UNIVERSITÉ DES MASCAREIGNES

SAVOIR, C'EST POUVOIR

Université de Limoges

IoT Using Raspberry Pi, Firebase and Android

Internet of Things, Web & Mobile

05 DECEMBER 2022

HEMANT RAMPHUL

Table of contents

TP 09: Create an IoT Android App using Raspberry Pi and Firebase

1. TP Overview

In a nutshell, I will create a smart device that can get temperature and humidity wirelessly using DHT11 with Raspberry Pi 4 over Wi-Fi on your Android smartphone. This project is divided into three main parts.

- **Android mobile application.**
  The mobile app which read temperature and humidity sensor data on real time.
- **Database.**
  Firebase database will receive some data from the mobile app or Raspberry Pi.
- **Electronic and control circuit.**
  The control circuit will be based on the raspberry pi board which running a python script reading the incoming data from the firebase database and according to these data will take some different actions.

2. How does the project work?

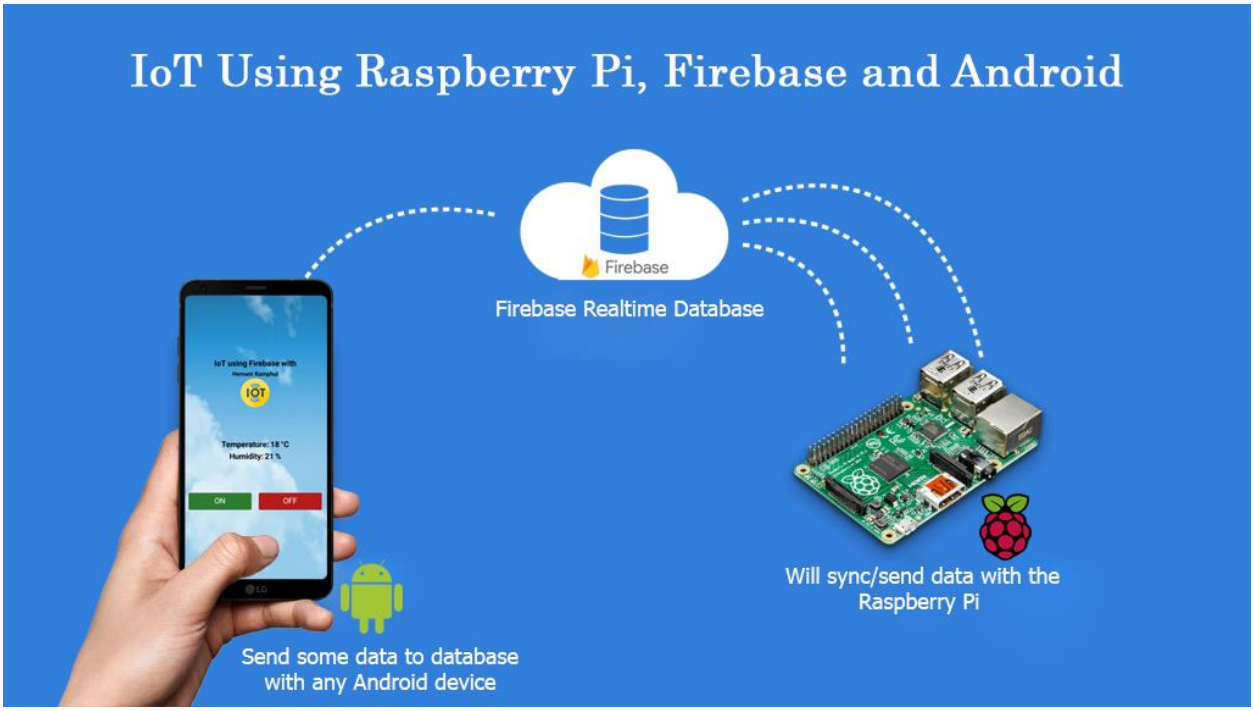


Fig 1. Project Working Logic

Simply, the mobile app and the raspberry pi board are connected to the same firebase database. This database saves the data which comes from the mobile app and syncs it with the raspberry pi board in milliseconds, according to these data the raspberry pi board will take some specific actions like turning on the light, changing its brightness, and more things can be added.

3. Wiring Diagram

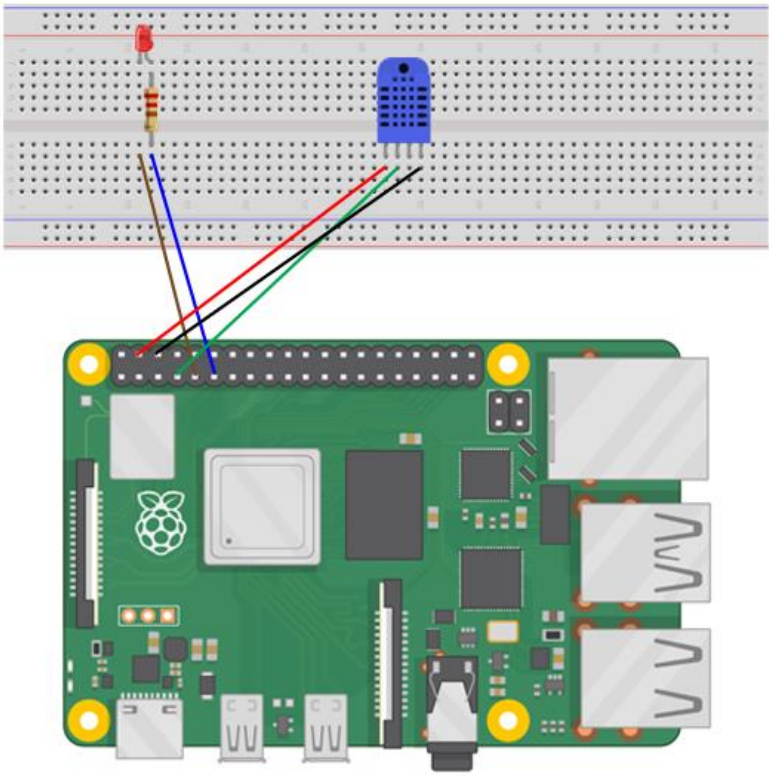


Fig 2. Project Wiring diagram with Raspberry Pi 4

The wiring diagram is very simple, connect DHT11 Sensor on a different pin

DHT11 Sensor Wiring Connection	
VCC	DC Power 5v
Data	GPIO 4
GND	Ground

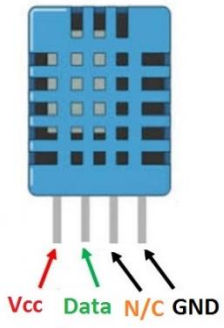


Fig 3. DHT11 Sensor and description

The raspberry pi has two different schemes in numbering its **GPIO** pins. you can use either pin numbers (**BOARD**) or the Broadcom **GPIO** pin numbers (**BCM**). **<u>You can only use one numbering scheme in each program.</u>**

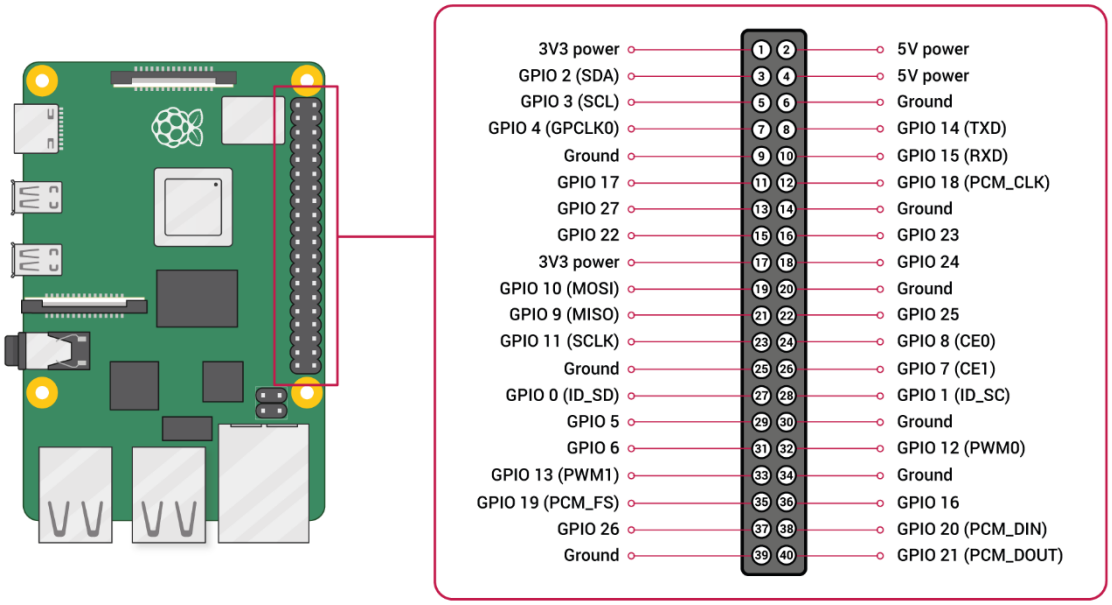


Fig 4. Raspberry Pi 4 Specifications Pin Diagram and Description

GPIO Board: Referring to the pin by its number of the plug, the pin number on the physical board which printed on it. Example: PIN#1, PIN#2, …

GPIO BCM: Referring to the pin by its "Broadcom SOC Channel" number, these are the numbers after the GPIO word. Example: GPIO12, GPIO19, …

4. Code Explanation

```
import time  # Import the sleep library, which responsible to do the timing
stuff.
import random as rand  # Import random library
import pyrebase  # Import database module.
import Adafruit_DHT  # import Adafruit dht library.

# Firebase configuration
# Dictionary named config with several key-value pairs that configure the
connection to the database.
config = {
    "apiKey": "_enter_your_api_key_here_",
    "authDomain": "iotdht11sensor.firebaseapp.com",
    "projectId": "iotdht11sensor",
    'databaseURL': 'https://iotdht11sensor-default-rtdb.firebaseio.com/',
    "storageBucket": "iotdht11sensor.appspot.com",
    "messagingSenderId": "658125581309",
    "appId": "1:658125581309:web:127027f2624c1c3439c794",
    "measurementId": "G-64BXKX08FT"
}

# Initialize the communication with the "firebase" servers using the previous
config data.
firebase = pyrebase.initialize_app(config)
# Take an instance from the firebase database which is pointing to the root
directory of your database.
database = firebase.database()

# Delay in-between sensor readings, in seconds.
DHT_READ_TIMEOUT = 15

# PIN/GPIO connected to DHT11/DHT22 data pin
DHT_PIN = 4

# Set up DHT11/DHT22 Sensor.
DHT_SENSOR = Adafruit_DHT.DHT11


# Define a function to save data to Firebase real time database
def saveSensorDataToFirebase(temperature, humidity):
    # Prepare data to save
    data = {
        "Humidity": humidity,  # set humidity
        "Temperature": temperature  # set temperature
    }

    # Get a child to database
    database.child("DHT11SensorData").child("SensorData").push(data)


# Define a function to grab a sensor reading. Use the read_retry method.
def readSensorData():
    # Reading from DHT11/DHT22 and storing the temperature and humidity
    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)
    return humidity, temperature
```

```
# Define a function that will post [Temperature] and [Humidity] on Thingspeak
server every 15 Seconds.
def start():
    # Get data from sensor
    humidity, temperature = readSensorData()

    # Note that sometimes you won't get a reading and the results will be
null
    # (because Linux can't guarantee the timing of calls to read the sensor).
    # If this happens try again!
    if humidity is not None and temperature is not None:
        # Display generated data in the console
        print('Temperature = {0:0.1f}℃ Humidity =
{1:0.1f}%'.format(temperature, humidity))
        # Sending data to Firebase RealTime Database
        saveSensorDataToFirebase(temperature, humidity)
    else:
        print('Failed to get reading. Try again!')


if __name__ == '__main__':
    while True:
        # Start the main function
        start()
        # Delay 15 sec
        time.sleep(DHT_READ_TIMEOUT)
```

Download full source code from (Github) 100% executable and each line is explained.

To send data to Firebase real-time database using Python, we must use library **pyrebase**
To use it, we must install it first by using the command below:

```
pip3 install pyrebase
```

Brief note: **Pyrebase** is a Python interface to Firebase's REST API.

Important:
If **pyrebase** not available for *Python 3*, try to install **pyrebase4** from (github.com/nhorvath/Pyrebase4)

```
pip3 install pyrebase4
```

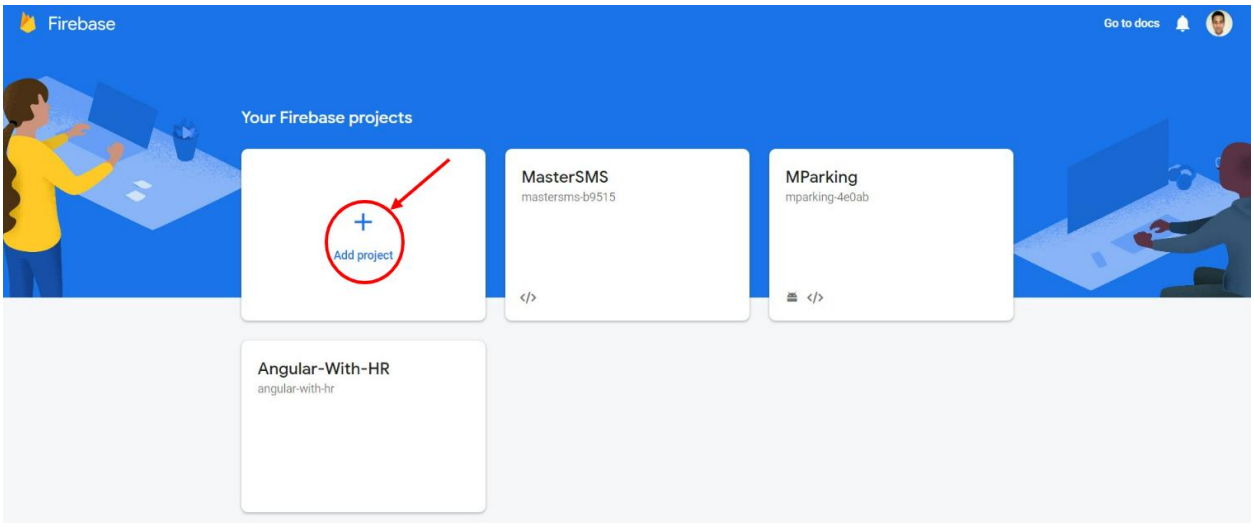
5. Building the Firebase Database

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client.

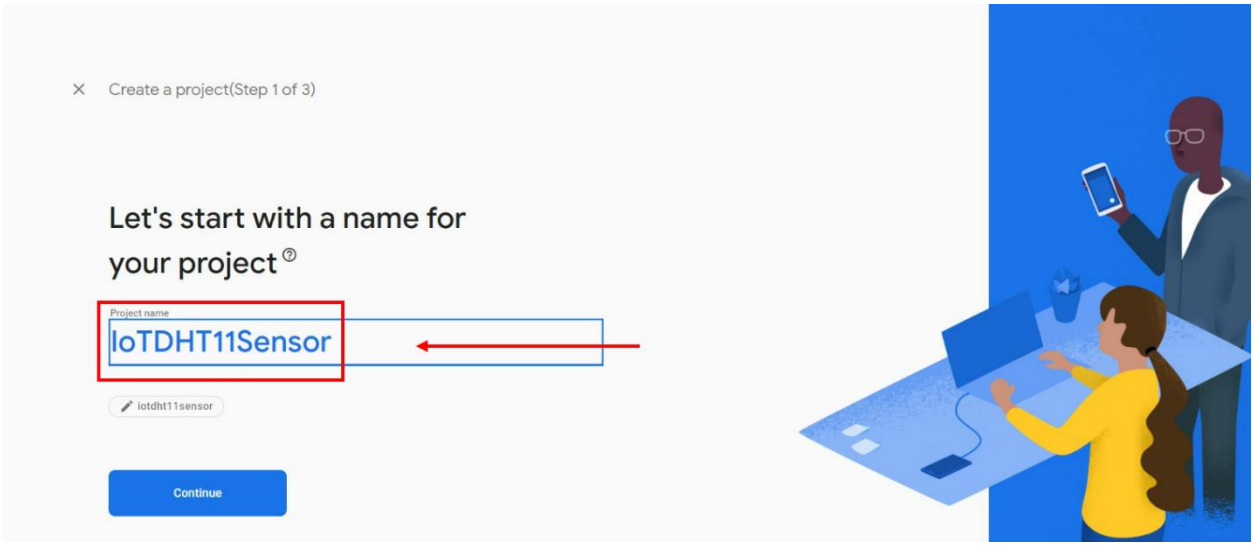
As stated before, the real-time database will receive some data from the mobile app based on the DHT11 Sensor.

So, to be able to use Firebase you need to sign in using your normal **Gmail** account (mail.google.com). Then go to your project console (console.firebase.google.com) to create a new project.

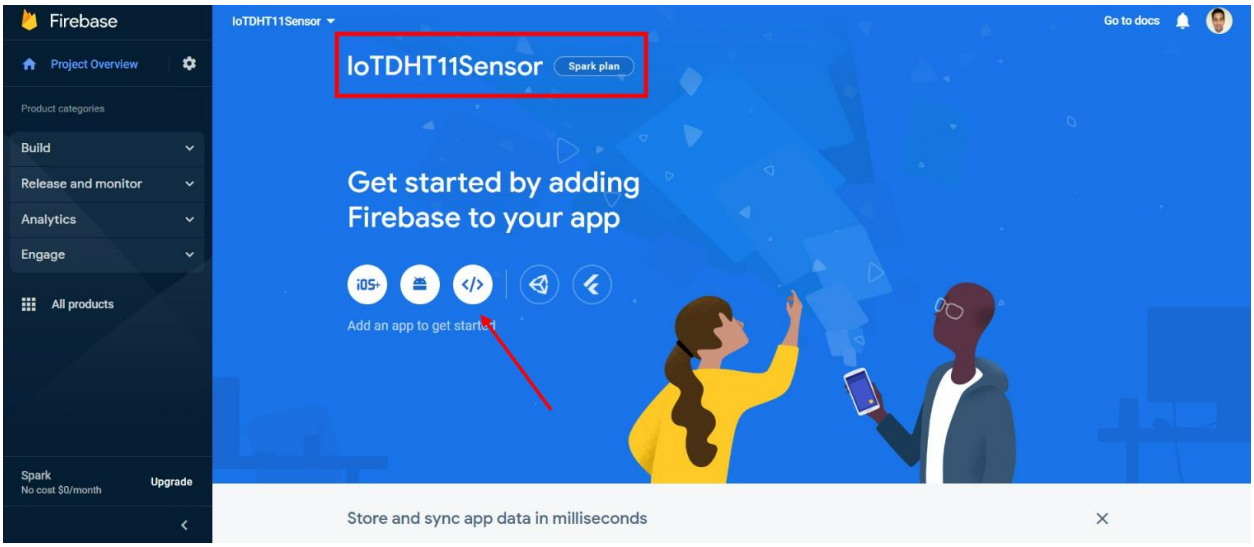
1. Create a new project in your console



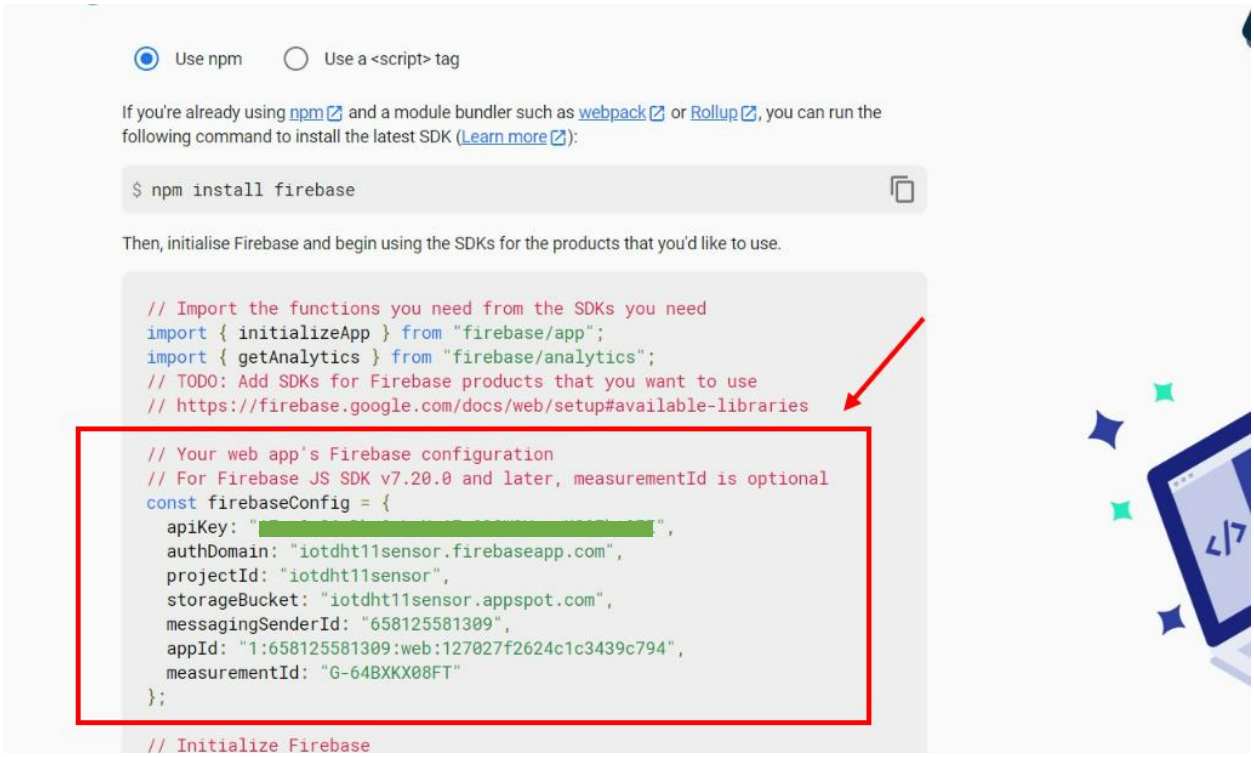
2. Then, give a name to your project



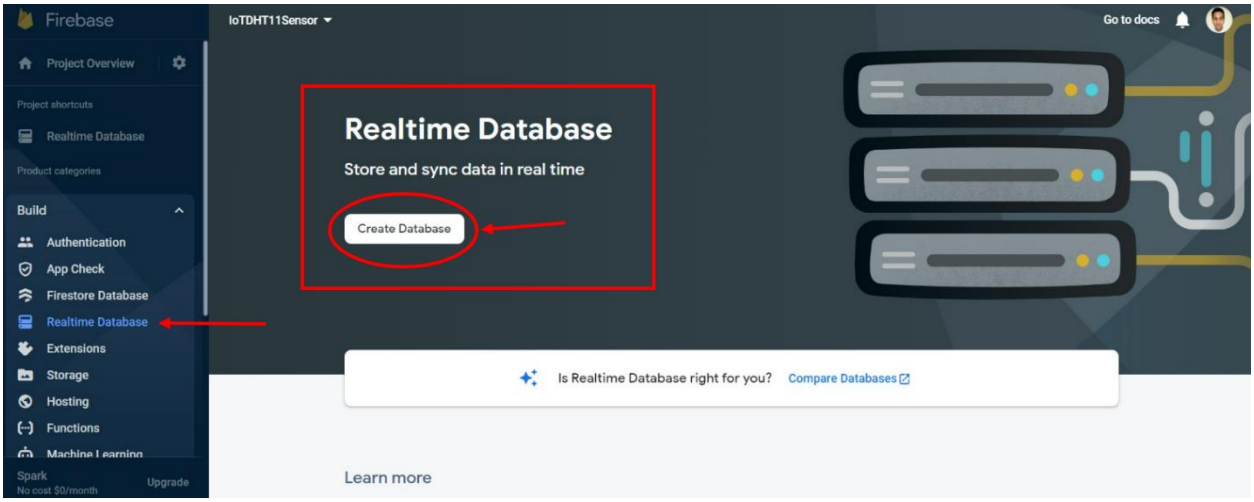
3. After creating your project successfully, your project overview page should look something like this. From the left menu, you need to register to an app by click on “Add a WEB App” to obtain your Firebase SDK configuration.



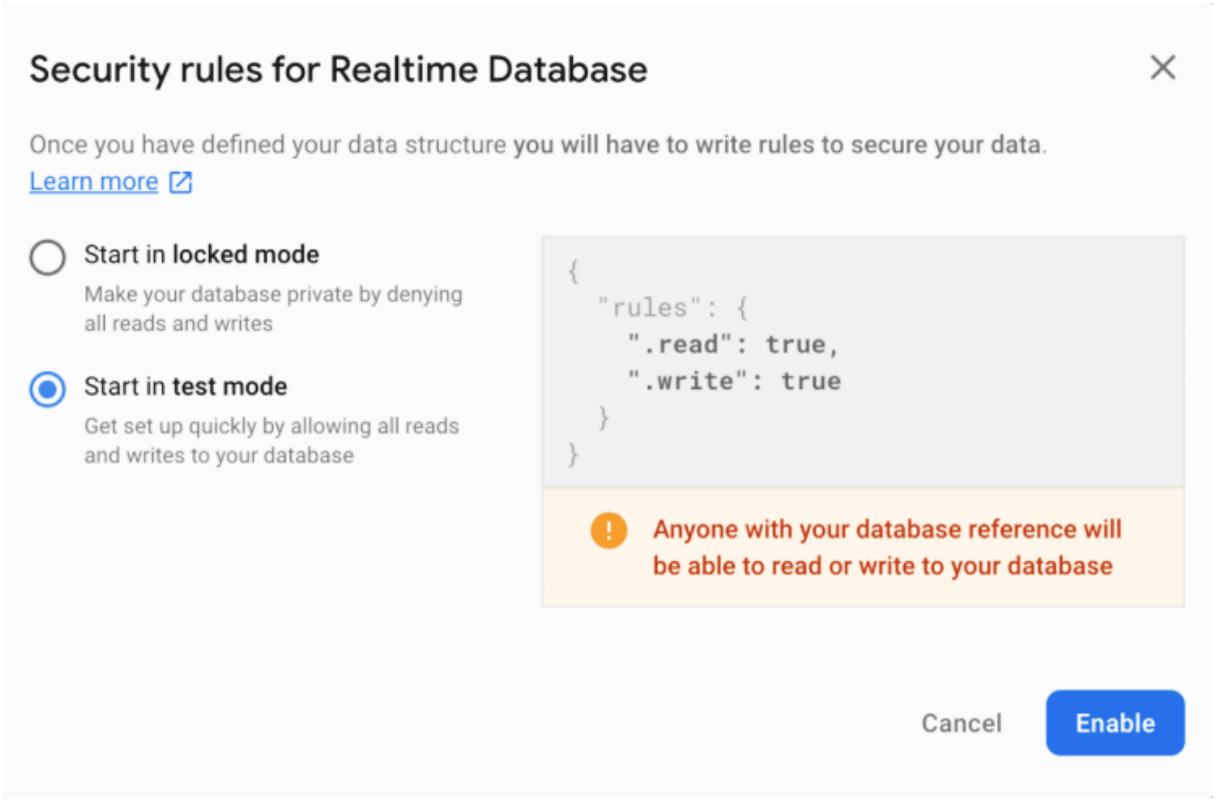
4. Example of Firebase configuration



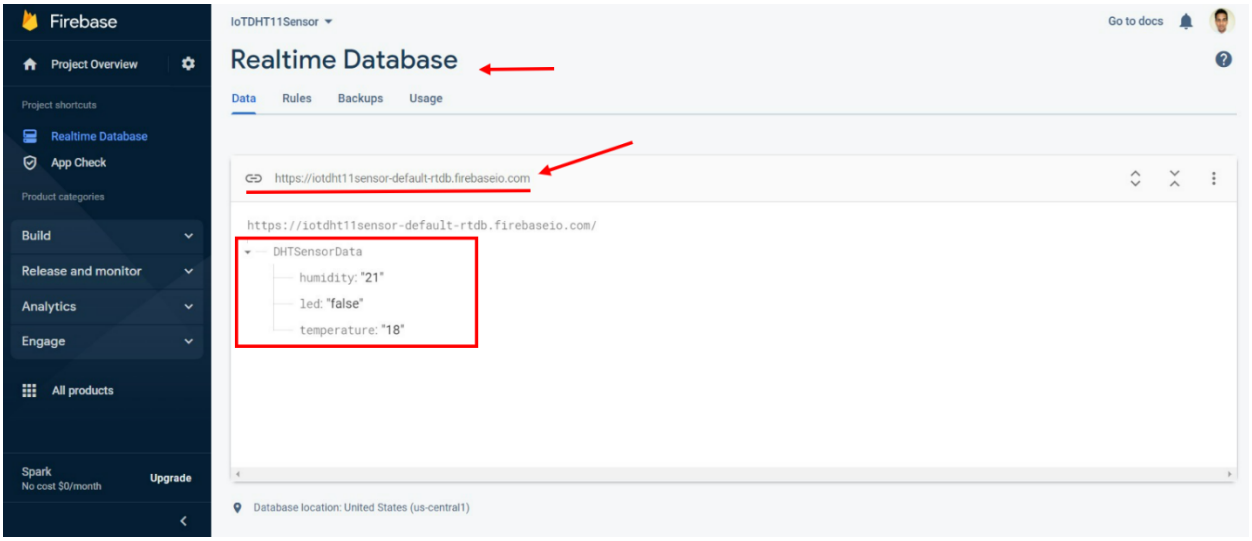
5. Then will choose **"Realtime database"** menu and click the **"Create Database"** button.



6. After that, select **"Test Mode"** and click **"Enable"**



7. Now, your Realtime database should look something like this.



After building the Realtime Database, we need to build the mobile app that will send the data to the firebase database.

6. Building the Android Mobile App (MIT APP Inventor)

Now I need to build the mobile app that will use to send data, I used the amazing tool the **"MIT APP Inventor"** to build it. It's fast and very easy to use.

App inventor lets you develop android mobile applications using a web browser and a physical mobile phone or just the emulator, the MIT app inventor saves your projects in their clouds to keep track of your work.

First, you need to go to the MIT App Inventor official website and login with your **Gmail** account, then press on the "Create apps" button, then start building your mobile app.

To build your mobile app using the app inventor tool, you should deal with two main parts

- The design of the app.
- The programming of the app.

6.1. The App Design

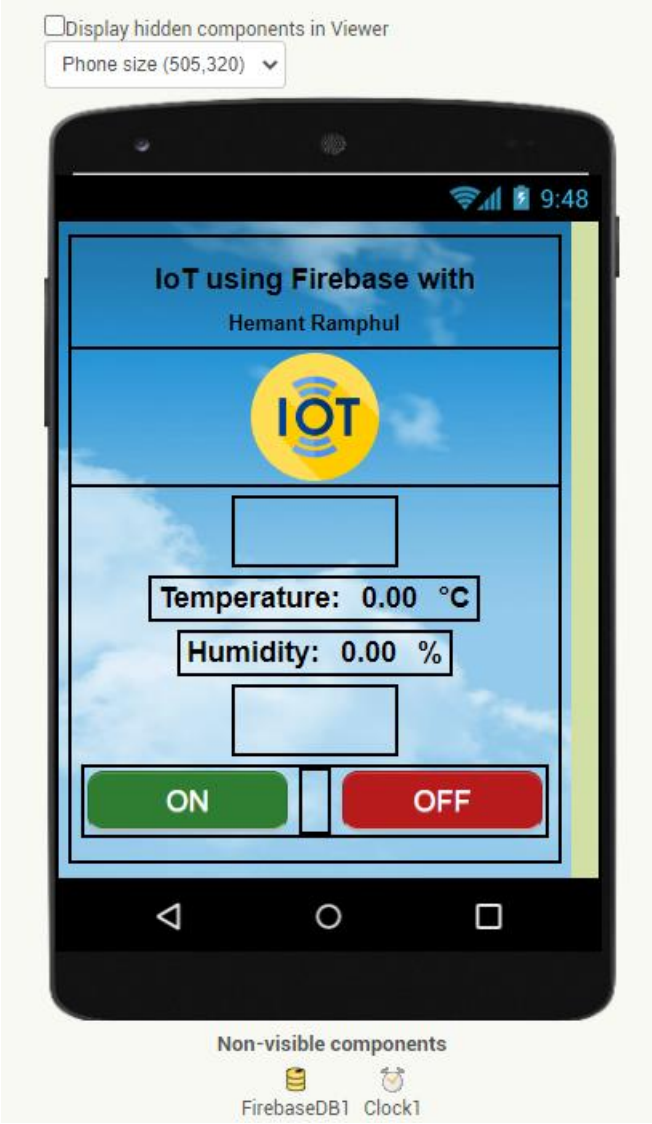


Fig 5. Wireframe of mobile app

The mobile application consists of two buttons to send data (Mode **ON** – start sending data, Mode **OFF** – stop sending data). The most important part here is to add the "**FirebaseDB**" client components to the app to allow us sending data to our firebase database project that we created before.

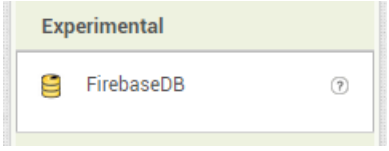


Fig 6. Adding FirebaseDB component

After adding the "**FirebaseDB**" component to the app, we need to put our created Firebase database URL to the mobile application "**FirebaseDB**" component to specify where does the data will get saved, the Firebase database URL can be found here.

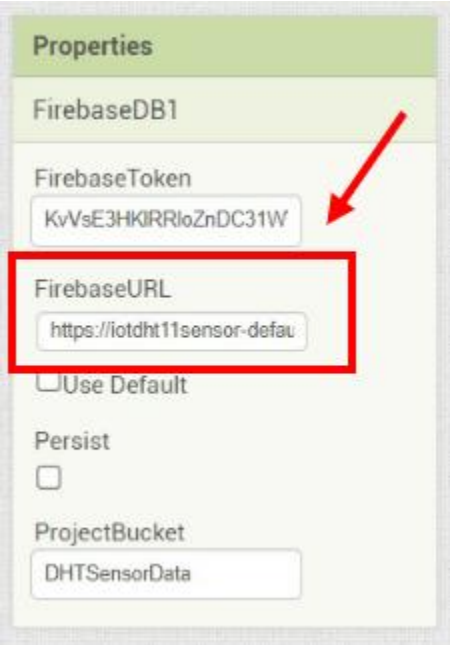


Fig 7. FirebaseDB component properties

6.2. The Programming of the App

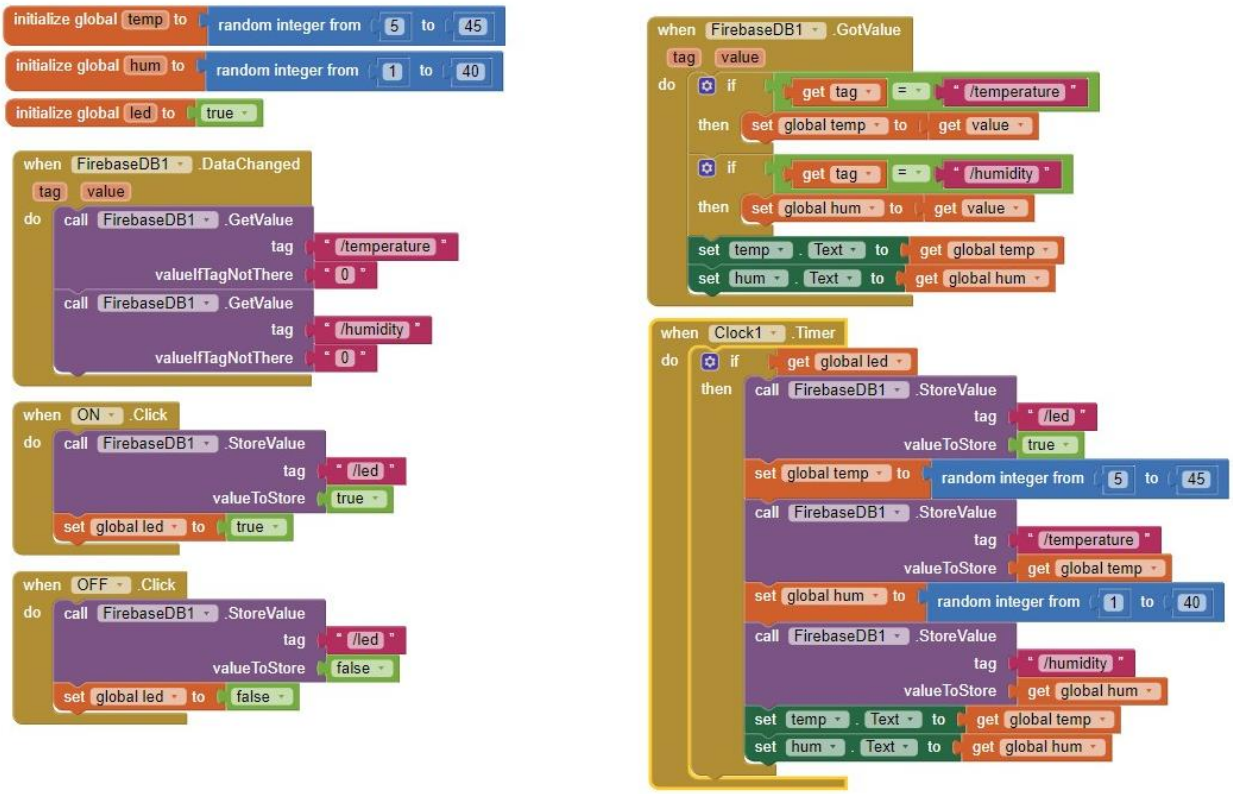


Fig 8. The App Code

6.3. Building the app APK file

Build the app APK file and download it to your computer, then send the APK file to your mobile and install it.

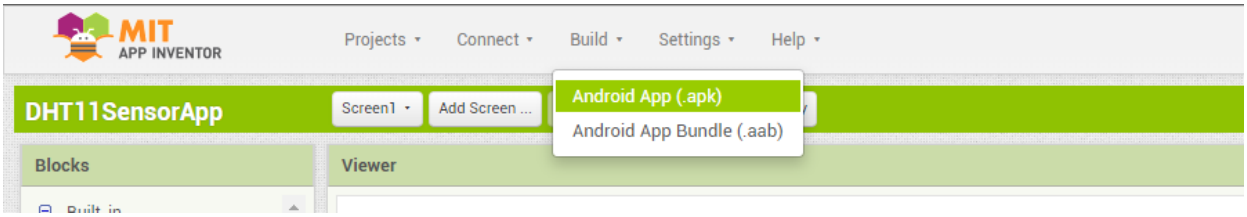


Fig 9. Build APK file

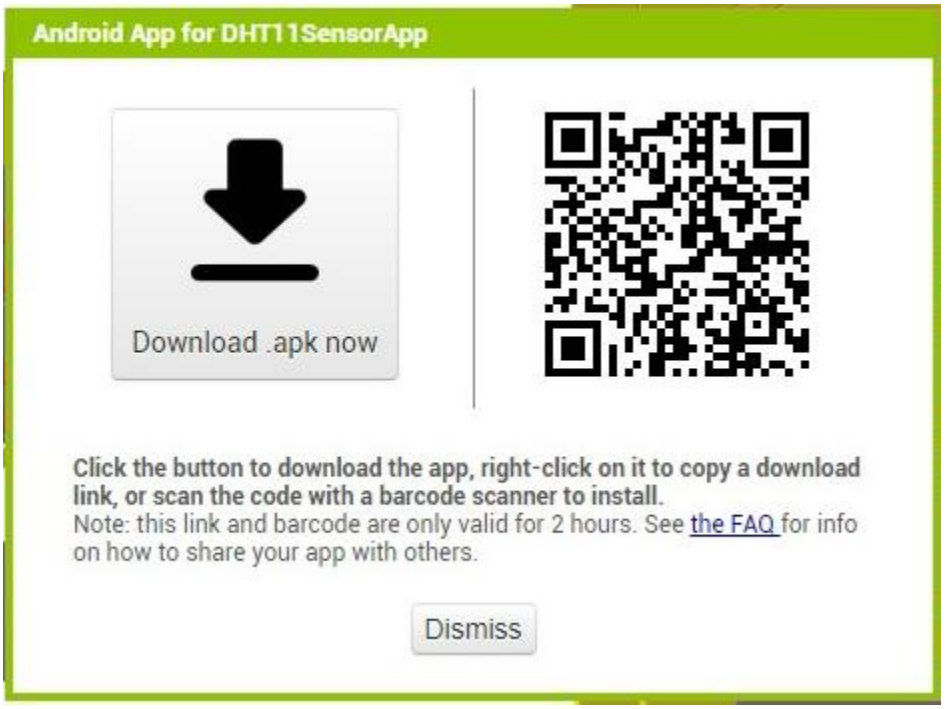
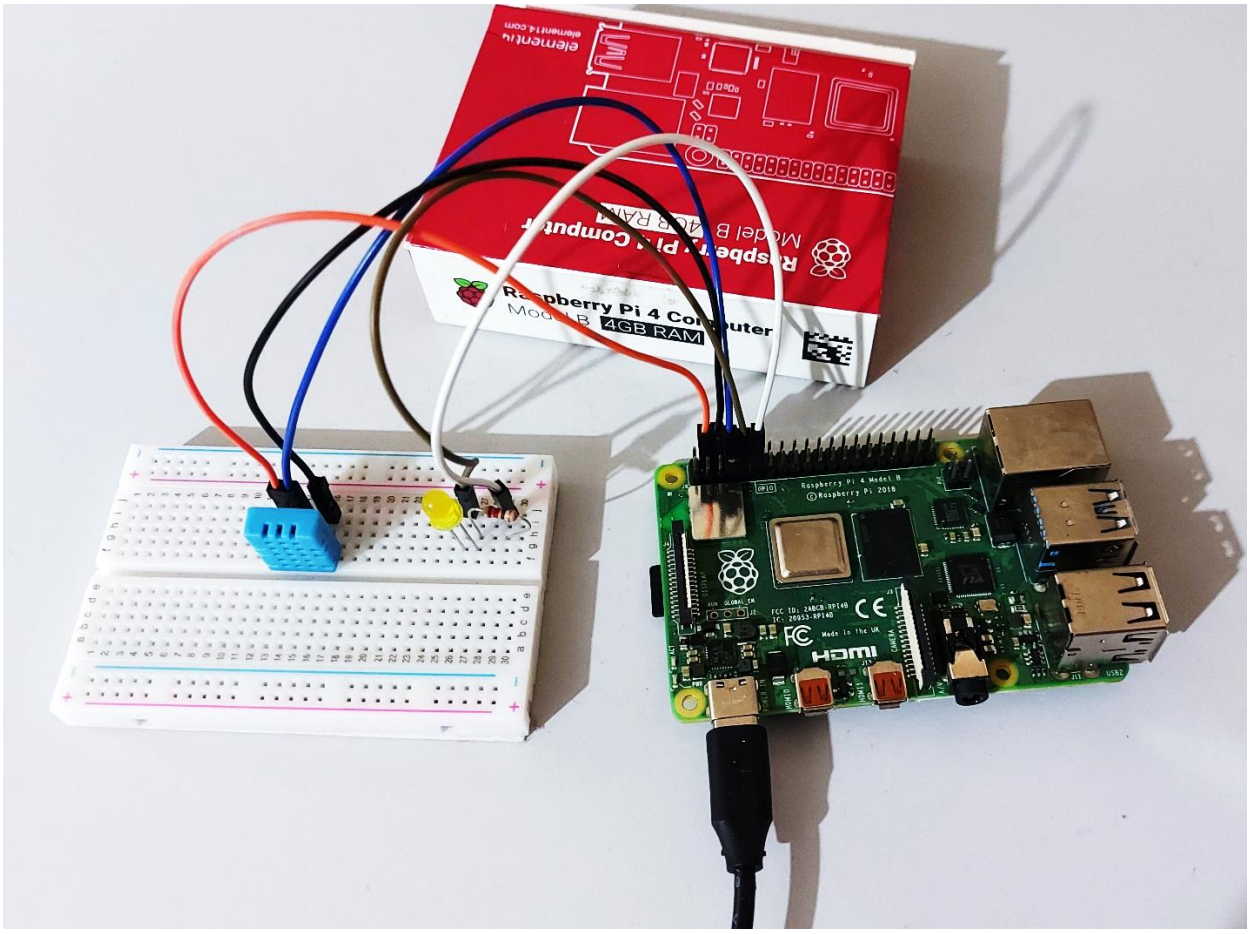


Fig 10. After build I can download it directly or by scan QR code

7. My Setup



8. My source code and APK

Full **source code** and **DHT11SensorApp.apk** file is available in Github which is 100% executable and each line is explained.

Download link:

https://github.com/hemantramphul/IoT-Using-Raspberry-Pi-and-Firebase-and-Android