

# PROYECTO NUESTRA TABLA PERIÓDICA



escuela técnica superior  
de ingeniería informática



Tutor del proyecto:

Luis Miguel Soria Morillo.

Integrantes del proyecto:

Jaime Cortés Vázquez.

Francisco Javier Viera Chaves.

Asier Herrería Oña.

# PROYECTO NUESTRA TABLA PERIÓDICA

## • Introducción:

Antes de ahondar en componentes y funciones del proyecto propuesto hay que entender los objetivos impuestos por el plan docente de la asignatura “Desarrollo de Aplicaciones Distribuidas” dentro de la titulación Ingeniería de Computadores.

Como bien indica el nombre de dicha asignatura el proyecto tiene que estar basado en un sistema distribuido que haga uso de los diferentes recursos y tecnologías propuestos por el profesorado.

### ¿QUE ES UN SISTEMA DISTRIBUIDO?

Un sistema distribuido está formado por un conjunto de computadores autónomos entre si conectados por una red equipados con un sistema distribuido.

Podemos definir entonces un sistema distribuido como aquel en el que los componentes hardware o software, que se encuentran en los equipos enlazados por una red se comunican y coordinan mediante paso de mensajes.



Ejemplo de sistema distribuido.

Para conseguir esto se han propuesto dos placas: Arduino y ESP8266.

Estas son plataformas de software libre usados para modelar prototipos que utilizan un software y hardware con poca complejidad y fácil de usar (como es nuestro caso).

Nuestro objetivo será conectar Arduino a un servidor web mediante la ESP8266, que de serie dispone de un módulo Wifi, e interactuar con él

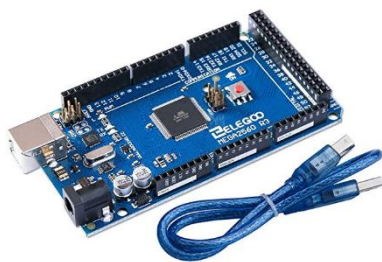
# PROYECTO NUESTRA TABLA PERIÓDICA

mediante el paso de mensajes generados por los sensores conectados a la placa (en nuestro caso serán botones).

## • Componentes:

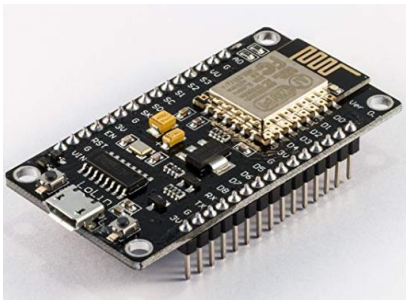
Para conseguir esto necesitaremos los siguientes componentes:

### 1. Placa Arduino.



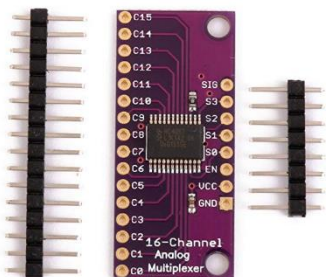
En esta placa irán todas las entradas y salidas utilizadas para conectar cada una de las celdas de la tabla periódica y poder comunicarnos así con el ESP8266.

### 2. Placa ESP8266NodeMCU.



Esta placa será la que recoja la información transmitida por el Arduino y se encargue de comunicarse con el servidor web. Su funcionamiento es similar al de Arduino, el motivo por el cual la usamos es por el módulo Wifi que incorpora de serie y facilita de esta forma la comunicación con el servidor web.

### 3. 9 Multiplexores CD74HC4067 de 16 canales.



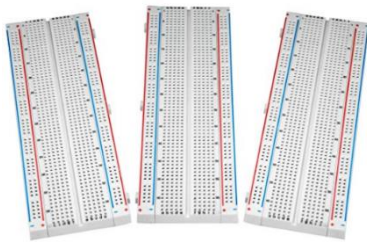
Este circuito integrado nos permitirá multiplicar por 16 la capacidad de puertas de nuestro Arduino. Posee 16 canales en donde se encontrarán los botones correspondientes a cada uno de los elementos de la tabla periódica, una salida por donde recogeremos la información correspondiente al botón que se ha pulsado,

# PROYECTO NUESTRA TABLA PERIÓDICA

una señal de enable que permite el funcionamiento del circuito o lo inhibe y 4 entradas de selección que nos permitirán escoger cual de los canales (botones) ha sido activado y pasar a la salida un resultado binario de 4 bits en este caso.

Como son 118 elementos serían necesarios 9 multiplexores, cada uno de ellos ocupa 5 pines en el Arduino, 45 pines en total, por ello vamos a hacer uso de la placa Arduino grande de 52 pines y no la pequeña de 14 que es más barata pero no cumple con los requisitos.

## 4. 2 Packs de 3 protoboas.



Estas placas nos proporcionaran las conexiones electrónicas necesarias entre los multiplexores, el Arduino y el ESP8266, sin necesidad de soldar.

## 5. Cables y botones.



Cables y botones necesarios para conectar cada uno de los elementos correspondientes a la tabla periódica con el circuito principal.

# PROYECTO NUESTRA TABLA PERIÓDICA

## 6. Actuadores (Altavoz y pantalla LCD).



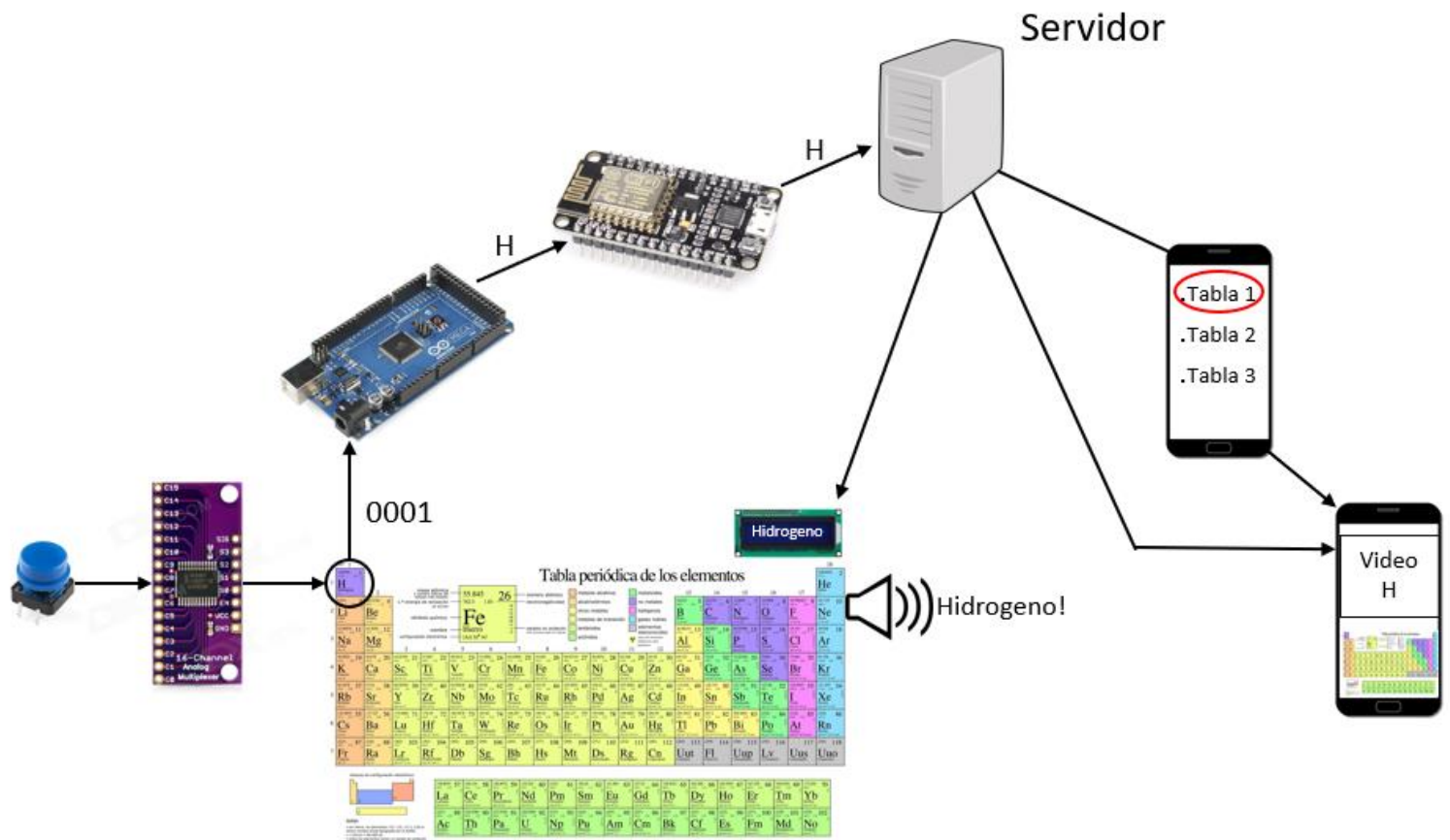
El altavoz simplemente se encargará de informar que elemento es el que ha sido seleccionado y a su vez el LCD mostrará el nombre de dicho elemento. ¿Por qué usamos esto si no es necesario para la realización del proyecto? Nos exigen hacer uso de unos actuadores que reaccionen a las comunicaciones que se realicen entre los componentes que forman la tabla periódica y el servidor web, por lo que la forma más fácil y barata de implementar esto es añadiendo un pequeño altavoz y una pantalla LCD a la tabla periódica que se accionen cuando haya comunicación entre nuestro sistema y el servidor.

Por último matizar que como es una aplicación distribuida necesitamos desarrollar una aplicación que interactúe con nuestro sistema, por lo que hemos optado por desarrollar una aplicación para dispositivos Android que muestre los videos explicativos correspondientes a cada elemento que se guardaran en una base de datos a la que accederá el servidor web, y nos dará mucho juego si en un futuro se quieren implementar muchas más funcionalidades que por razones de tiempo en primera instancia no van a poder implementarse.

Somos conscientes de que esta prohibido el uso de teléfonos móviles en centros escolares por lo que indagando un poco hemos encontrado que haciendo uso de un miniPC (Raspberry Pi 3) que conste de sistema operativo Android e integrándolo en nuestro sistema, podría conectarse a cualquier televisor del centro que disponga de HDMI y cumplir con la función deseada sin necesidad de hacer uso de un smartphone o similar.

# PROYECTO NUESTRA TABLA PERIÓDICA

- Esquema de conexiones y funcionamiento:



Como podemos ver los botones asociados a cada uno de los elementos de la tabla periódica irán conectados a los diferentes canales de los multiplexores, cuando se active un botón se generará una salida binaria de 4 bits que se identificará con el elemento al que corresponda el botón pulsado.

La salida obtenida tendrá que ser decodificada e interpretada por lo que ese número binario de 4 bits se corresponderá con el número asociado a cada elemento y de esta forma podremos saber cuál estamos escogiendo.



# PROYECTO NUESTRA TABLA PERIÓDICA

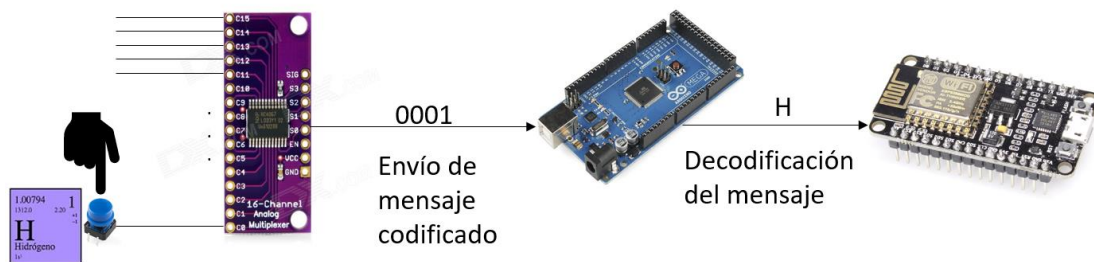
Ejemplo:

0001 -> 1 -> Hidrogeno (H)

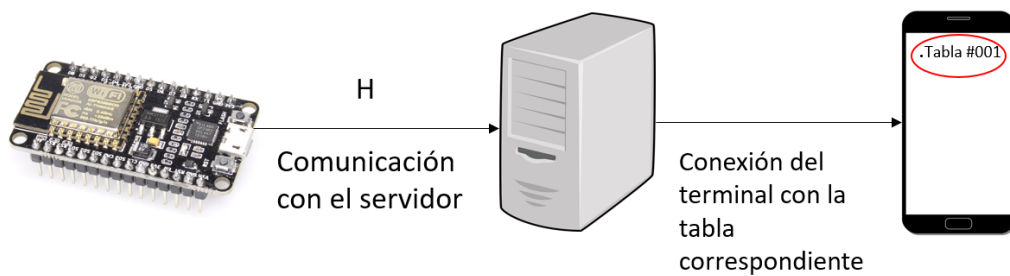
1010 -> 10 -> Neón (Ne)

1111 -> 15 -> Fósforo (P)

Este mapeado se realizará con cada uno de los elementos de la tabla periódica.



Una vez tenemos la salida decodificada e interpretada el Arduino se comunicará con el ESP8266 que mediante un mensaje solicitará el video del elemento elegido al servidor web.

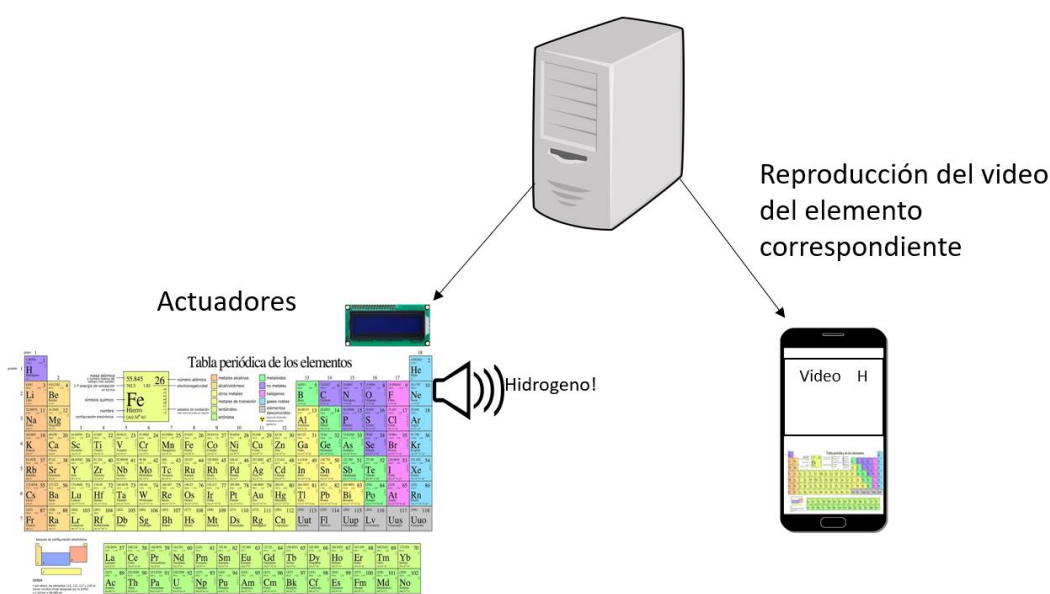


Todos los videos estarán guardados en una base de datos y cada uno de ellos estará asociado a un elemento, por lo que si llega una solicitud H se devolverá el video explicativo correspondiente al Hidrogeno.

# PROYECTO NUESTRA TABLA PERIÓDICA

Cuando el servidor web genere la respuesta los actuadores (el altavoz y el LCD) nos proporcionaran la información asociada a ese elemento.

Para poder ver el video necesitaremos tener una conexión con el terminal que ejecute nuestra aplicación (un teléfono móvil, Tablet, Raspberry Pi, o cualquier dispositivo que tenga Android y sea capaz de ejecutar la aplicación) por lo que habrá que conectarse con el servidor y una vez se realice dicha conexión podremos visualizar normalmente el video.



En principio la única funcionalidad que vamos a implementar es la de poder visualizar los videos del elemento seleccionado en un terminal remoto, en caso de que de tiempo nos gustaría añadir algunas ideas relacionadas con el mundo de la química, como por ejemplo un cuestionario interactivo con preguntas relacionadas con los elementos de la tabla, sus valencias, formulación...

Este proyecto tiene mucha visión de futuro, pero por falta de tiempo no podemos implementar todas las funciones que nos gustaría, ya que tenemos mas proyectos y asignaturas a parte de esta.



# PROYECTO NUESTRA TABLA PERIÓDICA

## • Análisis de la competencia:

Al ser un proyecto escolar pensado para un uso docente y presentarse en un concurso como una idea innovadora, no hemos encontrado ningún producto que sea similar.

Al no contar con una referencia real sobre el coste de dicho sistema presentamos a continuación una estimación del presupuesto necesitado para llevar a cabo dicho proyecto.

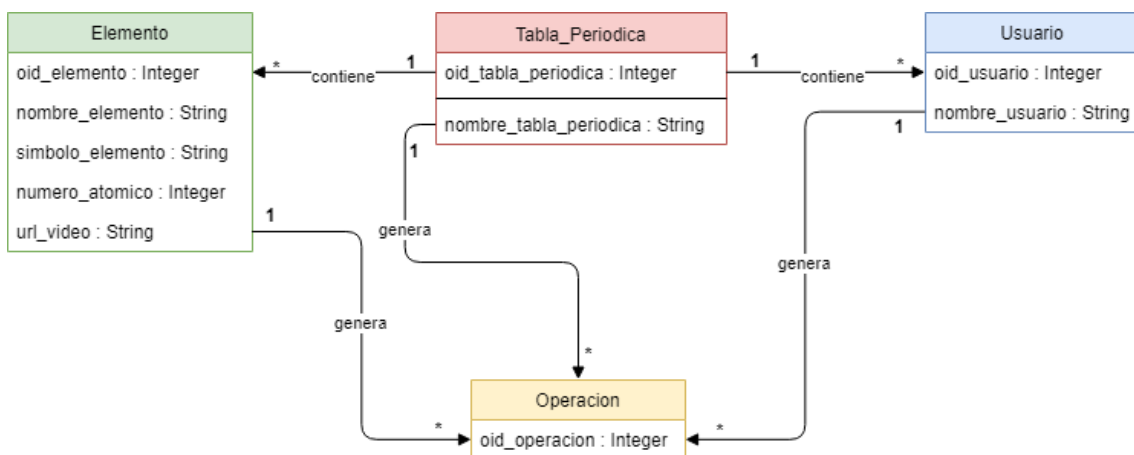
## • Presupuesto:

1. Placa Arduino -> 14€ ([link del producto](#)) .
2. Placa ESP8266 -> 7'50€ ([link del producto](#)) .
3. 9 x Multiplexores CD74HC4067 -> 18'45€ ([link del producto](#)) .
4. Pantalla LCD -> 10€ ([link del producto](#)) .
5. Altavoz -> 7€ ([link del producto](#)) .
6. 5 x Packs de 25 pulsadores -> 52,50€ ([link del producto](#)) .
7. 4 x Rollo cable 40x1m -> 32€ ([link del producto](#)) .
8. 2 x Pack de 3 protoboas -> 18'60€ ([link del producto](#)) .
9. Raspberry Pi 3 Model B+ -> 37'50€ ([link del producto](#)) .
10. Tarjeta SD 16 o 32 GB para Raspberry -> 4€ ([link del producto](#)) o 6€ ([link del producto](#)) .

# PROYECTO NUESTRA TABLA PERIÓDICA

## • Implementación Base de Datos:

Para realizar la implementación de la base de datos necesaria para nuestro proyecto hemos diseñado el siguiente diagrama UML a partir del cual se ha construido posteriormente la base de datos ya mencionada.



Con motivo de que pudieran implementarse físicamente más de un sistema hemos optado por declarar una entidad “Tabla\_Periodica” que contenga múltiples usuarios y elementos, de esta forma cada uno de los usuarios a la hora del registro podrá elegir el sistema físico con el que interactuar.

Una vez aclarado esto pasaremos a describir cada una de las entidades planteadas, así como la relación que guardan entre ellas.

### Tabla\_Períodica:

Es la base de toda la estructura planteada, representa cada uno de los sistemas que pudieran llegar a implementarse físicamente.

Guarda una relación 1-n con las entidades “Usuario”, “Elemento” y “Operación”, ya que una tabla periódica contiene varios usuarios, elementos y operaciones.

# PROYECTO NUESTRA TABLA PERIÓDICA

## Usuario:

Referencia a cada uno de los usuarios que se registren mediante la aplicación web, estos estarán asociados a una única tabla periódica que se concretará en el momento del registro.

Guarda una relación 1-n con la entidad Operación, puesto que un usuario puede generar varias operaciones seleccionando múltiples elementos distintos pertenecientes a una tabla periódica.

## Elemento:

Es la entidad que referencia a cada uno de los elementos correspondientes a la tabla periódica, cada uno de ellos tendrá asignado un botón físico con el que el usuario interactuará para generar las distintas operaciones.

Tienen una relación 1-n con la entidad Operación ya que un usuario al seleccionar un elemento se pueden generar múltiples operaciones, asociadas tanto al usuario como al elemento.

## Operación:

Es la entidad generada cuando un usuario asociado a una tabla periódica acciona uno de los botones correspondientes a cada uno de los elementos. De esta forma podremos controlar todas las interacciones que realice un usuario en una tabla y sobre los elementos que las realiza.

# PROYECTO NUESTRA TABLA PERIÓDICA

Ahora que comprendemos la estructura planteada en nuestra base de datos pasaremos a profundizar en cada una de las tablas implementadas.

Tabla_Periodica	
PK	oid_tabla_periodica
	nombre_tabla_periodica

“Tabla\_Periodica” consta de una primary key de tipo Integer “oid\_tabla\_periodica” que se genera y autoincrementa automáticamente con cada tabla que se crea y nos sirve para identificar cada una de las tablas periódicas.

A parte hemos considerado oportuno introducir un parámetro de tipo String “nombre\_tabla\_periodica” que al igual que el “oid\_tabla\_periodica” sirve para identificar cada una de las tablas periódicas, pero este se ha creado con la intención de dar facilidades a nivel de usuario asignándole un nombre a cada una de ellas.

Usuario	
PK	oid_usuario
FK1	nombre_tabla_periodica
	nombre_usuario

“Usuario” al igual que “Tabla\_Periodica” tiene su primary key de tipo Integer autogenerada y autoincrementada que sirve para identificar al usuario.

Como hemos visto anteriormente en el diagrama UML “Tabla\_Periodica” y “Usuario” tienen una relación 1-n por lo que será necesario una foreign key “nombre\_tabla\_periodica” que servirá para asociar al usuario con la tabla periódica correspondiente.

Por último, constara de un parámetro “nombre\_usuario” que será utilizado para que el propio usuario se registre con el nombre que guste en la aplicación web.

# PROYECTO NUESTRA TABLA PERIÓDICA

Elemento	
PK	oid_elemento
FK1	nombre_tabla
	nombre_elemento
	simbolo_elemento
	numero_atomico
	url_video

“Elemento” como las dos tablas anteriores consta de un identificador de tipo Integer con las mismas características que poseen los de las tablas anteriores “oid\_elemento”.

Al igual que la tabla “Usuario” esta posee una relación 1-n con “Tabla\_Periodica” por tanto será necesaria una foreign key de tipo String “nombre\_tabla\_periodica” para identificar a la tabla periodica a la que pertenece el elemento.

Todos los demás parámetros son las características de cada elemento como pueden ser “nombre\_elemento” (nombre de cada uno - Hidrogeno), “símbolo\_elemento” (símbolo correspondiente al elemento - H ), el número atómico (número de protones que posee cada átomo del elemento - 1) y “url\_video” (sirve para identificar el video explicativo del elemento correspondiente). Todos estos son únicos y no pueden repetirse.

# PROYECTO NUESTRA TABLA PERIÓDICA

Operacion	
PK	oid_operacion
FK1	nombre_tabla
FK2	nombre_usuario
FK3	nombre_elemento

Y por último la tabla “Operación” estará compuesta como todas las tablas anteriores de una primary key de tipo Integer “oid\_operacion” que tendrá la misma función y características que las anteriores.

Al tener una relación 1-n con cada una de las tablas anteriores (“Tabla\_Periodica”, “Usuario”, “Elemento”) esta vez serán necesarias 3 foreing keys de tipo String que identifiquen respectivamente a cada una de las tablas referenciadas.



# PROYECTO NUESTRA TABLA PERIÓDICA

## • Servicio REST:

El servicio Rest es un servidor en el que existen una serie de métodos implementados que nos ayudarán a mostrar la información contenida en la base de datos.

Inicialmente, en el servicio Rest se realiza la conexión con la base de datos, mediante el uso de la sentencia `mySQLClient` que contendrá el verticle desplegado y una serie de configuraciones, entre las cuales se encuentra el host, que en este caso sería `localhost`, un usuario, una contraseña, el nombre de la base de datos y el puerto en el que escucha.

### Métodos GET:

#### 1. `handlerGetTablaPeriodica();`

Éste método es el encargado de tomar toda la información de la base de datos referente a todas las tablas “TablaPeriodica”, donde en ella se encuentra un identificador “`oid_tabla_periodica`”, de la propia tabla que será un tipo `INT` y de manera autoincrementar por ser la PK de la tabla. Y “`nombre_tabla_periodica`”, que como se explicó anteriormente, sirve para que el usuario tome la decisión de escoger la tabla física en la que quiere interactuar, siendo ésta de tipo `String` y un valor único, es decir, que no se pueda repetir. Para ello creamos una lista vacía de tipo `TablaPeriodica(Entitie)`, se recorrerá mediante un `for` incluyendo todos los `Json` que se reciben. Además, hemos implementado una serie de sentencias que gestionan los errores cometidos, así como el cierre de conexión tras obtener el resultado.

#### 2. `handlerGetTablaPeriodicaNombre();`

Éste método es un extendido del anterior, es decir, realiza exactamente lo mismo que `handlerGetTablaPeriodica()`, pero aquí aplicamos un filtro que consiste en mostrar toda la información de una sola tabla periódica correspondiente al “`nombre_tabla_periodica`” que especificamos en la URL.

#### 3. `handlerGetTodosUsuarios();`

# PROYECTO NUESTRA TABLA PERIÓDICA

Este GET toma toda la información acerca de todos los usuarios desde la tabla “Usuarios”, donde se encuentra un “oid\_usuario” de tipo INT y de manera autoincremental por ser la PK de esta tabla. También un “nombre\_usuario” de tipo String y único, mediante el cual un usuario se registrará en la aplicación web con ese nombre y se guardará en la base de datos. Además, hay un “nombre\_tabla\_periodica” de tipo String y único, que se especifica el nombre de la tabla física con la que se interactuará y se elegirá también en la aplicación web.

Como explicamos en el primer GET, todos realizan lo mismo, una lista que se recorre y se guarda los Json que se reciben.

#### 4. `handlerGetUsuarioNombre();`

Este es el extendido de `handlerGetTodosUsuario()`, en el que se le aplica el filtro por un “nombre\_usuario” especificado en la URL, y mostrará la información de un solo usuario registrado.

#### 5. `handlerGetTodosElementos();`

Este método GET toma toda la información relacionada con todos los elementos contenidos de una tabla periódica, es decir, mostrará información de todos los elementos. Contiene un “oid\_elemento” de tipo INT y de manera autoincremental al ser la PK de esta tabla. También un “nombre\_elemento” de tipo String, que hará referencia al nombre del elemento que busquemos, un “símbolo\_elemento” de tipo String que hace referencia a su símbolo, un “numero\_atómico” de tipo INT que tiene el valor del peso atómico del elemento, una “url\_video” de tipo String que será una url del video que se mostrará por pantalla al elegir un elemento específico y un “nombre\_tabla\_periodica” de tipo String que hará referencia al nombre de la tabla física.

El procedimiento es el mismo que los anteriores.

#### 6. `handlerGetElementoNombre();`

Este método realiza filtro de los elementos pasándole un “nombre\_elemento” a la URL, así escogerá solo el elemento solicitado.

# PROYECTO NUESTRA TABLA PERIÓDICA

## 7. `handlerGetTodasOperaciones();`

Este método recoge información sobre todas las operaciones realizadas en la tabla periódica, es decir, sobre los elementos pulsados en la tabla. Contiene un “oid\_operacion” de tipo INT y de manera autoincremental por ser PK de la tabla. También un “nombre\_tabla\_periódica” de tipo String, que como hemos explicado antes, sirve para referenciar la tabla física. También un “nombre\_elemento” de tipo String que será el elemento que solicitemos al pulsar un botón específico. Y por último un “nombre\_usuario” de tipo String que se registrará quien ha realizado dicha operación.

El procedimiento es el mismo que los anteriores.

## 8. `handlerGetOperacionesNombreUsuario();`

Mediante éste método, recogemos la información de una sola operación, pasándole como filtro un “nombre\_usuario” en la URL, para determinar las operaciones que ha realizado un determinado usuario.

## Métodos PUT/POST:

## 9. `handlerInsertTablaPeriodica();`

Método implementado para realizar los inserts de la tabla “Tabla\_Periodica” en la base de datos con el correspondiente cuerpo:

```
1  {  
2      "nombre_tabla_periodica" : "ejemplo"  
3  }
```

## 10. `handlerInsertUsuario();`

# PROYECTO NUESTRA TABLA PERIÓDICA

Método implementado para realizar los inserts de la tabla "Usuario" en la base de datos con el correspondiente cuerpo:

```
1 {  
2     "nombre_usuario" : "juan" ,  
3     "nombre_tabla_periodica" : "tabla01"  
4 }
```

## 11.handlerInsertElemento();

Método implementado para realizar los inserts de la tabla "Elemento" en la base de datos con el correspondiente cuerpo:

```
1 {  
2     "nombre_elemento" : "Hidrogeno" ,  
3     "simbolo_elemento" : "H" ,  
4     "numero_atómico" : 1 ,  
5     "url_video" : "hidrogeno.avi" ,  
6     "nombre_tabla_periodica" : "tabla01"  
7 }
```

## 12. handlerInsertOperacion();

Método implementado para realizar los inserts de la tabla "Operacion" en la base de datos con el correspondiente cuerpo:

```
1 {  
2     "nombre_tabla_periodica" : "tabla01",  
3     "nombre_usuario" : "juan" ,  
4     "nombre_elemento" : "Hidrogeno"  
5 }
```