HO CHI MINH CITY, UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEER

Application Based Internet of Things Report - LAB 2B

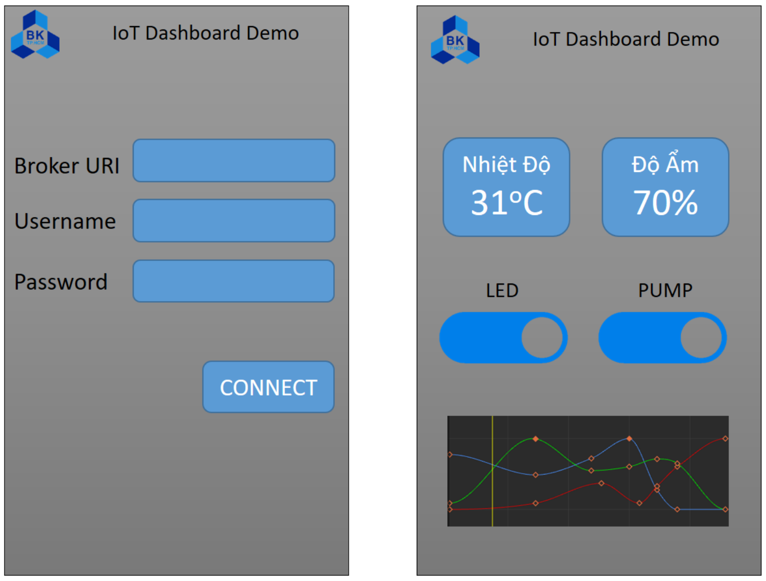
Student: Trương Công Thành
ID: 1814036

HỒ CHÍ MINH CITY

Content

1 Introduction

In this second LAB, students are proposed to create a simple dashboard using Unity 3D editor. Basically, the dashboard has 2 screens (GUIs), as depicted following:



Hình 1: *Software mockup GUI*

The source code of the second LAB is also required to publish in your Github. The details are described in the next section of this report.

2 Requirements

2.1 Screen 1

There are 3 input fields, including the broker of the server, username and password. Using Thingsboard server, the broker should be demo.thingsboard.io, the username is your access token and the password is empty.

The values of these fields can be set by default (in the design phase, by setting the text property of the input component).

When the CONNECT button is pressed, the app will connect to the server. If there is an error, a simple textview can be used to display this error. Otherwise, the second UI is launched.

The testing account for this app **bkiot** and **12345678** for the username and password. The broker URI is **mqttserver.tk**. The default port is **1883**.

2.2 Screen 2

The app needs to subcribe to the following topic

/bkiot/STUDENT_ID/status

in order to receive the current values of sensors (e.g. temperature and humidity) and update these values on textviews. Students can change the information according to their use cases. However, at least 2 different information of the sensors are required.

Two toggle buttons are required to controll two different devices (e.g. a simple LED or a PUMP). When the button is clicked, the data is published to

/bkiot/STUDENT_ID/led

for the LED and

/bkiot/STUDENT_ID/pump

for the PUMP.

The data for each button is a json string, described as follow:

- {"device":"LED","status":"ON"}
- {"device":"PUMP","status":"OFF"}

2.3 Advance UI elements

The end of the second is an example of an advance eleement in Unity3D. It can be a graph, a gause or a map. This part is the extra point in this lab.

3 Report

3.1 Screen 1

Students are proposed to capture the first screen and place it in this report.

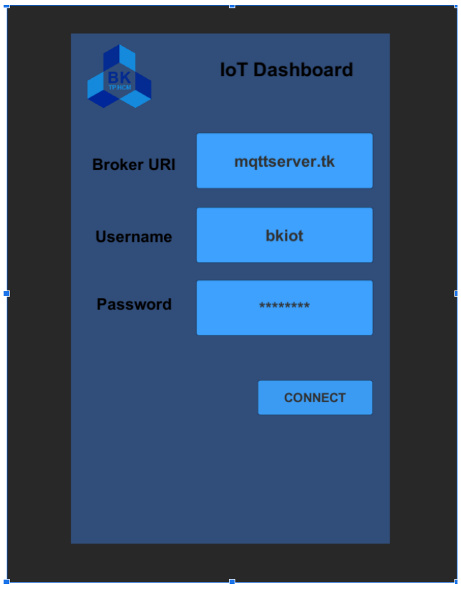
3.2 Screen 2

Students are proposed to capture the second screen and place it in this report.

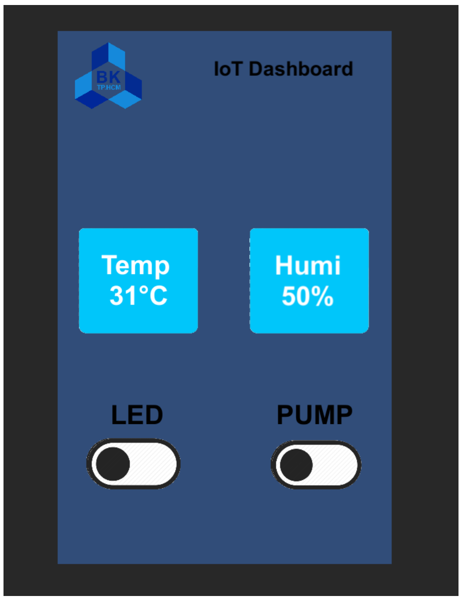
3.3 Github link

The github link of your software is provided here.
Because my git is error. Can i send you driver link now? And i will update git link on driver later. Thank you so much.

`https://drive.google.com/drive/folders/1kX9DbnLw7S2IfEA43rBZL-k-gziCrQwB?usp=sharing`



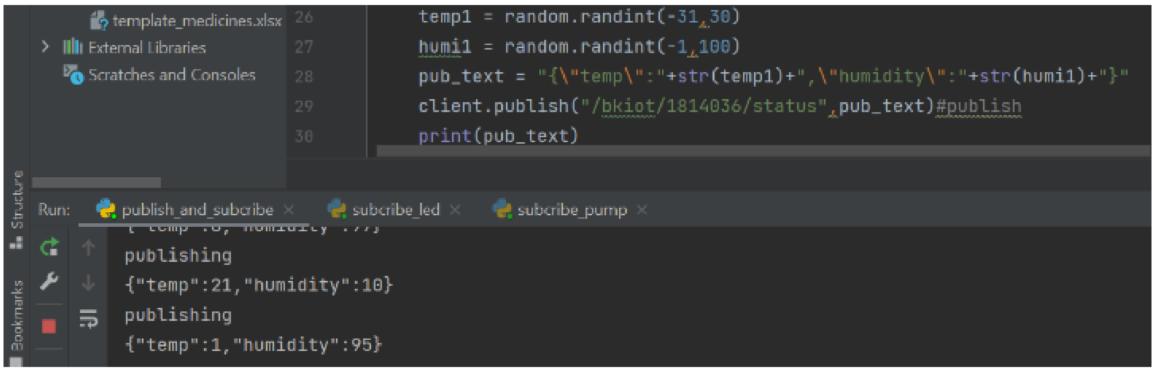
Hình 2: *Screen 1*



Hình 3: *Screen 2*

4 Demo point

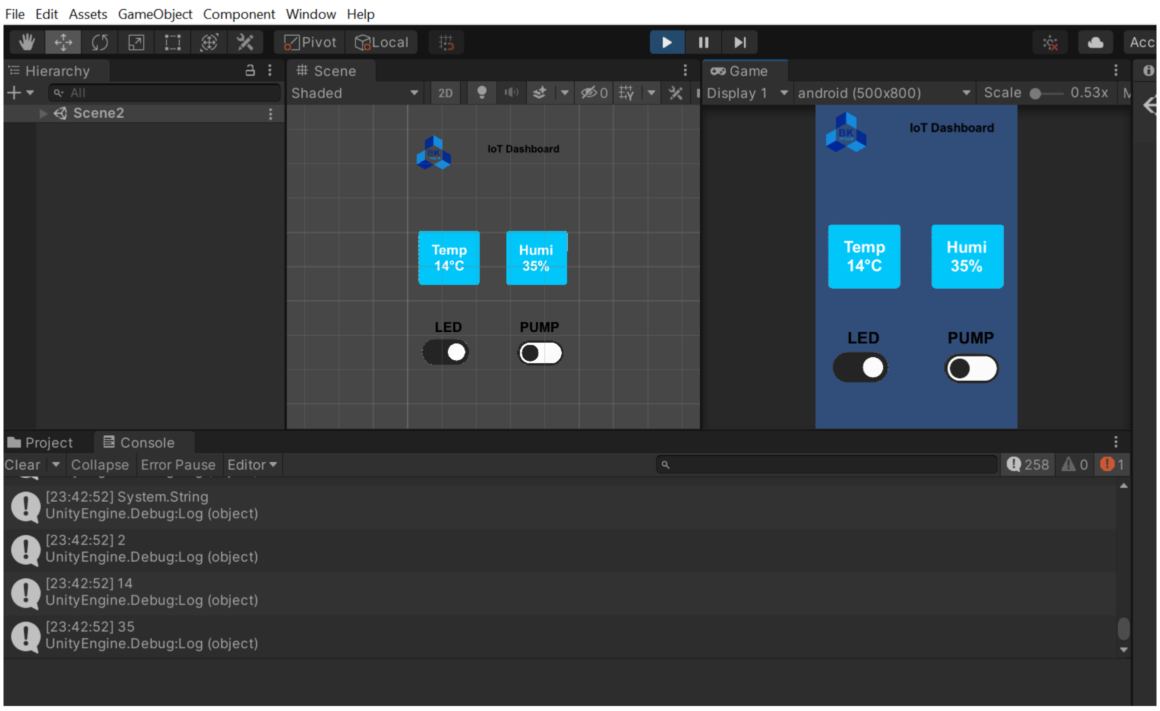
There is a sesson for live demo. A short meeting for each student will be taken place at google meet. The schedule for this meeting will be avaible soon.



```
temp1 = random.randint(-31,30)
humi1 = random.randint(-1,100)
pub_text = "{\"temp\":"+str(temp1)+",\"humidity\":"+str(humi1)+"}"
client.publish("/bkiot/1814036/status",pub_text)#publish
print(pub_text)
```

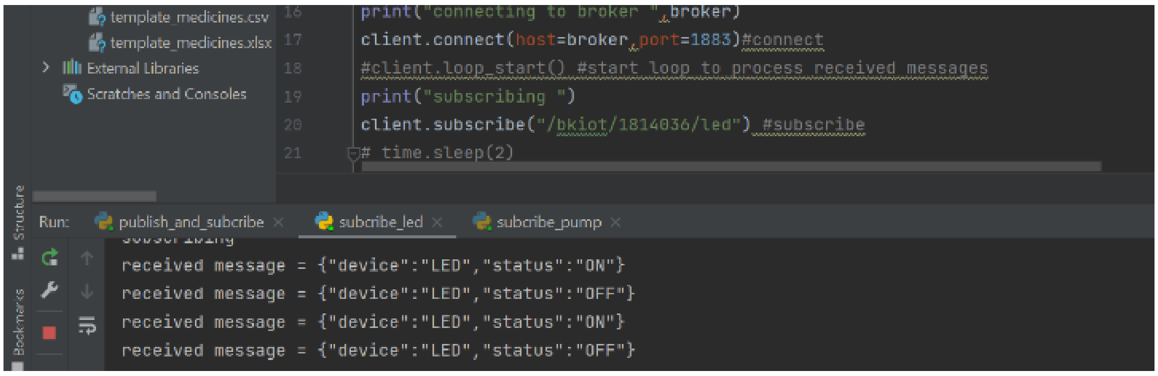
```
publishing
{"temp":21,"humidity":10}
publishing
{"temp":1,"humidity":95}
```

Hình 4: *Publish data on broker server by python*



```
[23:42:52] System.String
UnityEngine.Debug:Log (object)
[23:42:52] 2
UnityEngine.Debug:Log (object)
[23:42:52] 14
UnityEngine.Debug:Log (object)
[23:42:52] 35
UnityEngine.Debug:Log (object)
```

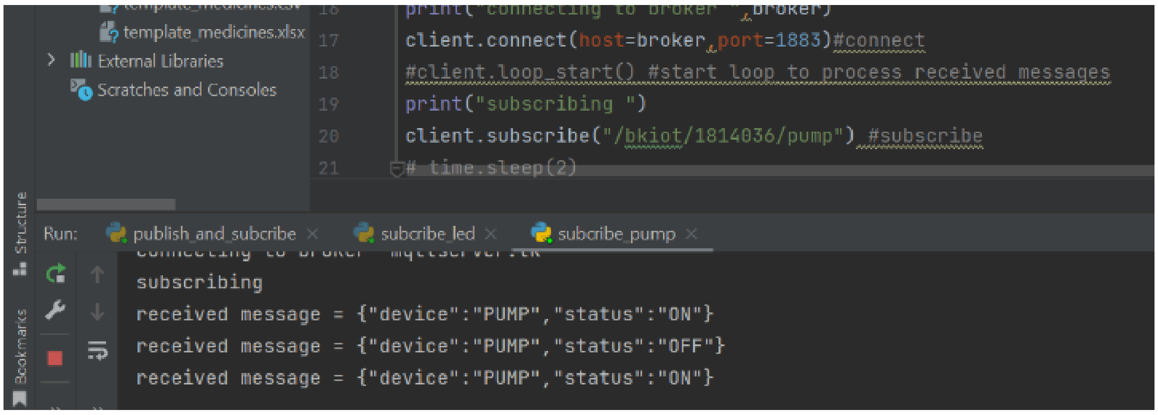
Hình 5: *Subcribe to receive temp and humidity values, also publish led and pump status to broker*



```
print("connecting to broker ",broker)
client.connect(host=broker,port=1883)#connect
#client.loop_start() #start loop to process received messages
print("subscribing ")
client.subscribe("/bkiot/1814036/led") #subscribe
# time.sleep(2)
```

```
received message = {"device":"LED","status":"ON"}
received message = {"device":"LED","status":"OFF"}
received message = {"device":"LED","status":"ON"}
received message = {"device":"LED","status":"OFF"}
```

Hình 6: *Subcribe to receive led status*



```
print("connecting to broker ",broker)
client.connect(host=broker,port=1883)#connect
#client.loop_start() #start loop to process received messages
print("subscribing ")
client.subscribe("/bkiot/1814036/pump") #subscribe
# time.sleep(2)
```

```
subscribing
received message = {"device":"PUMP","status":"ON"}
received message = {"device":"PUMP","status":"OFF"}
received message = {"device":"PUMP","status":"ON"}
```

Hình 7: *Subcribe to receive pump status*