

|  |
| --- |
|  |
| Software para Robots  Grupo B3 |
| 19 diciembre  Integrantes: Chen Xin Pan Wang UO276967  Mateo Rico Iglesias UO277172  Eduardo Blanco Bielsa UO285176 |
|  |

Práctica 11

(ThingSpeak11.2) Hola Mundo en ThingSpeak (0,5 Punto) ✅

Proceso:

(Se ha trabajado con PLatform.io en VSCode)

* Primero se buscó información sobre cómo conectar el LOLIN D32 por WIFI -> <https://www.upesy.com/blogs/tutorials/how-to-connect-wifi-acces-point-with-esp32>
* Después se intentó conectar el Arduino al d32 para hacer que el Arduino obtuviese los datos del ldr y enviárselos al d32, pero después de profundas búsquedas, se averiguó que no comparten los mismos voltajes (5 y 3.3V respectivamente). Por tanto, se necesitaría un conversor entre medias. NOTA: *se encontró una supuesta manera de conectarlo, que es conectar ARDUINO—RX -> D32---TX, ARDUINO---TX -> D32---RX, D32 --- GND (pero no funcionaba).*
* Después de varias comeduras de cabeza, se nos ocurrió mirar si el d32 tenía pines analógicos (y así realizar él todo el trabajo), y si tenía. Por tanto, hicimos que el d32 obtuviese directamente los datos y los enviase a la api de ThingSpeak, obteniendo dos gráficos: la temperatura (*lo que se pedía*), y el tráfico de red (*para comprobar que el d32 seguía funcionando correctamente*).

Captura de pantalla de un celular

Descripción generada automáticamente

*(gráfica real tomada)*

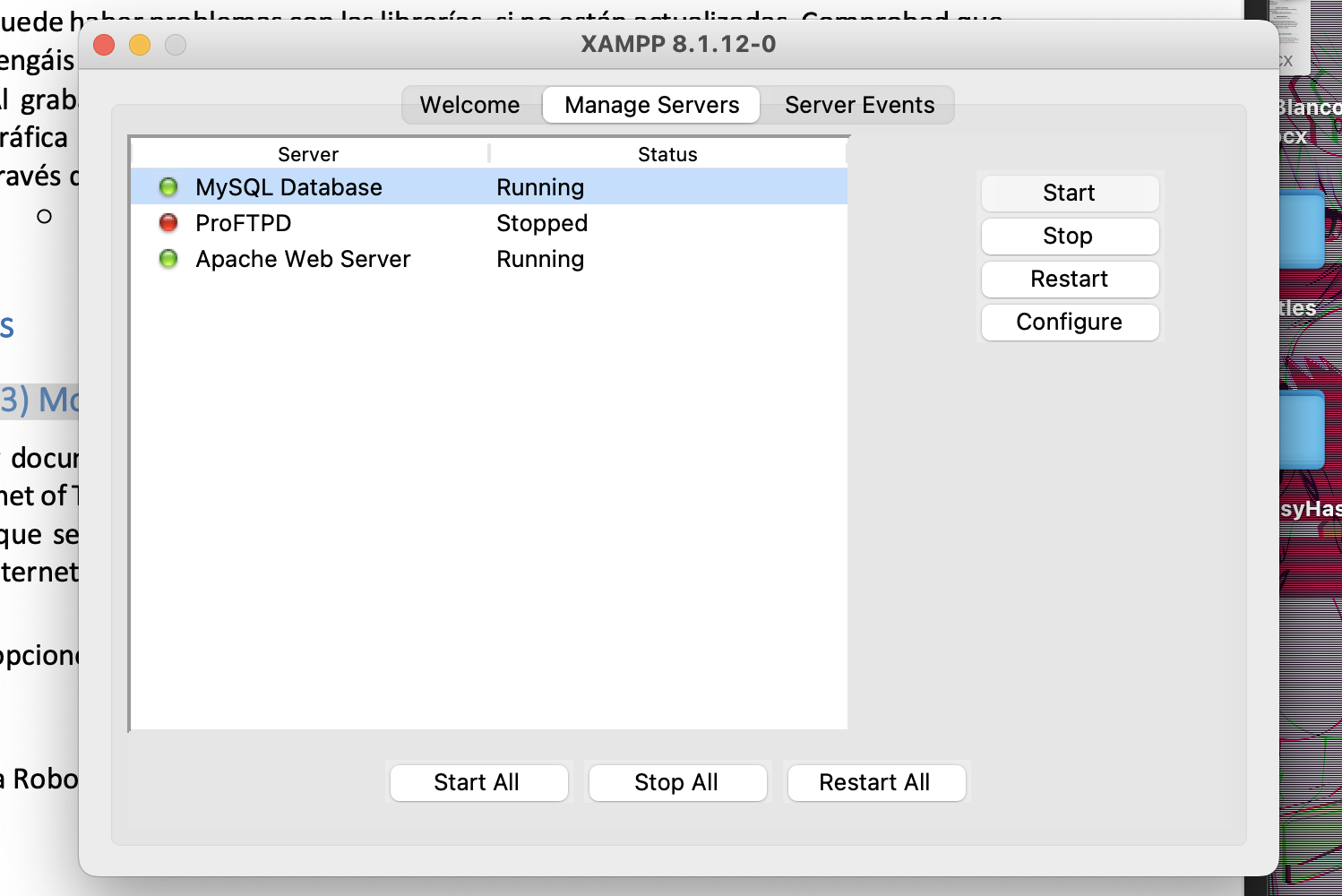
* Para la funcionalidad de los Talkbacks, se ha configurado un led blanco externo al d32, conectado a un pin digital que lo activa, creando una petición POST a la API de ThingSpeak (en la que creamos un array de comandos de tipo TURN\_ON y TURN\_OFF). En función del comando que sea (pues se va analizando la cola de comandos), se enciende o se apaga el led (con un cooldown de 20 segundos para no sobrecargar la API)
* Se cogió de la API de Mathworks una función POST estándar para crear las peticiones

(IoTServer11.3) Montar Servidor IoT (1,5 puntos) ✅

Hecho por Eduardo Blanco Bielsa

Nota: no se incluye vídeo de este ejercicio, pues sería muy extenso. A cambio, se ofrece una documentación detallada.

* Primero averigüé si se podía conectar el dht 11 a 3.3V. Lo comprobé en esta página y vi que sí se podía -> <http://www.esp32learning.com/code/esp32-and-dht11-sensor-example.php>
* Después revisé un proyecto que encontré por Internet que me sirvió como base e inspiración para crear el servidor de IOT -> <https://randomnerdtutorials.com/esp32-web-server-arduino-ide/>
* Decidí cambiar de idea, y en vez de accionar dos leds, mostrar la humedad y la temperatura del ambiente (así era más original).
* Para la base de datos, tuve que utilizar XAMPP - > <https://www.youtube.com/watch?v=fsQoA7NMhsU> y <https://microcontrollerslab.com/esp32-mysql-database-php/>
* Lo descargué y cree una base de datos MYSql y un servidor apache



* Luego cree la base de datos dentro del panel de control de php:

Captura de pantalla de un celular

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

* Cree una tabla para los valores tomados por el dht 11

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente

* Creé un script en php para conectar los datos del Arduino a la base de datos -> <https://microcontrollerslab.com/esp32-mysql-database-php/>

Texto

Descripción generada automáticamente

* Probé que el script funcionase:

Captura de pantalla de computadora

Descripción generada automáticamente *Todo correcto*

* Me surgió un error de permisos:

Texto

Descripción generada automáticamente

* Para solucionarlo, tuve que crear un usuario en MariaDB con todos los privilegios:

Texto

Descripción generada automáticamente

* Se buguea bastante en mac, estuve bastante rato buscando una solución para relanzar los Daemon httpd y encontré esto -> <https://stackoverflow.com/questions/4582504/xampp-apache-webserver-localhost-not-working-on-mac-os>
* Conseguí que funcionase, cambiando la uri de la petición que le enviamos a la base de datos (introduciendo la ip de mi mac (pues la base de datos es local))
* He aquí unas imágenes del diseño:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente*Detector a tiempo real*

* He aquí la base de datos (el d32 escribe la temperatura, la humedad, y la fecha de inserción del dato (que inserta cada segundo)):

Imagen que contiene Calendario

Descripción generada automáticamente

Anotaciones:

Toda actividad que lleve un ✅ es porque ha sido entregada correctamente.

Si se desea consultar, dejamos el enlace del correspondiente repositorio en github a través del cual fuimos trabajando durante el transcurso de la asignatura:

<https://github.com/gitblanc/SR>



