[Escuela de código]

Proyecto del curso IoT

Objetivo

El objetivo de este proyecto es que puedan poner en práctica los conocimientos vistos del curso realizando una serie de modificaciones al código que se construyó o elaboró durante las clases.

Utilice todas las herramientas a su disposición (terminal, MQTTExplorer, debugger) para ensayar y testear el funcionamiento de su implementación. En caso que tenga problemas, consulte y continúe explorando. Lo más rico de estos ejercicios es que pueda analizar las fallas y aprender de ellas por su cuenta como todo un buen detective.

Punto de partida

Requerimientos previos

Cómo punto de partida para su proyecto utilizará el ejemplo de aplicación “iotbridge” que se terminó de elaborar durante la clase de contenedores. Los requerimientos que se detallan a continuación ya los cumple y realiza la aplicación elaborada en clase, usted deberá verificar que su aplicación siga cumpliendo estos requerimientos:

- Todos los tópicos actuadores utilizados en el curso funcionen. En el menú de actuadores del dashboard los actuadores en el mock (simulador drone) deberán responder a las acciones tomadas en el dashboard.
- Qué los tópicos de sensores utilizados en el curso funcionen. En el menú de sensores del dashboard se deberá ver reflejado los valores de los sensores del mock (simulador drone).
- Qué los tópicos y mensajes de monitoreo que envía prometheus exporter lleguen al menú de estado del dashboard.
- Qué el keepalive llegue al menú de usuarios del dashboard.

Mocks disponibles para el proyecto

Puede utilizar como simuladores el “drone_iot” (que utiliza el celular) o el “drone_mock_iot” (no utiliza el celular). Puede utilizar cualquier de las dos opciones para ensayar su proyecto:

- LINK al drone iot (para celular)
- LINK al drone mock iot (sin usar celular)

Su misión

Su misión, en caso que desee tomarla, es desarrollar e implementar los siguientes requerimientos en el código de iotbridge.

Sensores y análisis de señales

Deberá llevar a cabo el análisis de señales de heading (orientación) para prender y apagar la luz del drone analizando su variación.

- Deberá seguir los pasos indicados en la tarea de la clase de análisis de señales (LINK)

Logging

Todos los logs que utilice para la parte de análisis de señales de heading deben quedar en el código como logging de DEBUG.

- Todos los logs de error deben quedar como error.
- Todos los otros logs los puede utilizar como info o warning
- NO debe quedar en el código final el uso de print, utilizar siempre únicamente logging.

Monitoreo

El dashboard tiene la posibilidad de realizar la siguientes modificaciones sobre su marca GPS en el mapa:

- usted puede cambiar el color del globito / marcador del mapa.
- usted puede modificar la descripción que aparece al presionar su marcador en el mapa.

Para eso, el sistema espera un JSON con la siguiente información:

```
{"color": "gold", "description": "drone en viaje"}
```

- campo color → este campo soporta que indiquemos cuatro colores distintos de marcador:
  - red
  - green
  - blue
  - gold
- campo descripción → aquí podremos colocar la información que deseemos acerca del viaje del vehículo (es texto).

¿Cómo enviar esta información al dashboard?

- Dentro de la función “on_connect_remoto” debe agregar la posibilidad de escuchar por el tópico "config/request", recuerde que el tópico debe tener antes la parte base según el usuario del campus en la variable de config:

```
dashboardiot/<usuario_campus>/config/request
```

- Dentro de la función de “procesar_remoto” deberá analizar si recibe el tópico "config/request", recuerde que antes del tópico está el tópico base que debe descartar:

```
dashboardiot/<usuario_campus>/config/request
```

- Este tópico es enviado por el dashboard una vez por segundo luego de que usted se logea con su usuario del campus en el dashboard.
- Debe tener configurado en el archivo .env el mismo usuario de campus que utiliza para loguearse para que le llegue este mensaje.

En caso de recibir este mensaje usted debe enviar el JSON string con los datos de color y descripción al siguiente tópico:

```
dashboardiot/<usuario_campus>/config/ack
```

Deberá ver en el dashboard como el punto en el GPS cambia de color e informacón según los datos enviados al tópico “config/ack”.

IMPORTANTE: Los datos no se actualizarán en el mapa hasta no recibir un nuevo dato del sensor GPS. Por eso es importante que mientras esté desarrollando esta funcionalidad también tenga encendido el mock para enviar constantemente datos de GPS al dashboard.

Contenedores

El proyecto se entrega en un repositorio, con el código de la aplicación iotbridge modificado (archivo .py). Además el repositorio deberá contener todo lo necesario para poder lanzar el programa en un contenedor:

- En el repositorio deberá tener una carpeta “iotbridge” y colocar allí adentro el código de la aplicación (tal cual visto en la clase de contenedores → main.py, logging.json, requeriements.txt, etc).
- Dentro de la carpeta de la aplicación iotbridge, deberá agregar el archivo de Dockerfile necesario para construir la imagen del sistema.
- Dentro de la carpeta del repositorio deberá incluirse el archivo docker-compose para buildear la aplicación. El docker-compose deberá estar configurado para lanzar correctamente el contenedor de iotbrige (tal cual visto en clase).
- El docker-compose deberá incluir el uso de prometheus exporter (tal cual visto en clase).

A continuación encontrará a modo de referencia el link a la clase de contenedores (LINK).

Cómo presentarlo

Una vez que haya finalizado el proyecto, podrá subir al campus el link de su repositorio en la unidad de proyecto, para que podamos ensayarlo.