

Práctica 4 : NodeJS

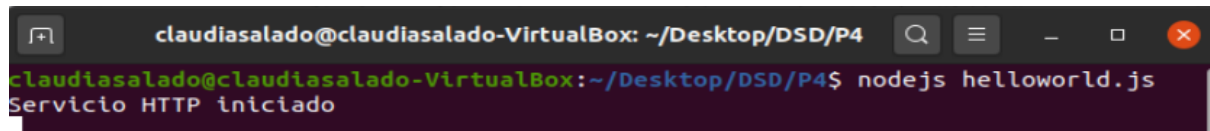
Parte 1 : Implementación de los Ejemplos	2
Ejemplo 1:	2
Ejemplo 2:	3
Ejemplo 3:	3
Ejemplo 4:	4
Ejemplo 5:	5
Parte 2 : Ejercicio - Sistema Domótico	6
Funcionamiento:	6
Código Prueba.js:	9
Código Servidor.html:	13

Claudia Salado Méndez DSD3

Parte 1 : Implementación de los Ejemplos

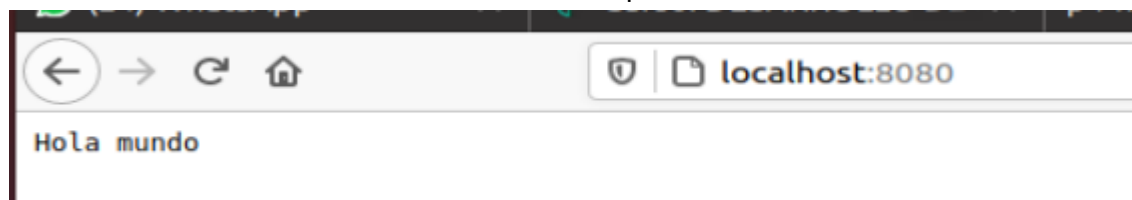
Ejemplo 1:

Lanzamos helloworld.js



```
claudiasalado@claudiasalado-VirtualBox: ~/Desktop/DSD/P4
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs helloworld.js
Servicio HTTP iniciado
```

A continuación nos metemos en localhost:8080 para visualizar el resultado.



Mientras tanto en la terminal va apareciendo:



```
Servicio HTTP iniciado
{ host: 'localhost:8080',
  'user-agent':
    'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:88.0) Gecko/20100101 Firefox/88.0',
  accept:
    'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
  'accept-language': 'en-US,en;q=0.5',
  'accept-encoding': 'gzip, deflate',
  connection: 'keep-alive',
  'upgrade-insecure-requests': '1' }
{ host: 'localhost:8080',
  'user-agent':
    'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:88.0) Gecko/20100101 Firefox/88.0',
  accept: 'image/webp,*/*',
  'accept-language': 'en-US,en;q=0.5',
  'accept-encoding': 'gzip, deflate',
  connection: 'keep-alive',
  referer: 'http://localhost:8080/' }
```

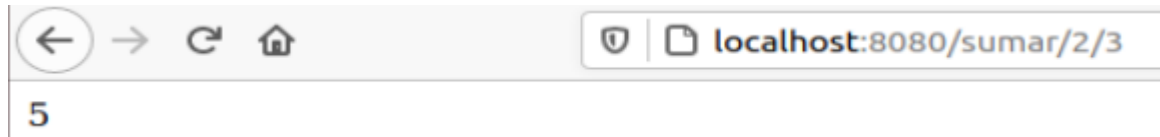
En este código se obtiene un módulo disponible en Node.js, en este caso el http que nos deja implementar servicios que sirvan contenidos usando el protocolo http. Se crea un servidor http en su constructor donde hay dos parámetros el response y el request, en el request se codifica el mensaje y este se imprime con el response.

Ejemplo 2:

Probamos ahora el ejemplo de la calculadora.js:

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs calculadora.js
Servicio HTTP iniciado
```

Nos metemos en el puerto 8080 y ponemos /sumar/2/3 por ejemplo, entonces nos muestra el resultado:



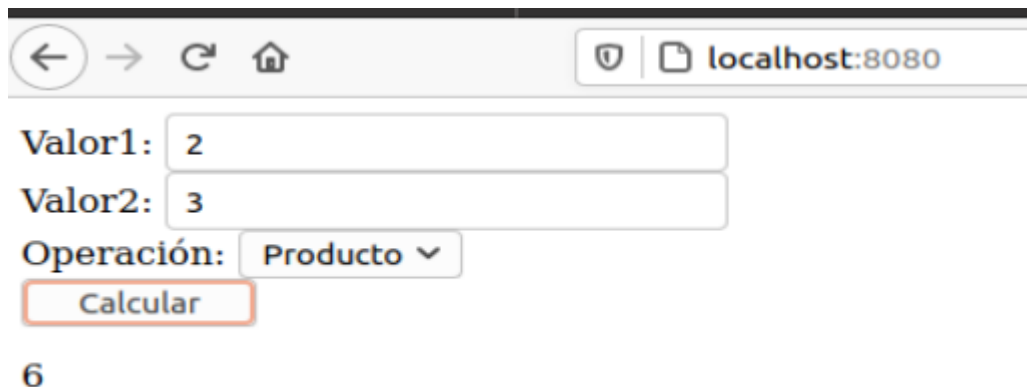
Este Código utiliza peticiones REST como la que hemos usado en el ejemplo, localhost:8080/sumar/2/3 y nos devuelve directamente el resultado, este tiene formato html, hay que especificar que el tipo MIME de la respuesta es text/html.

Ejemplo 3:

Lanzamos calculadora-web.js

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs calculadora-web.js
Servicio HTTP iniciado
```

Nos metemos en el localhost y rellenamos el formulario le damos a calcular y nos sale el resultado.



Mientras tanto nos sale en la terminal la operación que hemos llevado a cabo.

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs calculadora-web.js
Servicio HTTP iniciado
Petición REST: producto/2/3
```

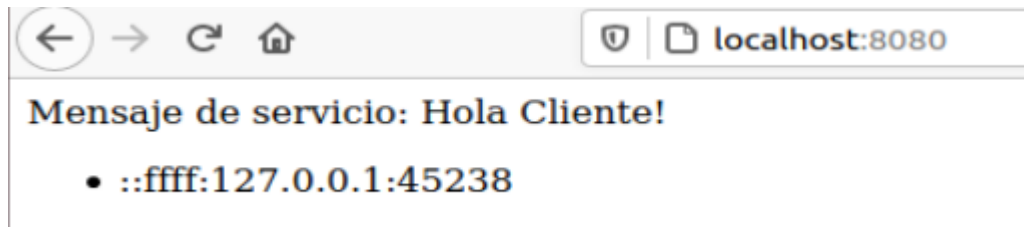
Aquí se utiliza un módulo al que llamamos fs este permite realizar operaciones de entrada y salida sobre unos ficheros del servidor. Se leen los ficheros almacenados en la carpeta donde se ejecuta el código, y los devuelve al cliente como respuesta, la lectura de estos ficheros es asíncrona.

Ejemplo 4:

Ejecutamos el connections.js:

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs connections.js
Servicio Socket.io iniciado
```

Y nos metemos en el localhost:



Mientras tanto también se nos muestra lo siguiente en la terminal:

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs connections.js
Servicio Socket.io iniciado
Petición invalida: /favicon.ico
New connection from ::ffff:127.0.0.1:45238
```

Este ejemplo es una implementación sobre Socket.io de un servicio que nos notifica de todas las direcciones de los clientes que están conectados al servidor.

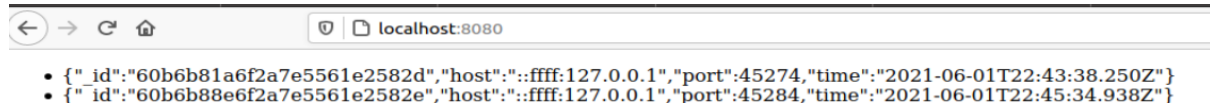
Esto se envía a todos los clientes cada vez que uno nuevo se conecte o desconecte. Se utiliza la función emit sobre un array donde están los clientes, además le envía al cliente un mensaje.

Ejemplo 5:

Ejecutamos en la terminal mongo-text.js:

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs mongo-test.js
Servicio MongoDB iniciado
(node:21857) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(node:21857) DeprecationWarning: collection.insert is deprecated. Use insertOne, insertMany or bulkWrite instead.
```

Nos metemos en el localhost y nos sale esto al recargar:



A screenshot of a web browser window showing a JSON array of two objects. The browser's address bar shows 'localhost:8080'. The page content displays two JSON objects, each with fields for '_id', 'host', 'port', and 'time'.

- {"_id":"60b6b81a6f2a7e5561e2582d","host":"::ffff:127.0.0.1","port":45274,"time":"2021-06-01T22:43:38.250Z"}
- {"_id":"60b6b88e6f2a7e5561e2582e","host":"::ffff:127.0.0.1","port":45284,"time":"2021-06-01T22:45:34.938Z"}

Mientras tanto por la terminal se nos muestra:

```
claudiasalado@claudiasalado-VirtualBox:~/Desktop/DSD/P4$ nodejs mongo-test.js
Servicio MongoDB iniciado
(node:21857) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(node:21857) DeprecationWarning: collection.insert is deprecated. Use insertOne, insertMany or bulkWrite instead.
insertOne
Petición invalida: /favicon.ico
```

En este ejemplo tenemos un servicio que recibe dos clases de notificaciones mediante socket.io, el contenido de la primera es introducido por el servicio en la base de datos, en la colección de claves-valor. Cuando recibe el segundo tipo se hace una consulta sobre la base de datos en base al contenido de la notificación y devuelve los resultados al cliente

Parte 2 : Ejercicio - Sistema Domótico

Funcionamiento:

He modificado un poco lo que debería hacer el servicio, este lo que hace es que detecta cuando la temperatura o la luminosidad superan los umbrales y nos lo muestra en el log de eventos, también si se encienden las luces, la calefacción o el aire nos avisa en que franja horaria estamos y si resulta cara o no:

- Cuando la temperatura supera la máxima el aire acondicionado se enciende y en el caso de que estuviera la calefacción encendida se apaga esta.

The screenshot displays the home automation interface. On the left, there are two control panels: 'TEMPERATURA' (Temperature) and 'LUMINOSIDAD' (Luminosity). The 'TEMPERATURA' panel shows 'Actual' (Actual) at 42, 'Mínimo' (Minimum) at 15, and 'Máximo' (Maximum) at 35. It has a button 'Actualizar valores' (Update values) and two toggle switches: 'Aire Acondicionado' (Air Conditioning) which is turned on, and 'Calefacción' (Heating) which is turned off. The 'LUMINOSIDAD' panel shows 'Actual' at 4, 'Mínimo' at 3, and 'Máximo' at 7. It also has a button 'Actualizar valores' and two toggle switches: 'Persianas' (Blinds) which is turned off, and 'Luces' (Lights) which is turned off. On the right, there is a section titled 'Servicio Domótico' (Home Automation Service) with a sub-header 'EVENTOS' (Events). Below this, a log shows the following events: '06/05/2021 - 20:15:36 - Aire Acondicionado Encendido' (Air Conditioning turned on), 'Atención: Estas en la franja horaria más cara' (Attention: You are in the most expensive time slot), 'Aviso: temperatura fuera de umbral' (Warning: temperature out of range), '06/05/2021 - 20:15:36 - Calefacción Apagada' (Heating turned off), '06/05/2021 - 20:15:36 - Persianas Abierta' (Blinds open), and '06/05/2021 - 20:15:36 - Luces Apagadas' (Lights turned off).

- Cuando la temperatura es menor que la mínima pasa el caso contrario se enciende la calefacción y se apaga el aire.

The screenshot displays the home automation interface. On the left, there are two control panels: 'TEMPERATURA' (Temperature) and 'LUMINOSIDAD' (Luminosity). The 'TEMPERATURA' panel shows 'Actual' (Actual) at 14, 'Mínimo' (Minimum) at 15, and 'Máximo' (Maximum) at 35. It has a button 'Actualizar valores' (Update values) and two toggle switches: 'Aire Acondicionado' (Air Conditioning) which is turned off, and 'Calefacción' (Heating) which is turned on. The 'LUMINOSIDAD' panel shows 'Actual' at 4, 'Mínimo' at 3, and 'Máximo' at 7. It also has a button 'Actualizar valores' and two toggle switches: 'Persianas' (Blinds) which is turned off, and 'Luces' (Lights) which is turned off. On the right, there is a section titled 'Servicio Domótico' (Home Automation Service) with a sub-header 'EVENTOS' (Events). Below this, a log shows the following events: '06/05/2021 - 20:17:49 - Calefacción Encendida' (Heating turned on), 'Atención: Estas en la franja horaria más cara' (Attention: You are in the most expensive time slot), 'Aviso: temperatura fuera de umbral' (Warning: temperature out of range), '06/05/2021 - 20:17:49 - Aire Acondicionado Apagado' (Air Conditioning turned off), '06/05/2021 - 20:17:49 - Persianas Abierta' (Blinds open), and '06/05/2021 - 20:17:49 - Luces Apagadas' (Lights turned off).

- Cuando la temperatura está entre la máxima y la mínima, se apaga tanto la calefacción como el aire acondicionado. No nos da avisos ni de haber superado los umbrales ni de que estamos en la franja horaria de gasto X.

TEMPERATURA

Actual: 21

Mínimo: 15

Máximo: 35

Actualizar valores

☐ Aire Acondicionado

☐ Calefacción

LUMINOSIDAD

Actual: 4

Mínimo: 3

Máximo: 7

Actualizar valores

☐ Persianas

☐ Luces

Servicio Domótico

EVENTOS

06/05/2021 - 20:19:26 - Aire Acondicionado Apagado

06/05/2021 - 20:19:26 - Calefacción Apagada

06/05/2021 - 20:19:26 - Persianas Abierta

06/05/2021 - 20:19:26 - Luces Apagadas

- Cuando la luminosidad supera el umbral máximo, significa que hay mucha luz, se cierran las persianas y se encienden las luces.

TEMPERATURA

Actual: 21

Mínimo: 15

Máximo: 35

Actualizar valores

☐ Aire Acondicionado

☐ Calefacción

LUMINOSIDAD

Actual: 9

Mínimo: 3

Máximo: 7

Actualizar valores

☒ Persianas

☒ Luces

Servicio Domótico

EVENTOS

06/05/2021 - 20:21:18 - Aire Acondicionado Apagado

Atención: Estas en la franja horaria más cara

Aviso: luminosidad fuera de umbral

06/05/2021 - 20:21:18 - Calefacción Apagada

06/05/2021 - 20:21:18 - Persianas Cerrada

06/05/2021 - 20:21:18 - Luces Encendidas

- Cuando la luminosidad es muy baja las luces están apagadas y se suben las persianas, nos avisa también de que se ha superado el umbral mínimo en la luminosidad.

The interface consists of two control panels on the left and an event log on the right.

TEMPERATURA Panel:

- Actual: 21
- Mínimo: 15
- Máximo: 35
- Buttons: (disabled), ☐ Aire Acondicionado, ☐ Calefacción

LUMINOSIDAD Panel:

- Actual: 1
- Mínimo: 3
- Máximo: 7
- Buttons: (disabled), ☐ Persianas, ☐ Luces

Servicio Domótico EVENTOS Log:

- 06/05/2021 - 20:22:37 - Aire Acondicionado Apagado
- Aviso: luminosidad fuera de umbral
- 06/05/2021 - 20:22:37 - Calefacción Apagada
- 06/05/2021 - 20:22:37 - Luces Apagadas
- 06/05/2021 - 20:22:37 - Persianas Abierta

- Cuando la luminosidad está dentro de los dos umbrales las luces se quedan apagadas y las persianas abiertas, la diferencia es que no avisa de que se han superado los umbrales de luminosidad.

The interface is identical to the previous one, but with updated values in the control panels.

TEMPERATURA Panel:

- Actual: 21
- Mínimo: 15
- Máximo: 35
- Buttons: (disabled), ☐ Aire Acondicionado, ☐ Calefacción

LUMINOSIDAD Panel:

- Actual: 6
- Mínimo: 3
- Máximo: 7
- Buttons: (disabled), ☐ Persianas, ☐ Luces

Servicio Domótico EVENTOS Log:

- 06/05/2021 - 20:25:09 - Aire Acondicionado Apagado
- 06/05/2021 - 20:25:09 - Calefacción Apagada
- 06/05/2021 - 20:25:09 - Persianas Abierta
- 06/05/2021 - 20:25:09 - Luces Apagadas

Tanto luminosidad como temperatura trabajan simultáneamente, si exceden ambas los umbrales nos saldrá otro mensaje que es que los sensores exceden los umbrales.

Código Prueba.js:

Lo primero es el código para las conexiones y variables:

```
1 //Creamos las constantes y variables
2 const http = require ('http');
3 const url = require ('url');
4 const fs = require ('fs');
5 const path = require ('path');
6 const socketio = require ('socket.io');
7 var MongoClient = require('mongodb').MongoClient;
8 var MongoServer = require('mongodb').Server;
9
```

Estas tres funciones trabajan con la fecha, la primera lo que hace es guardar la fecha en el formato que queremos para mostrarlo por el log, la segunda lo único que hace es devolvernos la hora porque la necesitaremos para uno de los avisos, y la tercera lo que hace es que si un día, mes etc no tiene dos cifras por ejemplo es el día 3 pues lo pone como 03.

```
//Funcion para guardarla fecha en el formato que deseamos
function getTimestamp(){
    var date      = new Date();
    var dia       = ModificarFecha(date.getDate());
    var mes       = ModificarFecha(date.getMonth());
    var fecha     = dia + "/" + mes + "/" + date.getFullYear();
    var horas     = ModificarFecha(date.getHours());
    var minutos   = ModificarFecha(date.getMinutes());
    var segundos  = ModificarFecha(date.getSeconds());
    var hora      = horas + ":" + minutos + ":" + segundos;
    date          = fecha + " - " + hora;

    return date; //Nos devolvera por ejemplo 17/04/2021 - 23:39
}

function getHour(){
    var date = new Date();
    var hora = date.getHours();
    return hora;
}

//Al hacer lo anterior puede ser que se tenga solo una cifra por tanto es funcion
añade un 0 antes en ese dato.
function ModificarFecha(date){
    var nueva = date;
    if(date<10){
        nueva = "0"+date;
    }
    return nueva ;
}
```

Esta función lo que hace es guardar los ficheros html y css que vamos a usar si no los detecta da error por la terminal

```
//Funcion que añade los ficheros html y css
var httpServer = http.createServer (
  function (request, response) {

    switch(request.url){
      case "/estilo.css":
        response.writeHead(200, {"Content-Type": "text/css"});
        var file = path.join(process.cwd(), "/estilo.css");
        fs.exists (file, function (exists) {
          if(exists){
            fs.readFile (file, function (err, data) {
              if (!err)
                response.end(data);
            });
          }
        });
        break;

      default:
        response.writeHead(200, {"Content-Type": "text/html"});
        var file = path.join(process.cwd(), "/servidor.html");
        fs.exists (file, function (exists) {
          if(exists){
            fs.readFile (file, function (err, data) {
              if (!err)
                response.end(data);
            });
          }
        });
        break;
    }
  }
);
```

Aquí hago la conexión con MongoDB y creó las colecciones para temperatura y luminosidad

```
MongoClient.connect("mongodb://localhost:27017/domotica", { useNewUrlParser: true }, function(err, db) {
  if(err)
    throw err;

  var bdatos = db.db("domotica");
  var mensaje = null;
  var puerto = 8080;

  console.log("Servidor listo en puerto: " + puerto);
  var io = socketio(httpServer);

  bdatos.createCollection("luminosidad", function(err, collection){
    if(!err){
      console.log("Colección creada en Mongo: " + collection.collectionName);
    }
  });

  bdatos.createCollection("temperatura", function(err, collection){
    if(!err){
      console.log("Colección creada en Mongo: " + collection.collectionName);
    }
  });

  io.sockets.on('connection',function(socket) {
    console.log("Conectado");
  });
});
```

A continuación insertamos en luminosidad temperatura los valores máximos, mínimos y el actual.

```

    /** Inserción de colección */
    /** Valores */
    socket.on('enviar-datos-temperatura', function (datos) {
        bdatos.collection("temperatura").insert({valor:datos[0], minimo:datos[1], maximo:datos[2]},
        {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Temperaturas: {valor:"+datos[0]+", minimo:"+datos[1]+", maximo:"+datos[2]+"}");
                io.sockets.emit('log', getTimestamp() + " - Modificados valores de temperatura");
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

    /** Inserción de colección Luminosidad */
    /** Valores */
    socket.on('enviar-datos-luminosidad', function (datos) {
        bdatos.collection("luminosidad").insert({valor:datos[0], minimo:datos[1], maximo:datos[2]},
        {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Luminosidad : {valor:"+datos[0]+", minimo:"+datos[1]+", maximo:"+datos[2]+"}");
                io.sockets.emit('log', getTimestamp() + " - Modificados valores de luminosidad");
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

```

También tenemos que crear los estados de los sliders y los mensajes que queremos que aparezcan cuando estos se enciendan o apaguen. Como tenemos 4, pues tenemos que hacer uno para cada uno.

```

    /** Estado */
    socket.on('slider_1', function (datos) {
        bdatos.collection("temperaturas").insert({estado:datos}, {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Temperaturas: {estado:"+datos+"}");
                io.sockets.emit('log', getTimestamp() + " - Aire Acondicionado " + datos);
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

    /** Estado */
    socket.on('slider_3', function (datos) {
        bdatos.collection("temperaturas").insert({estado:datos}, {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Temperaturas: {estado:"+datos+"}");
                io.sockets.emit('log', getTimestamp() + " - Calefacción " + datos);
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

    /** Estado Persianas */
    socket.on('slider_2', function (datos) {
        bdatos.collection("luminosidad").insert({estado:datos}, {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Luminosidad: {estado:"+datos+"}");
                io.sockets.emit('log', getTimestamp() + " - Persianas " + datos);
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

    /** Estado Luces */
    socket.on('slider_4', function (datos) {
        bdatos.collection("luminosidad").insert({estado:datos}, {safe:true}, function(err, result) {
            if (!err) {
                console.log("Insertado en Luminosidad: {estado:"+datos+"}");
                io.sockets.emit('log', getTimestamp() + " - Luces " + datos);
            }
            else
                console.log("Error al insertar datos en la colección.");
        });
    });

```

Por último tengo los avisos, he creado tres distintos, uno que saltara cuando los umbrales tanto de luminosidad como de temperatura superen los máximos, el otro mensaje nos dice que sensor a superado un umbral ya sea el de luminosidad o el de temperatura, y el último tipo de mensaje es el que mira la hora en la que estamos y dependiendo de la que sea y a que franja horaria pertenezca nos dice cual es el coste al encender la calefacion, las luces o el aire acondicionado.

```
/** Avisos */
socket.on ('aviso', function (sensor) {
  var log = "Aviso: " + sensor + " fuera de umbral";
  console.log (log);
  io.sockets.emit ('aviso', log);
});

socket.on ('accion', function () {
  var log = "Aviso: los sensores exceden los umbrales máximos";
  console.log (log);
  io.sockets.emit ('aviso', log);
})

socket.on ('tarifa', function () {
  var hora = getHour();
  if ((hora>=0) && (hora<8))
    var log = "Atención: Estas en la franja horaria más barata";
  else if (((hora>=8) && (hora<10)) || ((hora>=14) && (hora<18)) || ((hora>=22) && (hora<=23)))
    var log = "Atención: Estas en la franja horaria intermedia";
  else
    var log = "Atención: Estas en la franja horaria más cara";
  console.log (log);
  io.sockets.emit ('tarifa', log);
})
```

Código Servidor.html:

Primero pongo el código correspondiente a la caja de las temperaturas, donde se nos muestra el máximo, mínimo, y el valor actual y los sliders de la calefacción y del aire.

```
<!-- Aire Acondicionado -->
<div class="temperatura">
  <form class="formulario" id="aire">
    <h2> Temperatura </h2>
    <div class="elemento centrado">
      <label for="valor">Actual: </label>
      <input name="valor" type="number" value="25" min="0" max="50"/>
    </div>
    <div class="elemento centrado">
      <label for="min">Mínimo: </label>
      <input name="min" type="number" value="15" min="0" max="25"/>
    </div>
    <div class="elemento centrado">
      <label for="max">Máximo: </label>
      <input name="max" type="number" value="35" min="25" max="50"/>
    </div>

    <input class="centrado" type="submit" name="submit" value="Actualizar valores"
      onclick="enviar('aire');return false;"/>
    <div class="switch_">
      <div class="main" style="grid-gap: 50%;">
        <label class="switch centrado">
          <h3> Aire Acondicionado </h3>
          <input type="checkbox" id="slider_1" onclick="encender('slider_1');">
          <span class="slider round"></span>
        </label>
      </div>
      <div class="main" style="grid-gap: 50%;">
        <label class="switch centrado">
          <h3> Calefacción </h3>
          <input type="checkbox" id="slider_3" onclick="encender('slider_3');">
          <span class="slider round"></span>
        </label>
      </div>
    </div>
  </form>
</div>
```

Justo a continuación tengo lo mismo para la caja o apartado de luminosidad.

```
<!-- Persianas -->
<div class="luminosidad">
  <form class="formulario" id="persianas">
    <h2> Luminosidad </h2>
    <div class="elemento centrado">
      <label for="valor">Actual: </label>
      <input name="valor" type="number" value="5" min="0" max="10"/>
    </div>
    <div class="elemento centrado">
      <label for="min">Mínimo: </label>
      <input name="min" type="number" value="3" min="0" max="10"/>
    </div>
    <div class="elemento centrado">
      <label for="max">Máximo: </label>
      <input name="max" type="number" value="7" min="0" max="10"/>
    </div>

    <input class="centrado" type="submit" name="submit"
      value="Actualizar valores" onclick="enviar('persianas');return false;"/>
    <div class="switch_">
      <div class="main" style="grid-gap: 50%;">
        <label class="switch centrado">
          <h3> Persianas </h3>
          <input type="checkbox" id="slider_2" onclick="encender('slider_2');">
          <span class="slider round"></span>
        </label>
      </div>
      <div class="main" style="grid-gap: 50%;">
        <label class="switch centrado">
          <h3> Luces </h3>
          <input type="checkbox" id="slider_4" onclick="encender('slider_4');">
          <span class="slider round"></span>
        </label>
      </div>
    </div>
  </form>
</div>
```

Por último en código html tengo puesto el recuadro donde nos aparecerán los eventos.

```
<div class="centrado log-parent actuador">
  <h1>Servicio Domótico</h1>
  <h1>EVENTOS</h1>
  <div id="log"></div>
</div>

</body>
```

Esta función guarda los datos de los valores de luminosidad y temperatura, a parte reconoce que sensor es y envía la información al servidor.

```
function enviar(id) {
  var formulario = document.getElementById(id);
  var datos = new Array();
  // Valor
  datos.push(formulario[0].value);
  // Mínimo
  datos.push(formulario[1].value);
  // Máximo
  datos.push(formulario[2].value);

  // Envío de información al Agente
  (id == "aire") ? sensor = "temperatura" : sensor = "luminosidad";
  agente (sensor, datos);

  // Envío de información al servidor
  socket.emit('enviar-datos-'+id, datos);
  // alert("Datos enviados: enviar-datos-"+ id + " -> " + datos.toString());
}
```

Esta función enciende/apaga los slicers, hay uno para la calefacción, otro para el aire acondicionado, para las persianas y para las luces, además es cuando usamos el aviso de la tarifa, que saltará cuando se encienda el aire, la calefacción o las luces.

```
//Encender con el slicer
function encender(slider_id){
  var slider = document.getElementById(slider_id);
  var estado;

  //Slider del aire acondicionado esta apagado en un principio
  if (slider_id=="slider_1") {
    ((slider.checked) ? estado="Encendido" : estado="Apagado");
    socket.emit(slider_id, estado);
    if (estado == "Encendido")
      socket.emit ("tarifa");
  }
  //Slider de las persianas estan abiertas en un principio
  else if(slider_id=="slider_2"){
    ((slider.checked) ? estado="Cerrada" : estado="Abierta");
    socket.emit(slider_id, estado);
  }
  //Slider de la calefacción esta apagada al principio
  else if(slider_id=="slider_3"){
    ((slider.checked) ? estado="Encendida" : estado="Apagada");
    socket.emit(slider_id, estado);
    if (estado == "Encendida")
      socket.emit ("tarifa");
  }
  //Slider de las luces estan apagadas al principio
  else {
    ((slider.checked) ? estado="Encendidas" : estado="Apagadas");
    socket.emit(slider_id, estado);
    if (estado == "Encendidas")
      socket.emit ("tarifa");
  }
  // alert("Slider: " + estado);
}
```

Esta función lo que hace es actualizar los eventos cada vez que nosotros pulsemos el botón de actualizar y detectar el tipo de aviso, para ponerlo de un color u otro, si es de que se han superado los umbrales será rojo, si es de la tarifa verde.

```
//Actualiza el registro de eventos y detecta el tipo de mensaje para cambiar el color
function actualizarLog (mensaje, importante) {
    var div = document.getElementById("log");
    if (importante==true)
        var evento = "<p style='color: red;'> "+mensaje+" </p>";
    else if (importante==false)
        var evento = "<p style='color: green;'> "+mensaje+" </p>"
    else
        var evento = "<p> "+mensaje+" </p>"
    div.innerHTML += evento;

    // Actualizar scroll
    div.scrollTop = div.scrollHeight;
}

//Se reciben los eventos
socket.on ('log', function (mensaje) {
    actualizarLog (mensaje);
});

socket.on ('aviso', function (mensaje) {
    actualizarLog (mensaje, true);
});

socket.on ('tarifa', function (mensaje) {
    actualizarLog (mensaje, false);
});
});
```

Por último tenemos el agente, que bueno ya lo que hace es detectar cuando se superan los umbrales y que slicer tiene que encender en cada caso al igual que emitir los avisos correspondientes.

```
//AGENTE
function agente (sensor, datos) {
    var temp = document.getElementById("aire");
    var luz = document.getElementById("persianas");
    var umbral_L = false;
    var umbral_T = false;

    //Si la temperatura supera la maxima se enciende el aire
    if (+temp[0].value > +temp[2].value){
        umbral_T = true;
        document.getElementById("slider_1").checked = true;
        encender("slider_1");
        document.getElementById("slider_3").checked = false;
        encender("slider_3");
    }

    //Si la temperatura es menor que la minima se enciende la calefacion
    else if (+temp[1].value > +temp[0].value){
        umbral_T = true;
        document.getElementById("slider_3").checked = true;
        encender("slider_3");
        document.getElementById("slider_1").checked = false;
        encender("slider_1");
    }

    //Si esta dentro del umbral se apaga el aire y la calefacion
    else if ((+temp[0].value < +temp[2].value ) && (+temp[1].value < +temp[0].value )){
        document.getElementById("slider_1").checked = false;
        encender("slider_1");
        document.getElementById("slider_3").checked = false;
        encender("slider_3");
    }

    else if ( (+datos[0] < +datos[1]) || (+datos[0] > +datos[2]) )
        socket.emit ("aviso", sensor);
}
```



```

//LUMINOSIDAD
//Si esta dentro del umbral se apagan las luces y las persianas se dejan abiertas
if ((+luz[0].value < +luz[2].value ) && (+luz[1].value < +luz[0].value )){
    document.getElementById("slider_2").checked = false;
    encender("slider_2");
    document.getElementById("slider_4").checked = false;
    encender("slider_4");
}
//Si la luminosidad supera el umbral maximo se encienden las luces y se cierran las persianas
else if (+luz[0].value > +luz[2].value ){
    umbral_L = true;
    document.getElementById("slider_2").checked = true;
    encender("slider_2");
    document.getElementById("slider_4").checked = true;
    encender("slider_4");
}
//Si la luminosidad supera el umbral minimo se apagan las luces y se abren las persianas
else if (+luz[0].value < +luz[1].value ){
    umbral_L = true;
    document.getElementById("slider_2").checked = false;
    encender("slider_2");
    document.getElementById("slider_4").checked = false;
    encender("slider_4");
}

if ((umbral_L == true) && (umbral_T == true))
    socket.emit ("accion");
else if (umbral_L == true)
    socket.emit ("aviso", "luminosidad");
else if (umbral_T == true)
    socket.emit ("aviso", sensor);
}

```