

|  |
| --- |
| IoT Project  2021 |
|  |
| November 27  IoT COMP4980  Authored by: Liang Zhong |

# Introduction

|  |
| --- |
| This IoT project use a raspberry pi, a humidity and temperature sensor to detect environment data, and send data via the internet connection(WIFI or Ethernet) from the raspberry pi to the firebase real-time database, then display the real-time humidity and temperature values stored in the real-time database at <https://iot2021-94458.web.app/>  The database chosen for this project is Google’s firebase real-time database, and the cloud web application hosting is Google’s firebase hosting. Object The object of this project is to learn how to use raspberry pi to interact with the sensor and the server in the cloud. Innovation Cloud technology is an innovation for this project. After I used Google’s cloud technology, I feel that Google’s API is hard to use, and Google’s technology is not flexible. Google use its own unique database structure, and APIs are not well documented. I haven’t used AWS, but I heard that AWS has a more flexible API library, better documentation, and more available database structures. Consider AWS has about 40% market share, and Google has 8% market share in the clouding business. There will be more jobs for AWS than for Google Clouding. I would rather to study AWS technology than Google clouding technology. Google is good at data analysis, so probably Google is better for global high technology company like Spotify, Financial companies who has simple data structures, but huge data volumes to be analyzed or streamed. For traditional business, I think AWS might be a better choice. Local Hardware Humiture Sensor Module: $9.99  <https://www.sunfounder.com/products/humiture-sensor-module>  Raspberry Pi  Ethernet or WIFI gateway    First, you must have a way to program the pi, and connect the pi to the internet.   1. Use USB to serial port to connect a laptop computer to the raspberry pi to program it. Windows: use putty app- connection type: serial, serial line: com3, speed:115200 2. Then connect pi to internet via its Ethernet port or WIFI.   Or   1. Use Pi’s ethernet port to connect to a laptop computer through SSH. Windows: use putty app – connection type: ssh, host name: Group3.local (Group3 is this pi’s id), port: 22, after connected, enter user: pi, password. Mac’s terminal: ssh [pi@Group3.local](mailto:pi@Group3.local) 2. Then connect pi to internet via WIFI only, since Ethernet port is used by connecting computer already.   Second, you must connect humidity and temperature sensor to the pi.  The digital temperature and humidity sensor DHT11 is a composite sensor that contains a calibrated digital signal output of temperature and humidity. This signal pin is connected to raspberry’s GPIO4. DHT11 has another 2 pins, the DHT11’s VCC pin is connected to pi’s 5V power supply, and DHT11 GND pin is connected to pi’s ground.  This diagram connects DHT11’s signal to pi’s GPIO17  The pi must have a running program to monitor the environment via sensor(DHT11), send data to the clouding database. This program is dht.py (please see its source code in the dht.py file submitted with this report in the same folder).  *Below is the finished working demo running in raspberry pi*   Cloud Web Application Third, you must set up firebase real-time database and online hosting.   1. Set up firebase real-time database        1. Set up firebase hosting   Using the Firebase CLI, you deploy files from local directories on your computer to our Hosting servers.  Implementation path 1: install the firebase CLI  For windows system, there are only 2 options:   1. **standalone binary-**Download the standalone binary for the CLI. Then, you can access the executable to open a shell where you can run the firebase command. For developers not using or unfamiliar with [Node.js](https://www.nodejs.org/) 2. Use npm (the Node Package Manager) to install the CLI and enable the globally available firebase command. For developers using [Node.js](https://www.nodejs.org/)     For macOS or Linux system, there are another option other than the above 2 options. This is the option I am using for the MacBook.   1. **automatic install script -** Run a single command that automatically detects your operating system, downloads the latest CLI release, then enables the globally available firebase command. For developers not using or unfamiliar with [Node.js](https://www.nodejs.org/), Automated deploys in a [CI/CD](https://en.wikipedia.org/wiki/CI/CD) environment   To install the Firebase CLI using the automatic install script in the project’s root folder(/iot for me), follow these steps:   1. Run the following cURL command:   $ curl -sL https://firebase.tools | bash  note: the terminal will ask user to enter the computer’s password. After password entered, start downloading from google.  [firebase-tools@9.23.0](mailto:firebase-tools@9.23.0) is now installed  all done     1. Login and test the firebase CLI:   Log into Firebase using your Google account by running the following command; This command connects your local machine to Firebase and grants you access to your Firebase projects.  $ firebase login  note: the terminal opens a web page that connects to google account for authentication, and after you approve the account connection, it will goes to localhost on your machine: hocalhost:9005/?state=……, which is a page show: firebase CLI login successful.   1. Initialize a firebase project:   Many common tasks performed using the CLI, such as deploying to a Firebase project, require a **project directory**. You establish a project directory using the firebase init command. A project directory is usually the same directory as your source control root, and after running firebase init, the directory contains a [firebase.json](https://firebase.google.com/docs/cli" \l "the_firebasejson_file) configuration file.  To initialize a new Firebase project, run the following command from within your app's directory:  $ firebase init    At the end of initialization, Firebase automatically creates the following two files at the root of your local app directory:   1. The firebase.json file is required to deploy assets with the Firebase CLI because it specifies which files and settings from your project directory are deployed to your Firebase project. 2. A .firebaserc file that stores your project aliases. You can associate multiple Firebase projects with the same project directory. 3. Serve and test your Firebase project locally   Run any of the following commands from your project directory to emulate your project using local HTTP functions. To emulate HTTP functions and hosting for testing on local URLs:  Note: must run the server when test the web app locally, then open app in a browser at localhost:5000  $ firebase serve (run server locally at terminal)    Firebase CLI install a lot of folders and files in the project root folder(/iot). However, you must re-write the index.html file which is the homepage of your hosted web app. You must write interface between this page on server to interface with firebase real-time database. This page is display to the world at https://iot2021-94458.web.app/. (Source code of this index.html is submitted with this report in the same folder).   1. Deploy to a firebase project   $ firebase deploy  After this deployment, this web page is live at https://iot2021-94458.web.app/  Please see the below screenshot of the completed web app deployment:    Final deployed web page:   Conclusion The demo YouTube link is <https://www.youtube.com/watch?v=Jpqg2_aCRcw>  There are two parts of this project: one is to use raspberry pi and sensors; one is to use Google firebase hosting and real-time database. The hard part is to use Google API to interact with real-time database. Google document is not very good for using. I must guess a lot.  At least, I can deploy a hosting webpage online now. I could add more things to that website later to make a personal website. This is a good learning practice to know the Google clouding technology. |