

# Taller de proyecto II

Trabajo Practico I

Año 2018

Monti, Agustín n 123/9

## Ejercicio 1

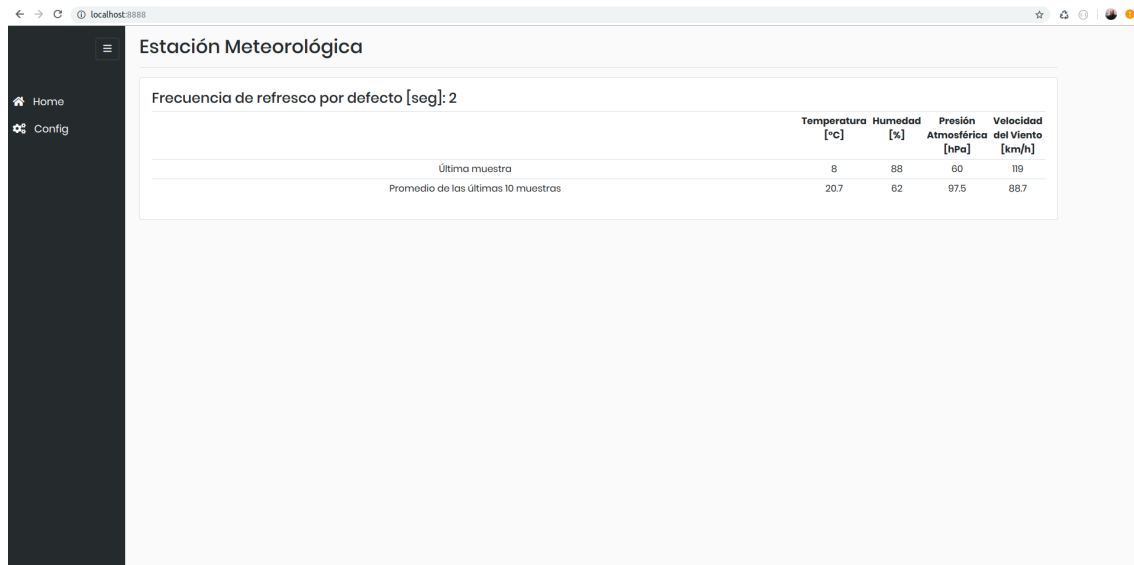
Para la simulación del proyecto se realiza un fork del siguiente repositorio [https://github.com/gmaron/tp2\\_template](https://github.com/gmaron/tp2_template).

Se utiliza un archivo Python para la generación de las muestras denominado *"process"* en el que de manera indefinida se crean datos pertenecientes a cada uno de los sensores. Dichas muestras son generadas cada 1 segundo. Este proceso es controlado por otro archivo auxiliar *"aux\_pro"*. Este último es el encargado de iniciar o detener al proceso generador. Por otro lado, el archivo *"app"* es el encargado de hacer de servidor manejando las peticiones del cliente. Ambos hacen consultas y escrituras sobre una base de datos *MySQL* mediante *phpMyAdmin*. El control de la misma se realiza en el archivo *"database"*. Básicamente se utiliza un *singleton* para asegurarse una sola instancia y tres métodos que permiten obtener la sesión, obtener las últimas diez muestras y almacenar nueva. El proceso generador utiliza el ultimo método simulando ser los sensores y el que hace de servidor consulta sobre las ultimas diez.

Como esquema flujo general del sistema se puede decir que la estación meteorológica comienza a obtener muestras cuando se ingresa a la página. En ese momento el archivo *"app"* recibe una petición del tipo GET y da inicio al proceso generador que continuamente crea una muestra del sensor de temperatura, humedad, presión y viento almacenándolos en la tabla *"samples"*. Del lado del cliente se está ejecutando un retardo de 2 segundos para refrescar la vista (valor por default al cargar la página, el usuario puede modificarlo) y obtener nuevas muestras de la DB.

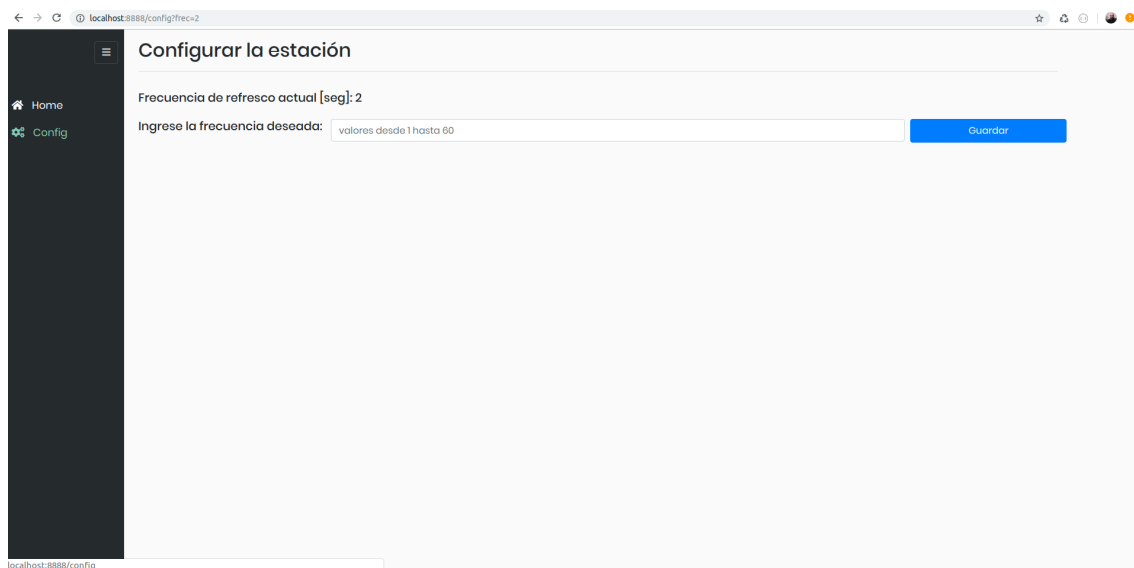
Dentro de la interacción del usuario podemos decir que se encuentran dos vistas. El acceso inicial a la página (*"Home"*) donde se visualizan las muestras de los sensores y la vista que permite cambiar el periodo de refresco llamada *"Config"*. Para agregar usabilidad se diseñó una *side-bar* que permite acceder a la visualización o a la configuración de manera sencilla. Este elemento tiene un botón que permite colapsarla de manera que en pantallas pequeñas no moleste la visual ocupando menor porcentaje de pantalla. Toda la visualización está hecha con *Bootstrap 4*.

Al acceder a la página por primera vez tenemos la visualización de las muestras y la frecuencia de refresco.



|                                     | Temperatura<br>[°C] | Humedad<br>[%] | Presión<br>Atmosférica<br>[hPa] | Velocidad<br>del Viento<br>[km/h] |
|-------------------------------------|---------------------|----------------|---------------------------------|-----------------------------------|
| Última muestra                      | 8                   | 88             | 60                              | 119                               |
| Promedio de las últimas 10 muestras | 20.7                | 62             | 97.5                            | 88.7                              |

Al hacer click en “*Config*” de la *side-bar* podemos editar la frecuencia de refresco de la página:



Frecuencia de refresco actual [seg]: 2

Ingrese la frecuencia deseada:  Guardar

Al volver a hacer click en “*Home*” de la *side-bar* volvemos a visualizar las muestras pero con la nueva frecuencia elegida:

|                                     | Temperatura<br>[°C] | Humedad<br>[%] | Presión<br>Atmosférica<br>[hPa] | Velocidad<br>del Viento<br>[km/h] |
|-------------------------------------|---------------------|----------------|---------------------------------|-----------------------------------|
| Última muestra                      | 18                  | 55             | 106                             | 192                               |
| Promedio de las últimas 10 muestras | 21.3                | 43.7           | 123.1                           | 122.9                             |

## Ejercicio 2

Los principales problemas de concurrencia surgen cuando se realizan operaciones de escritura y lectura en la base de datos. La manera de evitar este problema fue usar un singleton. Por otro podría suceder que se quiera leer muestras a una velocidad mayor a la que los sensores las producen. Esto se evitó limitando el mínimo del refresco de la página web a 1 y asegurándose que el proceso que genera muestras escriba primero en la DB.

Otra problemática que no se tuvo en cuenta es el no uso de un sistema operativo de tiempo real (*RTOS*), suponiendo que los periodos en que toman las muestras los sensores no son críticas. Caso contrario habría que optar por alguna de las opciones que encontramos en *RTOS*.

## Ejercicio 3

En sistemas de tiempo real los sensores trabajan bajo algún protocolo (si son muy económicos puede ser uno no estándar). En nuestro caso genera las muestras y automáticamente las guarda en la DB. Podría darse el caso en que estos sensores no estén escribiendo en la base por problemas de internet, si la base en el cloud, por ejemplo.

En el sistema real existirá mayor precisión, pero con cierto rango de error en la toma de valores.

Los sensores utilizados en una Estación Meteorológica presentan cierto tiempo de vida y su funcionamiento puede ser cada vez peor, generando errores a largo plazo.

En un sistema real de este tipo, cuando se toman valores con sensores, es necesario que exista cierto *delay* entre la toma de estos ya que los sensores no funcionan de forma inmediata.