Taller de Proyecto II

Practica 1

Facultad de Informática

Universidad Nacional de La Plata

Integrantes:
Cáceres, Elías 1124/2
Pereira, Luciano 1142/4

Proceso de adquisición de datos:

La implementación de la aplicación se realizó utilizando el framework Flask, el cual está hecho con el lenguaje Python. El proceso de la adquisición de los datos comienza en el acceso a la base de datos, donde están almacenadas las muestras climáticas, para esto definimos en el archivo database.py el método get_10_last_samples, que se encarga de devolver las últimas 10 muestras de los datos, es decir, las 10 más recientes. En caso de no haber aún 10 muestras, el método devuelve la cantidad existente en ese momento.
Este método nos permite por lo tanto, cumplir con las funcionalidades correspondientes a mostrar tanto la última muestra como el promedio de las 10 últimas muestras.
La adquisición de datos continúa con el script app.py, en el cual definimos la ruta "/" donde se definen todos los métodos necesarios para mostrar los datos requeridos. Las acciones que realiza éste script son las siguientes:

- Iniciar el script process.py, que se encarga de crear las muestras a partir de la generación de números aleatorios
- Utiliza el método "show_all" que llama a la función get_10_last_samples, y una vez obtenidas las muestras, calcula su promedio y lo retorna al index junto con la muestra más reciente

Ejecución de los procesos:

Process.py:

Este script contiene la función main, que se encarga de generar los valores de las variables climáticas (a través de la función getrand), crear la muestra, como instancia de la clase Samples, y finalmente guardar la muestra en la base de datos. Este proceso se repite cada 1 segundo.

App.py:

Este script es la conexión entre todos los scripts (process.py, models.py, aux_pro.py, database.py) y el front-end de la aplicación. Llama a las funciones de estos scripts en base a los request del usuario y sirve a las paginas html, de forma que los datos estén disponibles para el usuario.

Esquema General:

La app consiste de:

- Una base de datos con una tabla llamada Samples, que contiene las distintas variables climáticas (temperatura, presión, humedad y velocidad del viento)

- Un script llamado process.py que crea una muestra por segundo, a partir de la generación de números aleatorios

- Un script llamado app.py que accede a la base de datos y muestra los datos al usuario

- Una página HTML (responsive) que permite al usuario ver los datos de forma ordenada a través de un navegador web

Interacción del usuario:

La aplicación constará con una única página web, la cual comenzará el proceso de muestreo de datos automáticamente al ser accedida por un usuario, y se actualizará con una frecuencia por defecto de 3 segundos. En esta página el usuario podrá:

- Ver la muestra más reciente

- Ver el promedio de las últimas 10 muestras

- Seleccionar la frecuencia con la que la página actualizará los valores de las muestras, a partir de ciertos valores predefinidos (1, 2, 5, 10, 30, 60 segundos)

2)

Una problemática consiste en que la aplicación debe guardar muestras en la base de datos con cierta frecuencia y a la vez debe acceder a las muestras para mostrárselas al usuario, Se debe tener en cuenta además, la posibilidad del usuario de elegir la frecuencia de muestreo de hasta 1 segundo, pudiendo esto provocar una gran carga de accesos a la base de dato, lo cual a su vez puede provocar que alguno de estos accesos falle, generando un error en la aplicación.

En el ambiente real se tendría una sola base de datos, y la posibilidad de múltiples usuarios queriendo acceder a la misma, para visualizar los datos, por lo que el problema antes mencionado sería potenciado de forma que la chance de saturación de la base de datos sería mayor.

Una problemática general de tiempo real consiste en que el usuario puede querer ver muestras cuando todavía la placa no generó ninguna. En ese caso la aplicación trataría de acceder a datos que no están en la base de datos, provocando un error. Esto fue tenido en cuenta en la implementación de la aplicación, por lo que no genera problemas.

3)

En el sistema real, la información sobre las variables provendrían de un sistema externo, esto implica una adición en el procesamiento necesario a realizar y genera otra fuente de errores o problemas posibles.
Además se tiene que tener en cuenta el tiempo extra que puede llevar el proceso que va desde la obtención de las muestras por los sensores, hasta el almacenamiento en la base de datos. Esto es importante ya que los usuarios podrían tener la posibilidad de elegir una frecuencia de actualización muy alta, que supere este tiempo de procesamiento mencionado anteriormente, lo que provocaría una experiencia de usuario no satisfactoria.