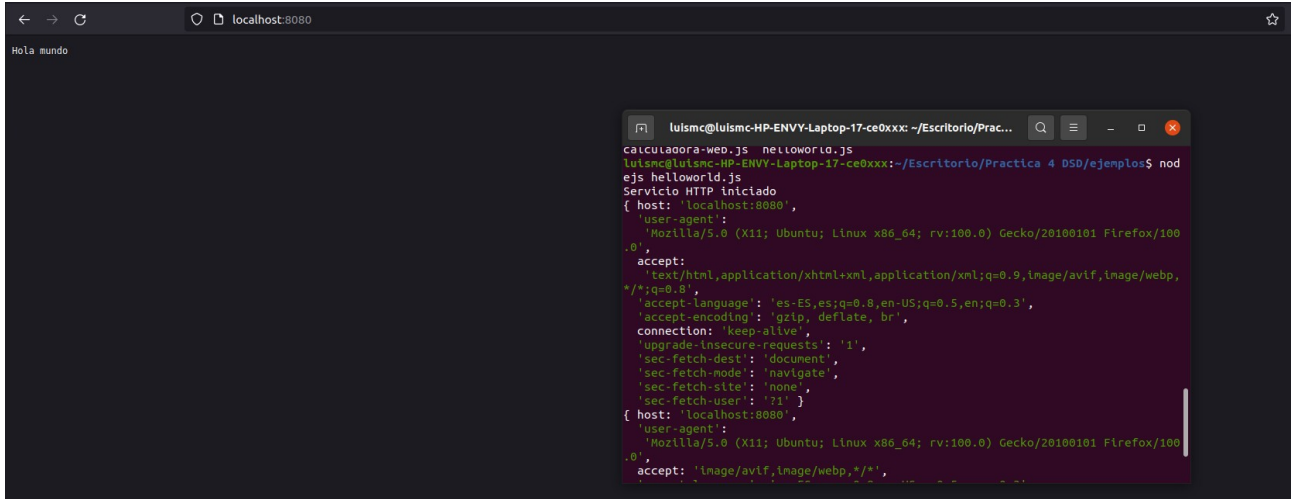


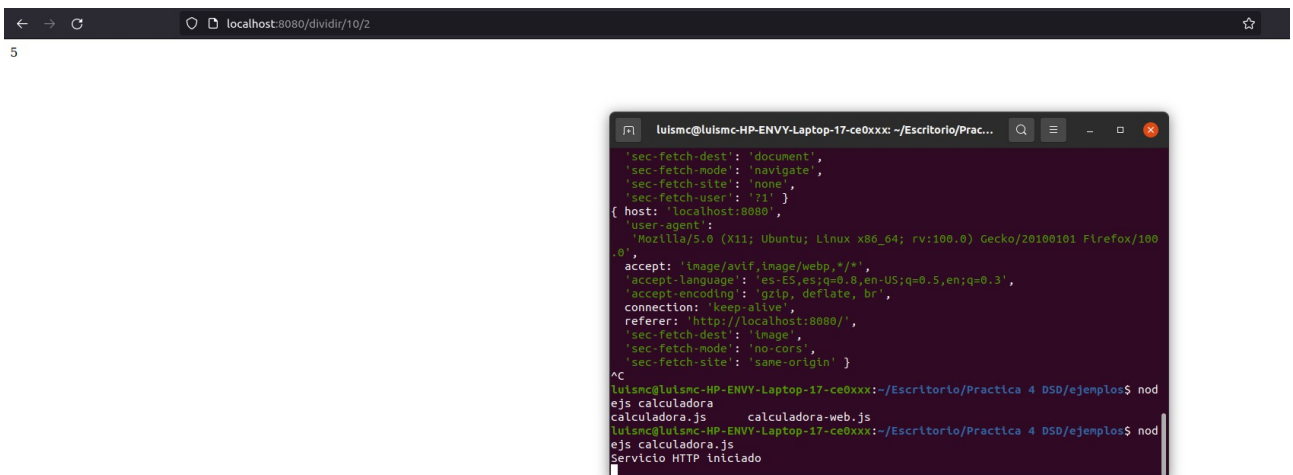
# PRACTICA 4: Node.js

## 1ª Parte: Demostración de los ejemplos

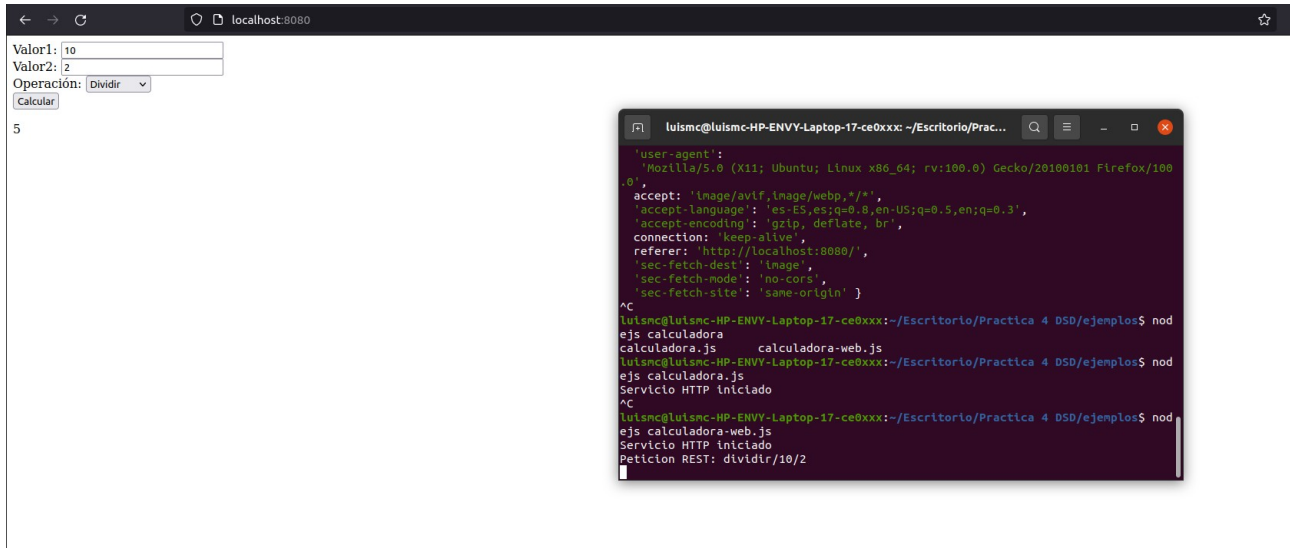
El primer ejemplo es helloworld.js, que nos muestra, a través del puerto localhost:8080, la frase en pantalla “Hola Mundo”. Simplemente se consigue ejecutando por terminal \$nodejs helloworld.js



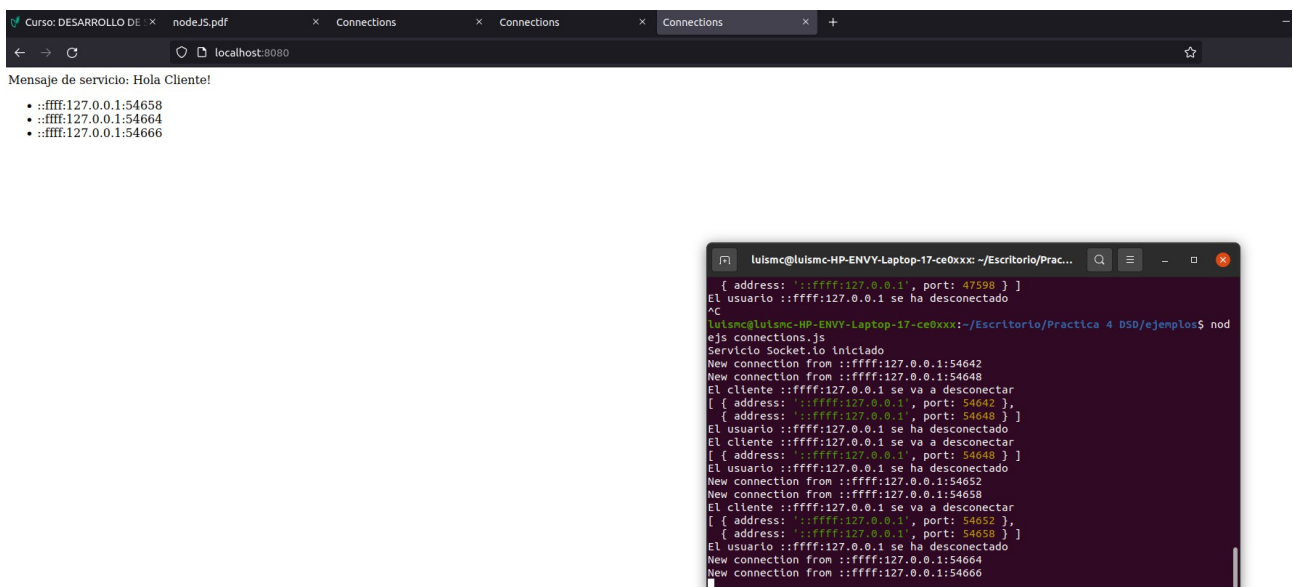
El siguiente ejemplo es calculadora.js, que nos permite introducir una operación matemática básica como suma, resta, multiplicación y división. Mediante la url se especifica que operación queremos realizar y cuales son los operando. Por ejemplo, vamos a realizar la división 10/2, por lo que a la url tendremos que añadirle: “dividir/10/2”.



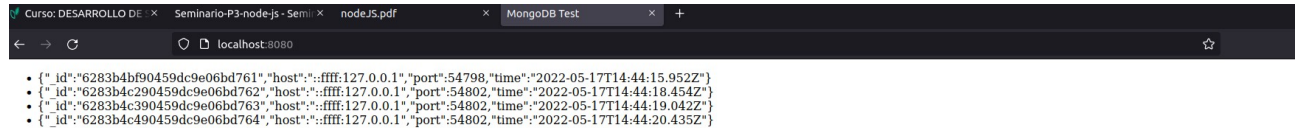
Hay una versión con interfaz de este programa de la calculadora, calculadora-web.js, donde podemos introducir directamente los números que queremos utilizar y la operación que queremos realizar.



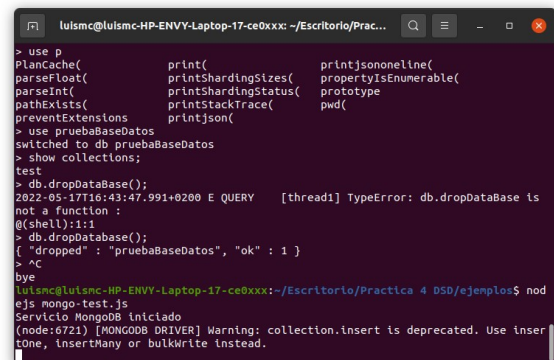
El siguiente ejemplo es connections.js, una implementación de un servicio que envía una notificación la cual contiene las direcciones de todos los clientes conectados a ese servicio. En esta ejecución tengo abiertas 3 pestañas, que son los 3 clientes que está conectados al servicio.



Por último, el ejemplo mongo-test.js, donde muestra, cada vez que se abre un cliente web, aparece una entrada nueva en la lista de las conexiones. Esto se produce debido a que se crea una entrada en la base de datos de prueba cada vez que se accede al localhost:8080, y se guarda un historial de todas las entradas que se han realizado previamente.



```
Curso: DESARROLLO DE x Seminario-P3-node.js - Semi x nodeJS.pdf x MongoDB Test x +
localhost:8080
• {"_id":"6283b4bf90459dc9e06bd761","host":"::ffff:127.0.0.1","port":54798,"time":"2022-05-17T14:44:15.952Z"}
• {"_id":"6283b4c290459dc9e06bd762","host":"::ffff:127.0.0.1","port":54802,"time":"2022-05-17T14:44:18.454Z"}
• {"_id":"6283b4c390459dc9e06bd763","host":"::ffff:127.0.0.1","port":54802,"time":"2022-05-17T14:44:19.042Z"}
• {"_id":"6283b4c490459dc9e06bd764","host":"::ffff:127.0.0.1","port":54802,"time":"2022-05-17T14:44:20.435Z"}
```



```
luisnc@luisnc-HP-ENVY-Laptop-17-ce0xxx: ~/Escritorio/Prac...
> use p
PlanCache(          print(          printJsononeline(
parseFloat(         printShardingSizes(  propertyIsEnumerable(
parseInt(           printShardingStatus(  prototype
pathExists(         printStackTrace(    pwd(
preventExtensions  printJson(
> use pruebaBaseDatos
switched to db pruebaBaseDatos
> show collections;
test
> db.dropDatabase();
2022-05-17T16:43:47.991+0200 E QUERY   [thread1] TypeError: db.dropDatabase is
not a function :
@(shell):1:1
> db.dropDatabase();
{ "dropped" : "pruebaBaseDatos", "ok" : 1 }
> ^C
bye
luisnc@luisnc-HP-ENVY-Laptop-17-ce0xxx:~/Escritorio/Practica 4 DSD/ejemplos$ nod
ejs mongo-test.js
Servicio MongoDB Iniciado
(node:6721) [MONGODB DRIVER] Warning: collection.insert is deprecated. Use inser
tOne, insertMany or bulkWrite instead.
```

## 2ª Parte: Implementación del sistema de domótica

Este sistema consiste en un servidor en el cual se van recogiendo los datos de los sensores, que posteriormente se pasan a la página del cliente en la cual se muestran tanto los datos, como la información del aire y de las persianas. A continuación voy a explicar los archivos principales:

### Cliente.html:

En esta página se cogen los datos de las variables que puedan ser susceptibles de cambios, como el graduaje de la temperatura o la luminosidad, o el estado del aire y las persianas.

El aire y las persianas se pueden modificar siempre a gusto del cliente.

También podemos modificar la luminosidad y la temperatura, pero nos saltará un mensaje de alerta si ponemos algún valor fuera de los límites establecidos y no se cambiarán a los valores introducidos por el cliente. Además, si esto sucede, cambiarán los estados tanto de del aire como de las persianas. Si ponemos una temperatura muy alta, el aire se encenderá y viceversa, y lo mismo con las persianas.

The screenshot shows a web interface with a yellow background. At the top, there are two input fields: 'Temperatura:' with the value '30' and 'Luminosidad:' with the value 'Nivel de luminosidad'. To the right of each input field is a dark grey button labeled 'Enviar temperatura' and 'Enviar luminosidad' respectively. Below these is a large orange rectangular box containing the following text: 'Temperatura: 30', 'Luminosidad: 50', 'Temperatura Máxima: 35°C', 'Temperatura Mínima: 15°C', 'Luminosidad Máxima: 60°C', 'Luminosidad Mínima: 10°C', 'Aire Acondicionado: ENCENDIDO', and 'Persianas: CERRADA'. At the bottom of the orange box are two dark red buttons: 'Abrir/Cerrar Persianas' and 'Apagar/Encender Aire Acondicionado'.

The screenshot shows the same web interface as above, but with a yellow alert banner at the top that reads 'Muy poca luminosidad (Min: 10)'. The 'Luminosidad:' input field now has the value '0'. The orange box contains the same text as before, except for the last two lines, which now read 'Aire Acondicionado: ENCENDIDO' and 'Persianas: ABIERTA'. The buttons at the bottom remain the same.

Por último lo que podemos ver es esta página un poco más abajo son las notificaciones de todos los eventos que van sucediendo, que previamente se han guardado en una base de datos.



Todos los parámetros se van actualizando debido a los eventos de escucha introducidos.

## Servidor.js

En este archivo se controlan todas las escuchas de los sensores y su posterior envío al cliente. Todos estos datos se guardan en una base de datos creada a partir de mongodb, de la misma manera que en los ejemplo de la 1ª parte de esta práctica. Cada vez que se realiza un cambio se guarda en la base de datos para su posterior impresión.

Se mandan los valores cambiados, los mensajes de alerta, y los estados de la temperatura, la luminosidad, las persianas y el aire.

Todo esto se hace mediante una comunicación con Node.js, escuchando las peticiones del cliente y emitiendo los resultados necesarios para poder enviar toda la información necesaria

## Pequeña parte opcional

No se si se puede considerar opcional del todo pero se ha decorado un poco la página del cliente para que sea más vistosa