Practical 5 - Working with Firebase

Step 1: Register an account at https://firebase.google.com/, refer to *Setup - Firebase.pdf*

* *Do not use TAR UC student gmail account as it doesn't come with Google Firebase features.*
* *Recommend to create a **new** Gmail account to be used among team members (gmail password is used as part of log in procedure to be shown in your python code)*

Step 2: There are many ways to establish connection to Google Firebase.

We use Pyrebase, that is a simple python wrapper for the Firebase API, pre-installed in our Raspberry Pi module

* If you are using your own Raspberry Pi, do install Pyrebase by typing:

  sudo pip3 install Pyrebase

* Reference: https://github.com/thisbejim/Pyrebase

Step 3: In **Thonny Python (IDE)**, open *test05.py* file and Save As "test06.py". Modify the program code with the following codes, replace the details with your own Firebase credentials, i.e., [API_KEY], [PROJECT_ID], [DATABASE_NAME] ...

test06.py

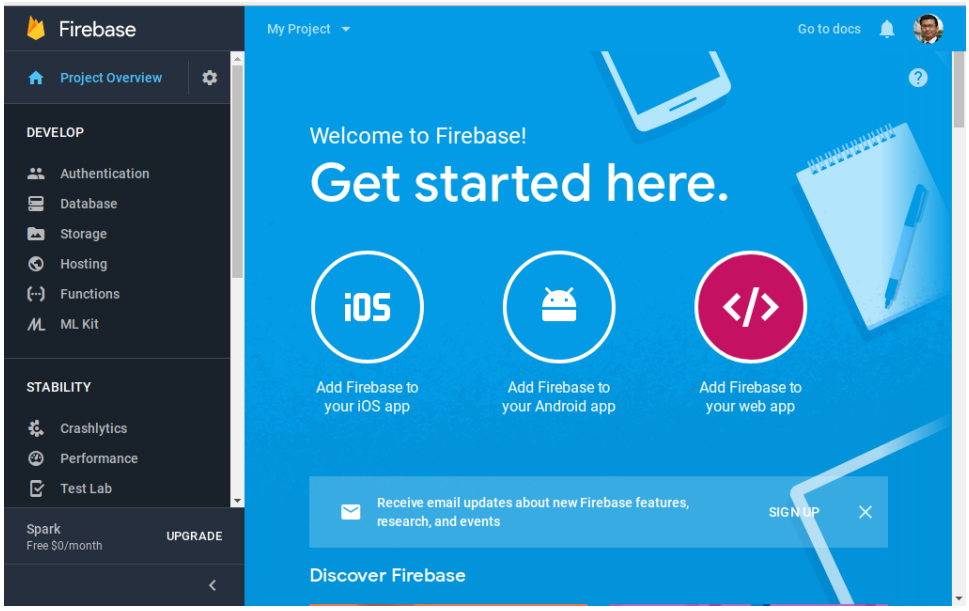
```
from time import *
from grovepi import *
from grove_rgb_lcd import *
from pyrebase import pyrebase

dhtsensor = 7
pinMode(dhtsensor, "INPUT")

config = {// For use with only user based authentication we can create the following configuration
  "apiKey": "[API_KEY]",
  "authDomain": "[PROJECT_ID].firebaseapp.com",
  "databaseURL": "https://[DATABASE_NAME].firebaseio.com",
  "storageBucket": "[PROJECT_ID].appspot.com"
}
firebase = pyrebase.initialize_app(config)
auth = firebase.auth()
user = auth.sign_in_with_email_and_password([EMAIL_USERNAME], [EMAIL_PASSWORD])
db = firebase.database()

while True:  *Firebase has A.I. to detect spamming. In order to let the data pushing action not being considered as spam, we use sleep(5) to delay for 5 seconds which ensure the code can well-behave when push the data to firebase.
    try:
        # adjust the sleep time if you have succesfully push data to your firebase
        sleep(5)
        [temp, hum] = dht(dhtsensor, 0)
        print("Temp = ", temp, '\u00b0C', " Hum = ", hum, " %")
        t = str(temp)
        h = str(hum)
        setRGB(0, 255, 0)
        setText("Temp = " + t + '\337'+ "C   Hum = " + h + " %")
        data1 = {"temperature":t}
        data2 = {"humidity":h}
        results = db.child("PI_001").push(data1, user['idToken'])
        results = db.child("PI_001").push(data2, user['idToken'])
        # remove the break if you have succesfully push data to your firebase
        break
    except KeyboardInterrupt:
        setText("Program Exited")
        break
    except TypeError:
        print("Type Error occurs")
    except IOError:
        print("IO Error occurs")
```

1. Get the Firebase credential from the Firebase site > Project Overview> Project Setting (refer to Setup - Firebase.pdf file)



2. Run your code.

Task 1: Test your temperature and humidity sensor by facing it with a few breaths and show it in your firebase's database. Modify the code to display the values without adding a new "child" (updating the same child values for every 2 seconds).

* Tips: refer to the Pyrebase online document reference.

Step 4: Test Database update with other sensors

1. Recall Practical 3 Step 5 or Practical 4 Step 1 to post push button or ultrasonic sensor data to the Firebase's Database.

refer push_sensor_data.py (I demonstrate for Practical 4 Step 1 which push the ultrasonic sensor data to Firebase Database)

Task 2: Adjust the database table and add additional sensors to store more data (besides humidity and temperature) from different Raspberry Pi, with time stamps.

Additional Support on using Google Firebase

Google Firebase with .NET
https://firebase.google.com/docs/reference/admin/dotnet

Google Firebase with C# (example)
https://www.example-code.com/csharp/fireBase.asp

Google Firebase with Android Project
https://firebase.google.com/docs/android/setup
https://firebase.google.com/docs/database/android/start