de **pokemon.service.js**

se configura el contexto para recibir como parámetro el id. Asimismo, se valida que este id sea un número y en caso contrario se devuelve un estado de error y un mensaje.

**pokemon.service.js**



A diferencia del Country, ahora no se usa un repositorio puesto que la data se va a obtener desde fuera a través de la URL <https://pokeapi.co/api/v2/pokemon/>

En esta URL se incluye el id. Los datos obtenidos se recuperan en la variable **data** y se crea un objeto de tipo pokemon al cual se le setean sus propiedades con los valores recuperados para luego retornarlo.

