

# **Step-by-step Guide**

**Greenhouse**

**Setup Guide**

**Version 1.0**  
**[20 August 2017]**

# Contents

## LIST OF ILLUSTRATIONS

Overview of Greenhouse

Pre-requisites

Setup Twilio

Create Twilio Account

Create a New Twilio API Key

Configure Twilio Phone Number

Enable Global SMS

Install Twilio py in your raspberry pi terminal

Setup AWS

Create new AWS account

Sign into the AWS IoT Console

Create and Register your raspberry Pi as “Thing”

Copy Rest API endpoint of your “Thing”

Create Certificate

Create a Security Policy for your RPI

Attach Security Policy and Thing to your Cert

Install the AWS Python Library

Setup Firebase

Create Firebase Account

Setup Cloudinary

Create Cloudinary Account

Setting Up Raspberry Pi

## LIST OF ILLUSTRATIONS

### **Figure**

Figure 1: Sign Up Page

Figure 2: Phone Verification

Figure 3: Verification Code

Figure 4: Project Name

Figure 5: Twilio Account SID

Figure 6: Create New API Key

Figure 7: New API Key

Figure 8: Success Message

Figure 9: Get Phone Number

Figure 10: Get Twilio Number

Figure 11: Random Twilio Number

Figure 12: Twilio Number

Figure 13: Global SMS

Figure 14: Sign Up Page

Figure 15: Role Page

Figure 16: Sign Up Page

Figure 17: Verify Page

Figure 18: Confirmation Email

Figure 19: AWS Service

Figure 20: AWS IoT

Figure 21: Things

Figure 22: Register a Thing

Figure 23: Create Thing

Figure 24: Details Page

Figure 25: REST API

Figure 26: Security

Figure 27: Certificate Page

Figure 28: Certificated Created

Figure 29: Rename of files

Figure 30: Activated Successfully

Figure 31: Attach a policy

Figure 32: New Policy

Figure 33: Configuration Steps

Figure 34: Going back to Previous Page

Figure 35: Nav Bar

Figure 36: Attach Policy

Figure 37: Attach policy to Certificate

Figure 38: Attach Thing

Figure 39: Attach Thing to Raspberry Pi

Figure 40: Get Started

Figure 41: Dashboard Page

Figure 42: Add Project

Figure 43: Firebase Console

Figure 44: Authentication Tab

Figure 45: Sign-In Method

Figure 46: Add Firebase to Web

Figure 47: API Key for Firebase

Figure 48: Email/Password

Figure 49: Setting

Figure 50: Project Setting

Figure 51: Service Account

Figure 52: Private Key

Figure 53: Signup Page

Figure 54: Dashboard

## **Table**

Table 1: Email and Password

## **Overview of Greenhouse**

A system that helps to sustain plant life automatically without human intervention.

### **Pre-requisites**

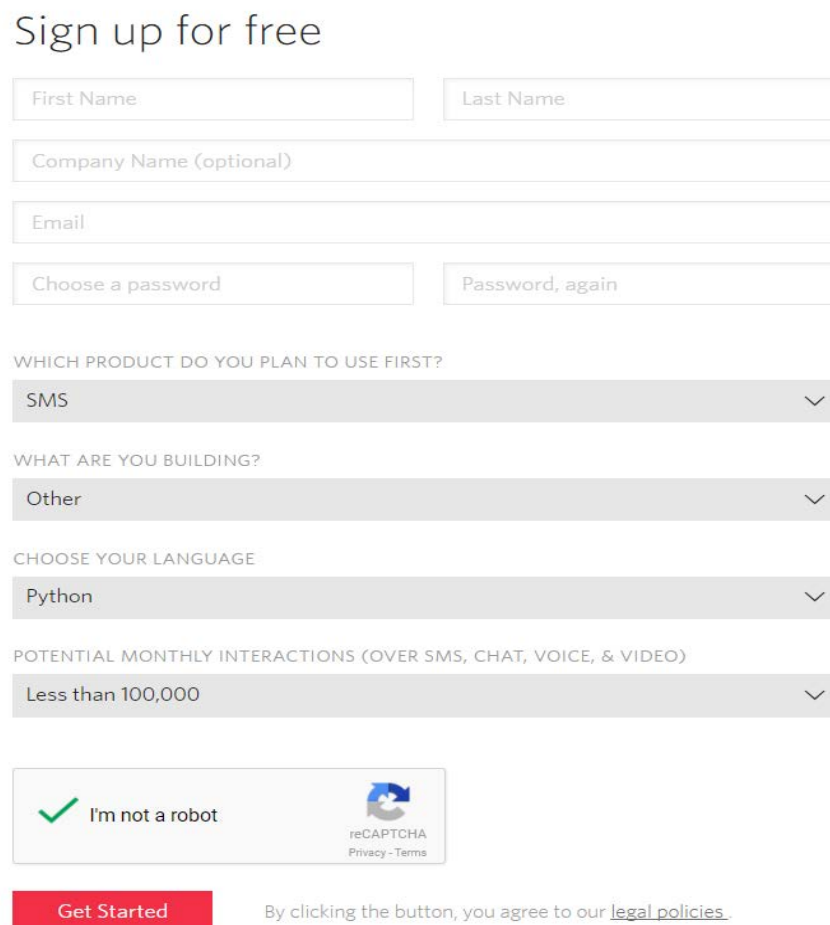
- Raspberry Pi 3
- Twilio Account
- AWS Educate Account
- Firebase Real-time Database Account
- Cloudinary

### **Setup Twilio**

Twilio is a cloud communications platform as a service company that allows software developers to programmatically make and receive phone calls and send and receive text messages using its web service APIs.

### **Create Twilo Account**

- First create an account in <https://www.twilio.com/try-twilio>



The image shows the Twilio sign-up page. At the top, it says "Sign up for free". Below this are several input fields: "First Name", "Last Name", "Company Name (optional)", "Email", "Choose a password", and "Password, again". There are four dropdown menus: "WHICH PRODUCT DO YOU PLAN TO USE FIRST?" (selected: SMS), "WHAT ARE YOU BUILDING?" (selected: Other), "CHOOSE YOUR LANGUAGE" (selected: Python), and "POTENTIAL MONTHLY INTERACTIONS (OVER SMS, CHAT, VOICE, & VIDEO)" (selected: Less than 100,000). At the bottom, there is a reCAPTCHA section with a green checkmark and the text "I'm not a robot", and a "Get Started" button. To the right of the button, it says "By clicking the button, you agree to our [legal policies](#)."

**Figure 1: Sign Up Page**

- b. Enter your phone number to verify

We need to verify you're a human.

A form for phone verification. It includes a dropdown menu for country codes (showing a flag and a downward arrow), a text input field with a pre-filled country code '+65' and a placeholder 'Phone Number', and a red button labeled 'Verify via SMS'.

We will send a verification code via **SMS** to number above

Or, we [call you instead](#).

- ☐ The phone number you provide will be used for authentication when you login to Twilio Console. A Twilio onboarding specialist may also use this number to reach out with free onboarding support. If you do not want to be contacted at this phone number, please check this box.

**Figure 2: Phone Verification**

- c. Enter your verification code

We need to verify you're a human

Please enter the verification code we sent to your phone. If you didn't receive a code, you can [try again](#)

A form for entering a verification code. It consists of a text input field with a placeholder 'Verification Code' and a red button labeled 'Submit'.

**Figure 3: Verification Code**

- d. Create a **“PROJECT NAME”**. For this, we going to name the project as ‘Greenhouse’

## Welcome to Twilio! Give Your Project a Name

You can make changes later if you need to.

PROJECT NAME

Greenhouse

Create Project

**Figure 4: Project Name**

- e. Copy your Twilio Account SID which will be shown on the dashboard when you login in.

## Console Dashboard

### Account Summary

ACCOUNT SID

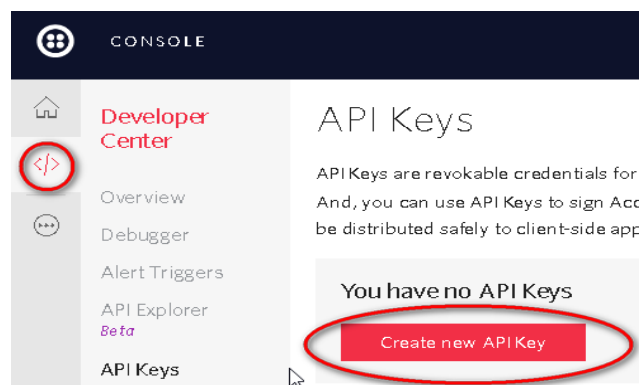
AUTH TOKEN

[Account Details](#)

**Figure 5: Twilio Account SID**

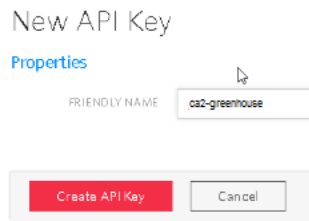
### Create a New Twilio API Key

- a. Go to **“Developer Centre → API Keys”** and click **“Create new API Key”**



**Figure 6: Create New API Key**

- b. Name it as ‘ca2-greenhouse’ or any preferred name.



New API Key

Properties

FRIENDLY NAME

**Figure 7: New API Key**

- c. You should see a success message as you in the figure below. Copy and paste the details on a notepad and keep in safe in your desktop.

ca2-greenhouse

Properties

**IMPORTANT NOTE:** This secret is only shown ONCE. Make note of it and store it in a safe, secure location.

FRIENDLY NAME ca2-greenhouse

SID SK5 [REDACTED] 35e

SECRET Wu [REDACTED] ba

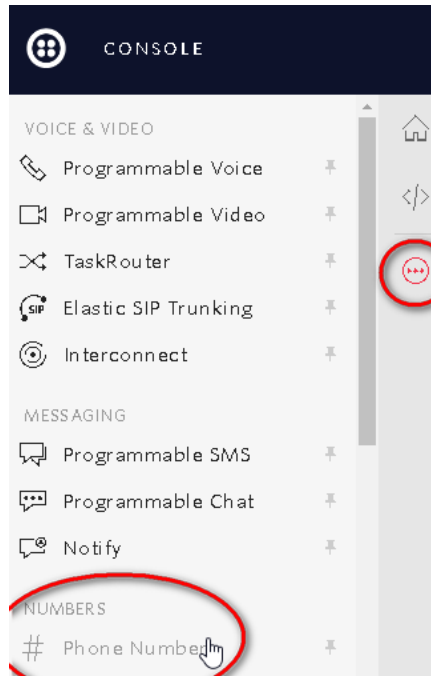
☐ Got it! I have saved my API Key Sid and Secret in a safe place to use in my application.

**Figure 8: Success Message**



## Configure Twilio Phone Number

- a. Now get yourself a Twilio phone number. As you see the figure below, go to “**All Products & Services**” and choose “**# Phone Numbers**” under the Numbers as you see in the figure below:



**Figure 9: Get Phone Number**

- b. Choose “**Get Started**” and “**Get your first Twilio phone number**”

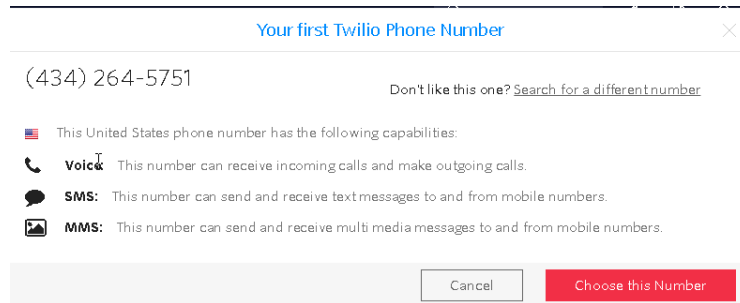
## Get Started with Phone Numbers

Getting started with Twilio's phone numbers is easy! Search for local, toll

[Get your first Twilio phone number](#)

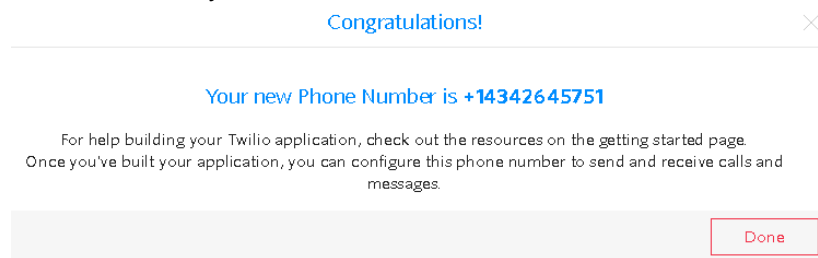
**Figure 10: Get Twilio Number**

- c. After you choose get started you will be assigned a random Twilio USA telephone number. That's ok, just click **“Choose this number”**



**Figure 11: Random Twilio Number**

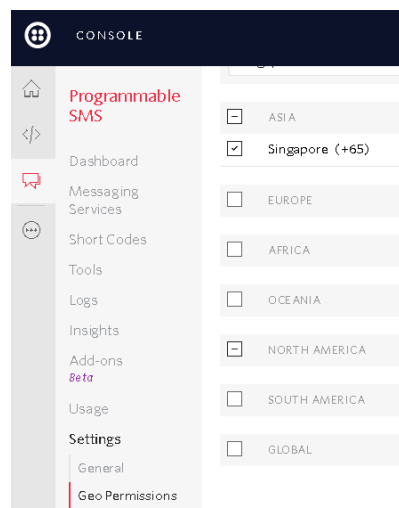
- d. Copy the phone number as you will need it later



**Figure 12: Twilio Number**

## **Enable Global SMS**

- a. Since you got a USA-based telephone number in the previous section but we will be sending SMS to your Singapore-based handphone number, you will also need to configure permissions for global SMS at <https://www.twilio.com/console/sms/settings/geo-permissions> . Add Singapore in the “Geo-Permissions” section.



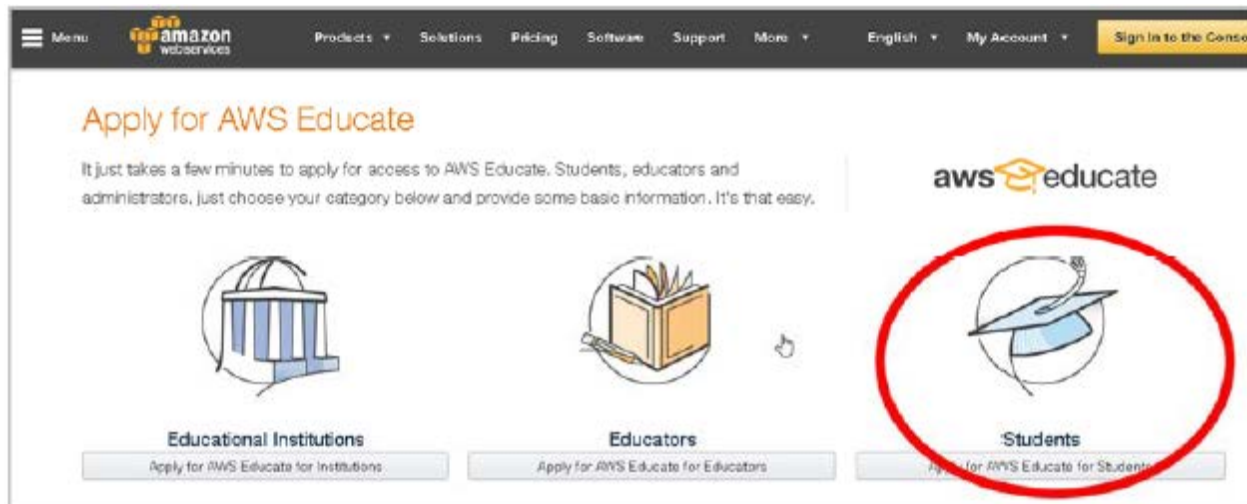
**Figure: 13: Global SMS**

## Setup AWS

AWS Educate is Amazon's Programme to help students learn real-world cloud technology skills before graduating. Under this programme, students will get a renewable annual credit of USD100 of AWS services. No credit card is necessary to sign up for this account. However, once you have used up the USD100 credit, your account will automatically be rendered unusable.

### Create new AWS account

- a. Access AWS Educate website: <https://aws.amazon.com/education/awseducate/apply/> .  
Select “**Students**” option as you see the figure below:



**Figure 14: Apply for AWS Page**

- b. Select Role as “**Student**”



## Apply to Join AWS Educate

AWS Educate is Amazon's program to help students learn real-world cloud technology skills before graduating. It provides students and educators with the resources needed to accelerate cloud-related learning.

**Get started by choosing your role!**

### Step 1. Choose Your Role

I am a(n) ☐ Educator ☒ Student

Next

**Figure 15: Role Page**

- c. On the sign up form, ensure the following (the one in red box) setting are specified, and fill in the rest of the form as appropriate and then click next button as you see in the figure below.
- **Institution Name** → “Singapore Polytechnic School of EEE”
  - **Email** → should be SP ichat email address (you can use your friend ichat email address if he/she is studying in SP & does not mind giving the email address)
  - Make sure “**Click here to select an AWS Educate Starter account**” is selected and not the other one.

**Institution Name** Singapore Polytechnic School of EEE Please write the full name of your school / institution.

**Country** --Select One--

**City**

**Field of Study** --Select One-- Please select the most appropriate

**First Name**

**Last Name**

**Email** youremail@ichat.sp.edu.sg Provide a valid, current email issued by your institution

Please choose only one of the following two options:

☐ Click here to enter an AWS Account ID (This option provides the greatest flexibility with post-graduation portability, full access to AWS services and an AWS promotional code. While a credit card is required, you will not be billed unless your usage exceeds the promo code balance.)

☒ Click here to select an AWS Educate Starter Account

Note: An AWS Educate Starter Account is a capped-account with usage limitations, including an approximately 25% reduction in access to AWS services, no post-graduation portability and is pre-loaded with your AWS promotional credit (no promo code is provided). This option is best for students without access to a credit card. See FAQs for details.

[Don't have one? Sign up now](#)

**Next**

**Figure 16: Sign Up Page**

- d. Follow the instructions to verify your account by providing a verification code that is sent to your ichat account

**aws educate**

### Step 3. Verify Email Address

We need to verify the email address you provided in your AWS Educate application before we can process it.

We sent an email address verification message to your mail box at your\_own\_email@ichat.sp.edu.sg. Please check your messages and input the verification code provided in the email.

Please do not close this page until you enter the verification code sent to your email address. If you close this page before entering the verification code, you will need to restart the application process. If you don't receive an email with the verification code in a few minutes, try checking your spam or junk mail folders.

**Verification Code**

Please click the box below to help assure that a person and not an automated program is submitting this application. If a set of letters is displayed enter them on the line. If you have any difficulty with the letters, you can click the reload icon to get a new set of letters, or click the headphones to hear audio of what to enter.

☐ I'm not a robot **reCAPTCHA** Privacy - Terms

**Next**

**Figure 17: Verify Page**

- e. Once your AWS account is ready to use, you will receive an email like the figure you see below:

Dear Bavani

Congratulations!

Your AWS Educate application has been approved. As a member of the AWS Educate program, you will gain access to the benefits listed below:

**AWS Educate Student Portal**

The AWS Educate Student Portal is the hub for AWS Educate students around the world to find AWS content to help with classwork, connect to self-paced labs and training resources.

[Click here](#) to set your password / login to the AWS Educate Student Portal. After logging in, click AWS Account at the top of the page to access AWS services, whether you entered an AWS ID or selected Starter Account on your application.

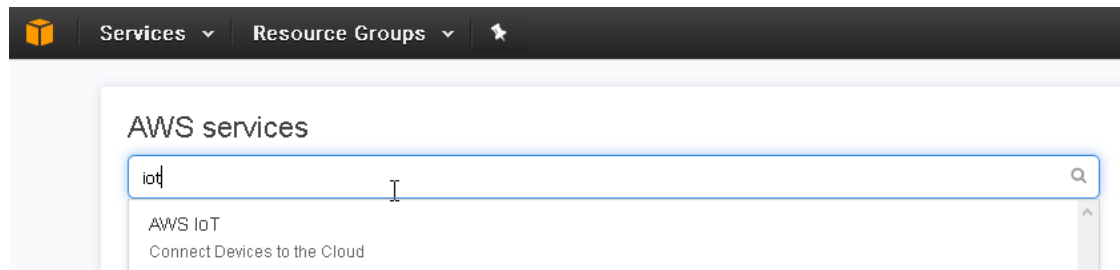
Bookmark the AWS Educate Student Portal for easy access, or [click here](#) to sign in directly.

You can access a video walk-through of the AWS Educate Student portal [here](#).

**Figure 18: Confirmation Email**

**Sign into the AWS IoT Console**

- Turn on your Raspberry Pi
- Sign in with your AWS Console at <https://aws.amazon.com>
- In the AWS dashboard, type “**AWS IoT**” to access the AWS IoT service



**Figure 19: AWS Service**

- On the Welcome page, choose “**Get Started**”



**Figure 20: AWS IoT**

## Create and Register your raspberry Pi as “Thing”

- a. In the left navigation pane click **“Registry to expand it, then select ‘Things’**



**Figure 21: Things**

- b. On the page where it state “You don’t have any things yet”, select **“Register a thing”** and if you have already created a thing before, then choose **“Create”**



## **You don't have any things yet**

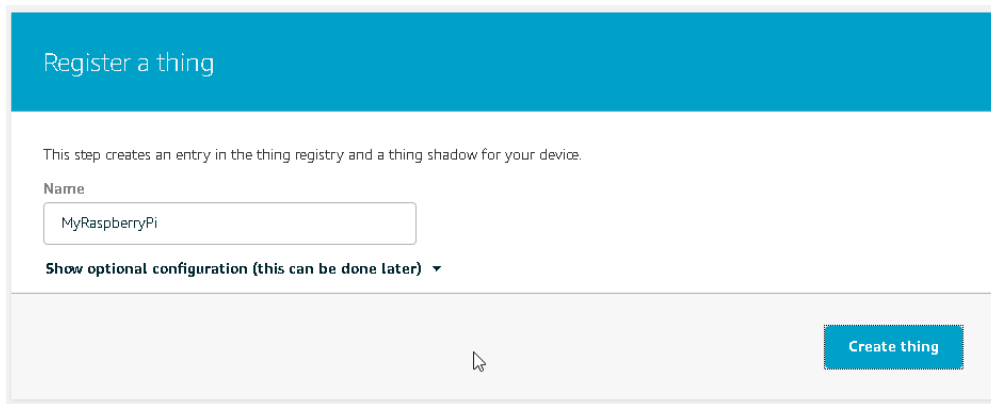
A thing is the representation of a device in the cloud.



**Figure 22: Register a Thing**

- A thing represents a device whose status or data is stored in the AWS cloud. The Thing Shadows is the state of the device, eg: is it “on” or “off” is it “red” or “green” etc.

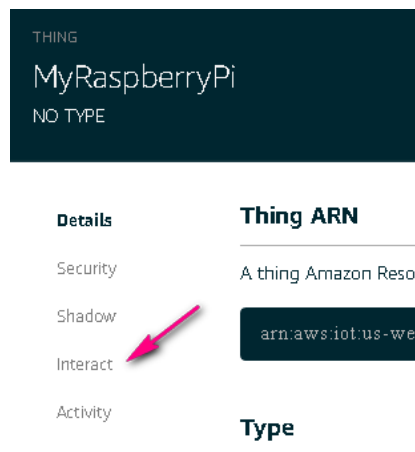
- c. Our “Thing” here is our RPI, so let’s name it as RaspberryPi and click “**Create thing**”



**Figure 23: Create Thing**

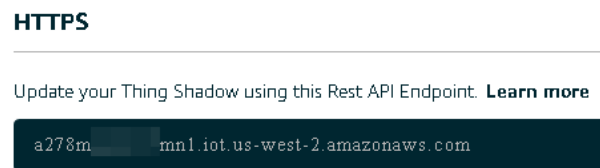
**Copy Rest API endpoint of your “Thing”**

- a. On the Details page, choose “**Interact**”.



**Figure 24: Details Page**

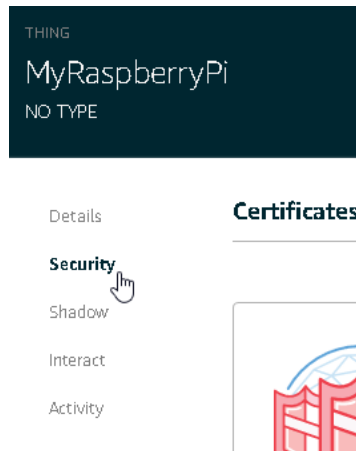
- b. Copy and paste the REST API endpoint into a notepad as you will need it later



**Figure 25: REST API**

## Create Certificate

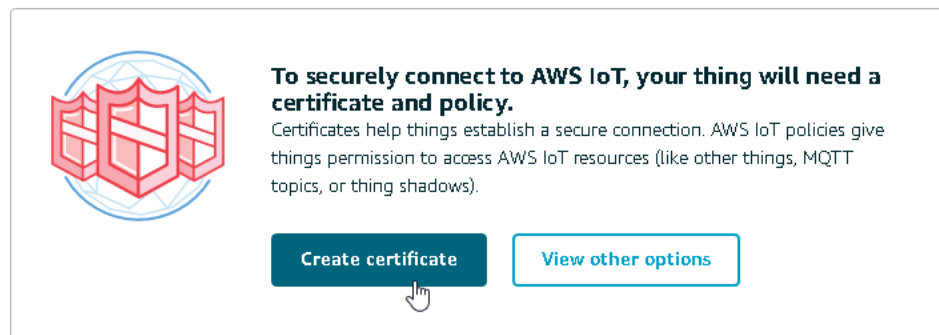
- a. Next select “**Security**”



**Figure 26: Security**

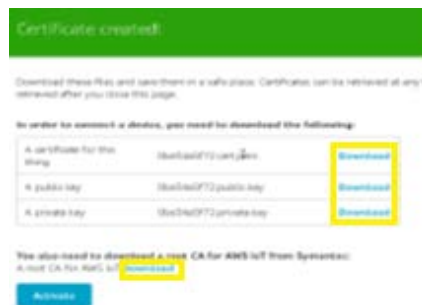
- b. Select “**Create certificate**” to generate an X.509 certificate and key pair

### **Certificates**



**Figure 27: Certificate Page**

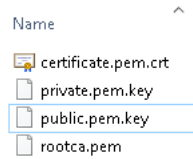
- c. After a while, you should see the screen as you see in the figure below, where there are total of 4 download links



**Figure 28: Certificated Created**

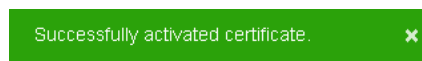


- d. Create a folder called **“devieSDK”** (or any name you preferred) and download all the 4 files into the folder and **rename them all with friendly names** like what you see in the figure below:



**Figure 29: Rename of files**

- e. Next, click the **“Activate”** button. Immediately, you should see **“Successfully activated certificate”** and the Activate button changes to **“Deactivate”**



**Figure 30: Activated Successfully**

### **Create a Security Policy for your RPI**

- a. Next, click on the **“Attach a policy”** button that is near the bottom right-hand corner of the page.



**Figure 31: Attach a policy**

- b. On the next page, choose **“Create new Policy”**



**Figure 32: New Policy**

- c. On the Create a policy page, key in the following configuration and then click **“Create”**

Create a policy to define a set of authorized actions. You can authorize actions on one or more resources (things).

Name

RaspberryPiSecurityPolicy

---

**Add statements**

Policy statements define the types of actions that can be performed by a resource.

|   |
|---|
| Action  |
| iot:*   |
| Resource ARN  |
| *   |
| Effect  |
| <input checked="" type="checkbox"/> Allow <input type="checkbox"/> Deny |

**Figure 33: Configuration Steps**

- d. You will see a page like the figure below, move ahead to clock the “**Back**” arrow to return to the previous page and continue with the instructions in the next section

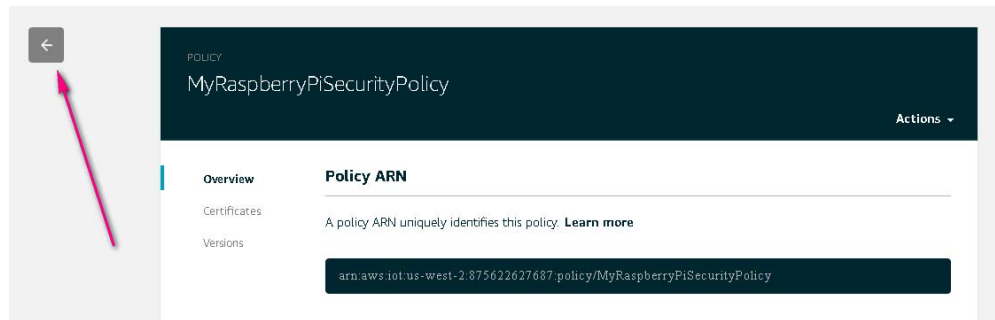


Figure 34: Going back to Previous Page

### Attach Security Policy and Thing to your Cert

- a. On the left nav bar, click “**Security**” than “**Certificates**”

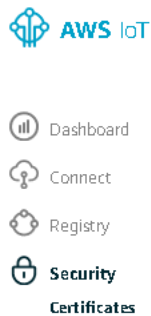


Figure 35: Nav Bar

- b. The X.509 certificate you created earlier is shown. Click the checkbox beside it, then click “**Actions**” button and choose “**Attach Policy**”

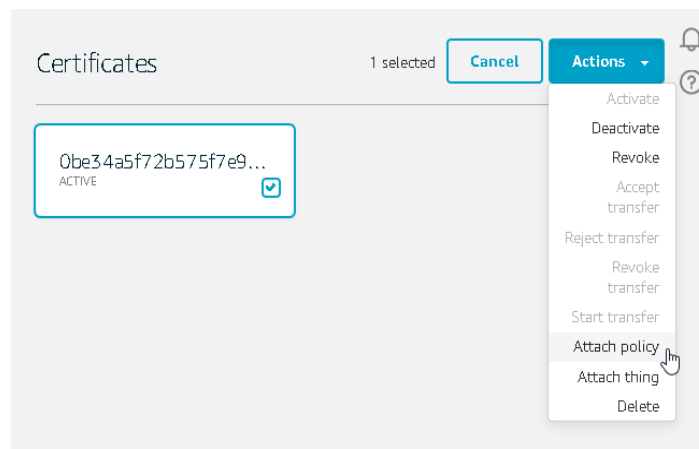
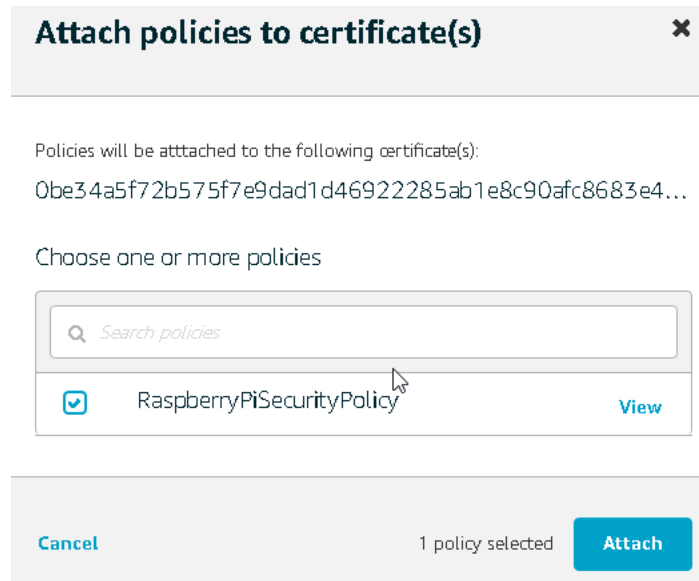


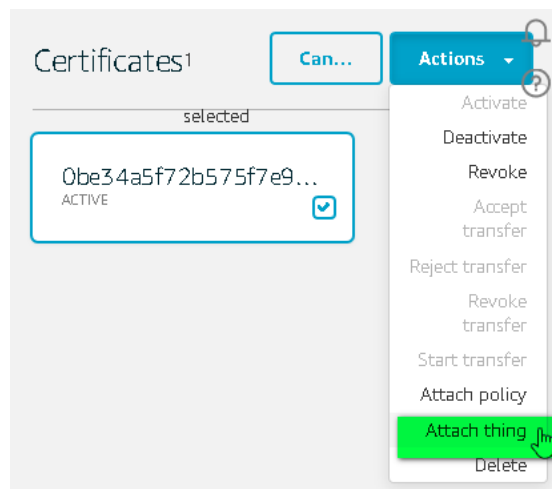
Figure 36: Attach Policy

- c. Check the “**RaspberryPiSecurityPolicy**” you created earlier and click “**Attach**” button.



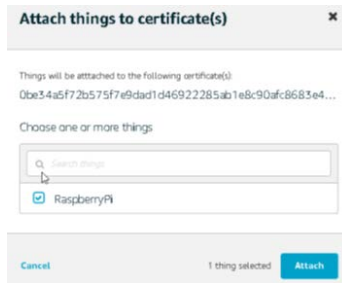
**Figure 37: Attach Policy to Certificate**

- d. Let's attach the “Thing” to this certificate. Click “**Actions**” button and choose “**Attach Thing**”



**Figure 38: Attach Thing**

- e. In the Attach things to certificate(s) dialog box, select the “**check box**” next to the thing you created to represent your Raspberry Pi, and then choose “**Attach**”



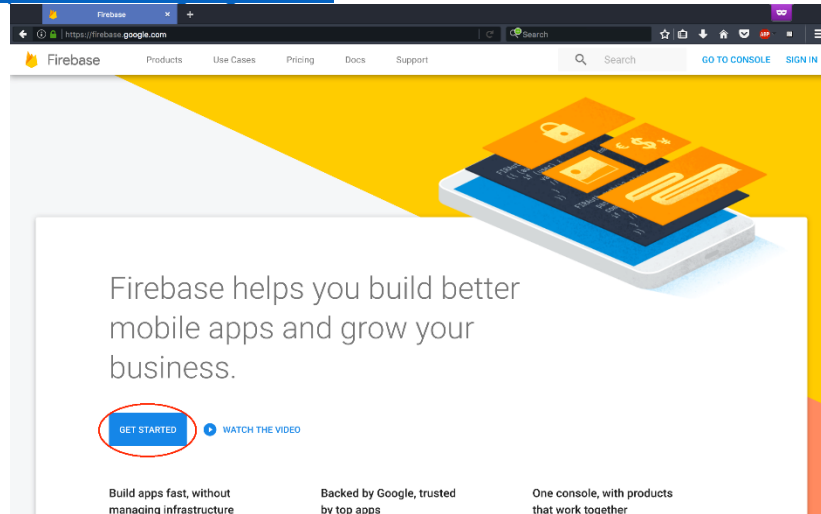
**Figure 39: Attach Thing to Raspberry Pi**

## Setup Firebase

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

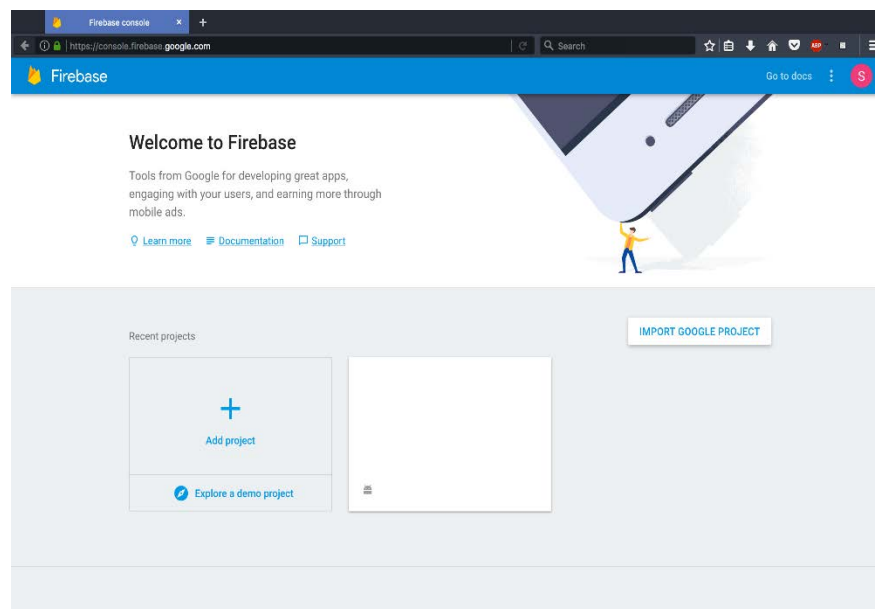
## Create Firebase Account

- a. Go to <https://firebase.google.com/> and click on the “GET STARTED”



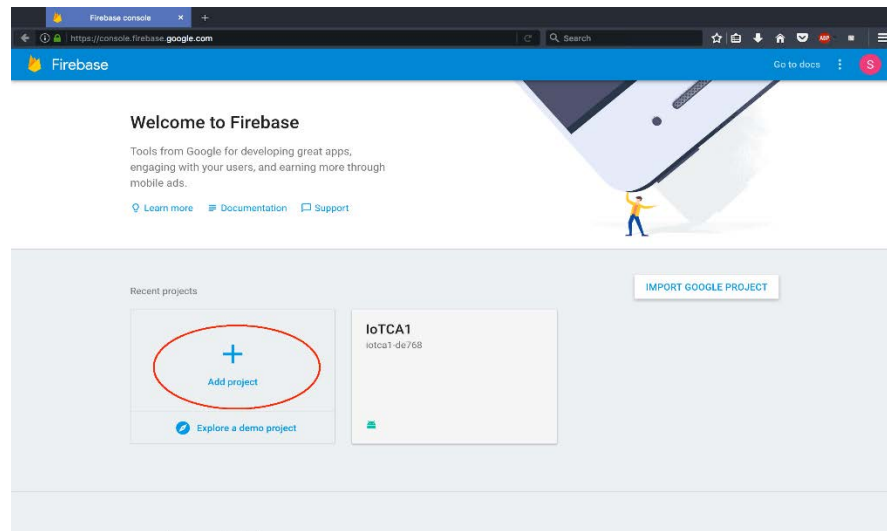
**Figure 40: Get Started**

- b. After clicking “GET STARTED”, Sign in or create a new account. You will see the dashboard as you see in figure below:



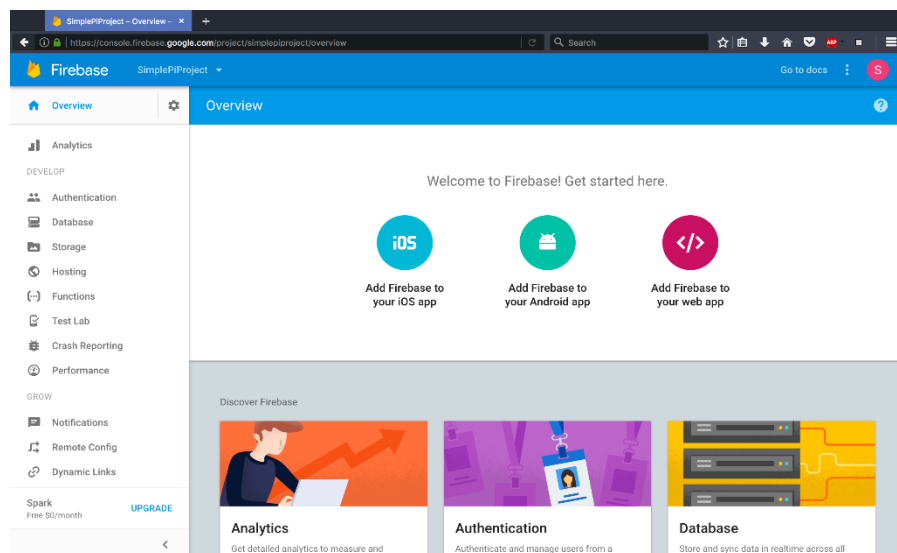
**Figure 41: Dashboard Page**

- c. Click on “**Add project**”, you will see a dialog box appear, enter your project a name and select your country/region. Then click on ‘**Create Project**’;



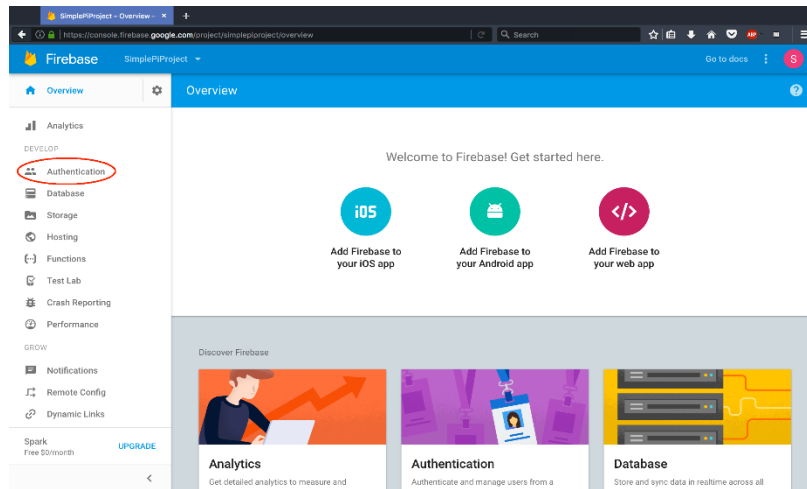
**Figure 42: Add Project**

- d. Look the figure below, that how the Firebase console should look like



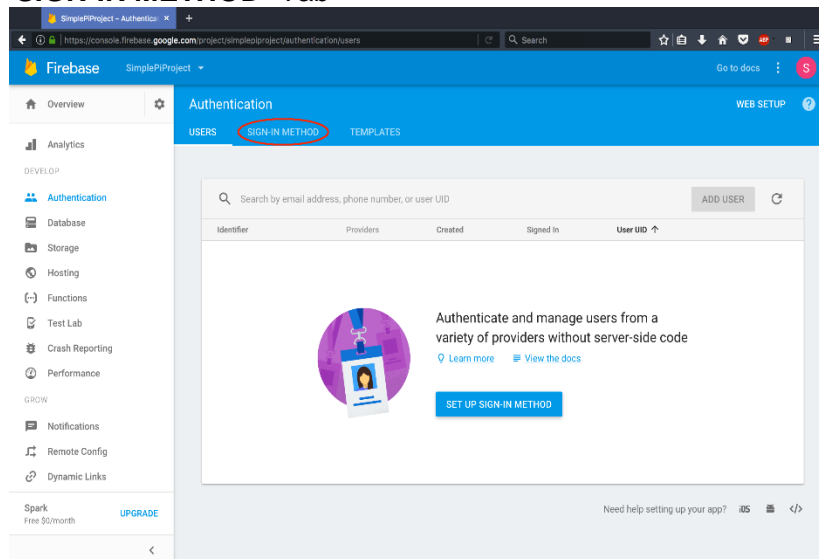
**Figure 43: Firebase Console**

- e. Click on ‘**Authentication**’ tab on the left.



**Figure 44: Authentication Tab**

- f. Click on “**SIGN-IN METHOD**” Tab



**Figure 45: Sign-In Method**

- g. The config for Firebase can be found in your Firebase Console by clicking on “**Add Firebase to your web app**”.

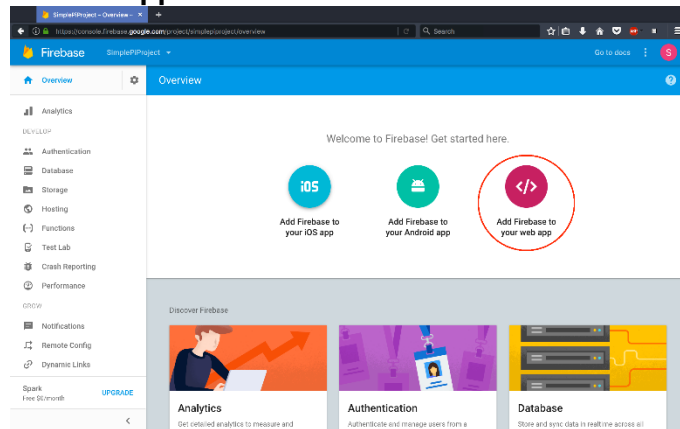


Figure 46: Add Firebase to Web

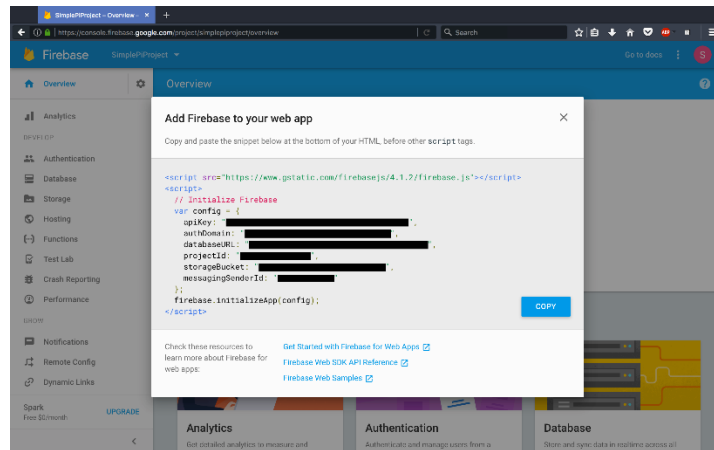


Figure 47: API Key for Firebase

- h. Click on “**Email/Password**” and enable it then click on “**SAVE**”.

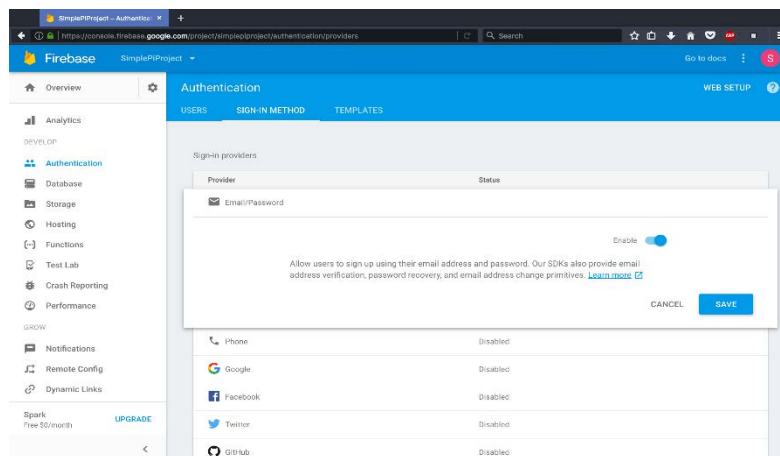


Figure 48: Email/Password



- i. Click on the **“Setting”** icon on the left

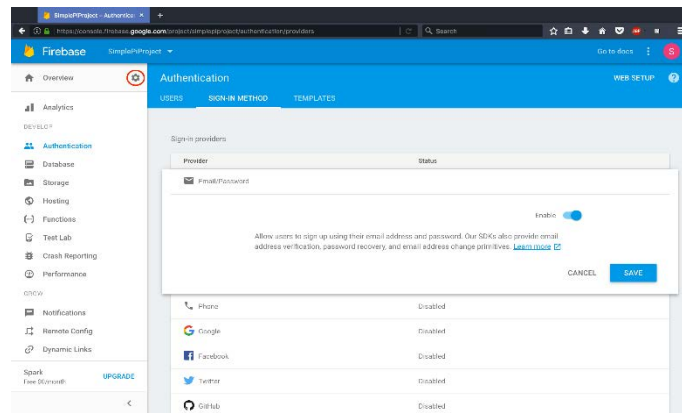


Figure 49: Setting

- j. Now click on **“Project setting”**

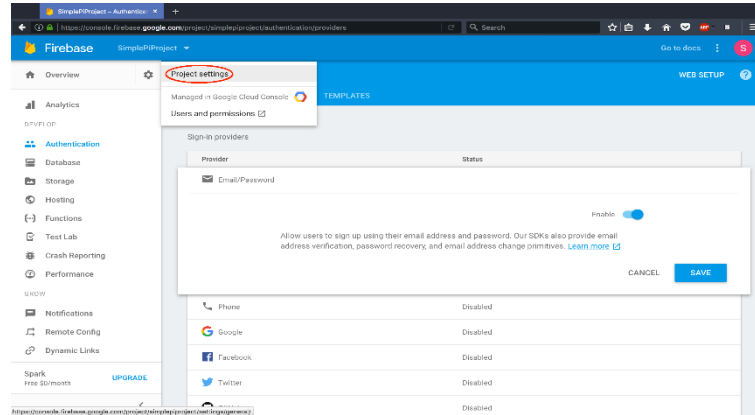
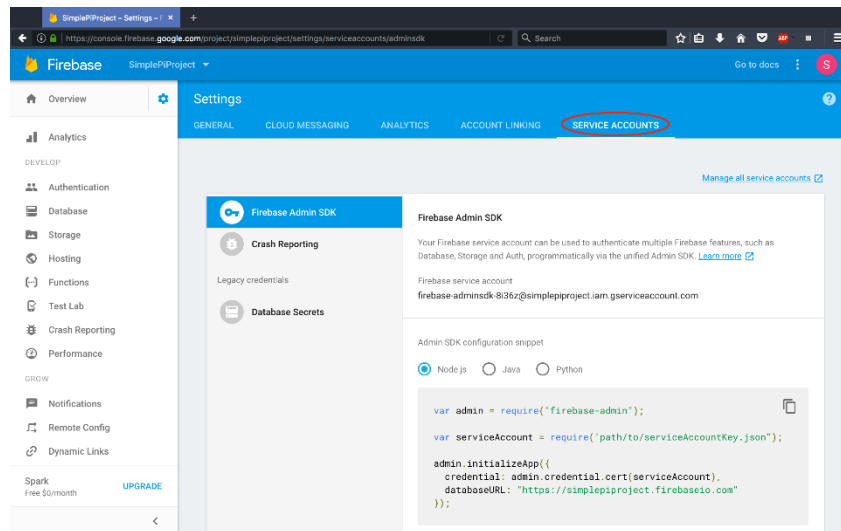


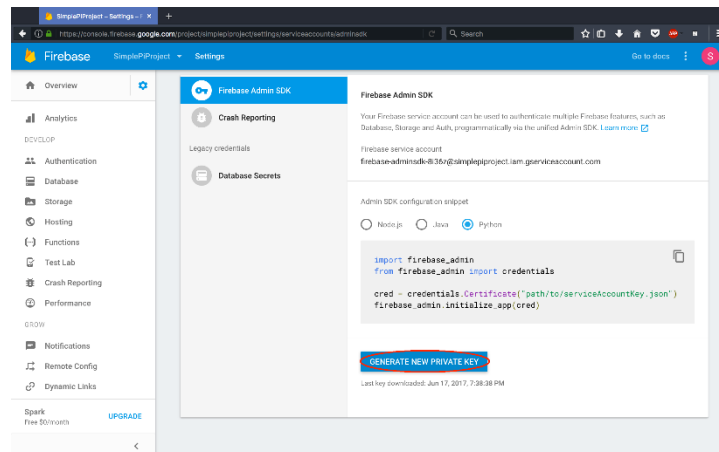
Figure 50: Project Setting

- k. Click on “**Service Accounts**” tab.



**Figure 51: Service Account**

- l. Scroll down and click on ‘**GENERATE NEW PRIVATE KEY**’.



**Figure 52: Private Key**

- m. Save the generated key (it will be a .json file) somewhere, we will be using that later.  
n. We will now go back to the Authentication Tab and create an account that we’ll be using for this project.

**Table 1: Email and Password**

|                          |                    |
|--------------------------|--------------------|
| Email: auth@firebase.com | Password: P@ssw0rd |
|--------------------------|--------------------|

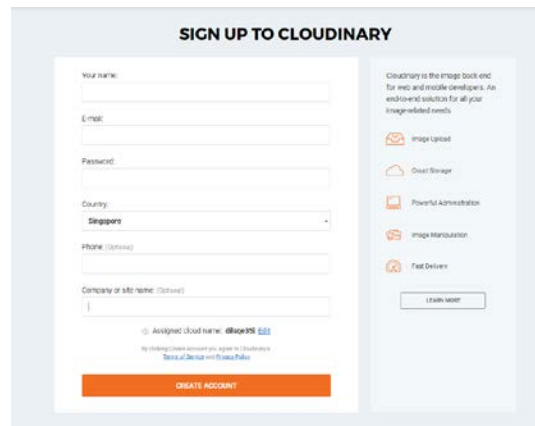
- o. Now we’re all set for our database.

## Setup Cloudinary

Cloudinary is the image back-end for web and mobile developers. An end-to-end solution for all your image-related needs.

### Create Cloudinary Account

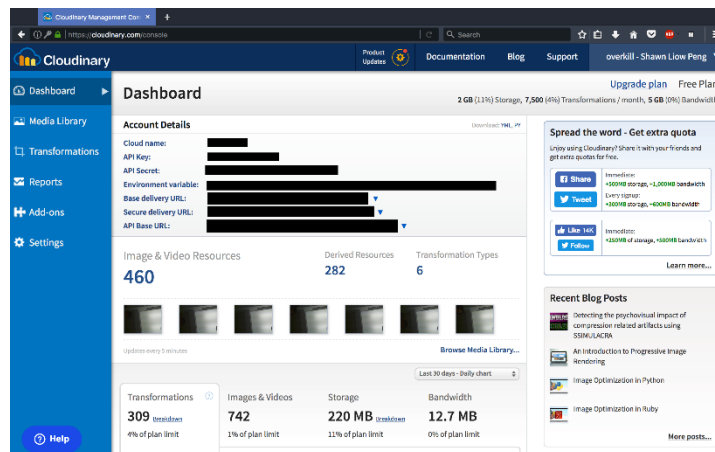
- Go to <https://cloudinary.com/users/register/free> and add your information and proceed on to click “**CREATE ACCOUNT**”



The image shows the 'SIGN UP TO CLOUDINARY' page. It features a registration form on the left with fields for 'Your name', 'Email', 'Password', 'Country' (a dropdown menu with 'Singapore' selected), 'Phone' (marked as optional), and 'Company or site name' (also marked as optional). Below the form is an orange 'CREATE ACCOUNT' button. To the right of the form, there is a brief description of Cloudinary as an image back-end solution, followed by icons and text for 'Image Upload', 'Cloud Storage', 'Powerful Administration', 'Image Manipulation', and 'Fast Delivery'. At the bottom right of this section is a 'Learn more' button.

**Figure 53: Signup Page**

- This is what the dashboard should look like



**Figure 54: Dashboard**

- Take note of everything under “**Account Details**”, we will be using them later.
- Now we’re done setting up Cloudinary for use

## **Setting up our Raspberry Pi**

For this project, we will be using Python 2.7.9. You can check your Python version by running the terminal and enter “**python --version**” command. If you do not have version, please install it.

Run the following commands on your terminal before we start with the development process:

→sudo pip install pycoudinary

→sudo pip install pyrebase

→sudo pip install twilio

→sudo pip install twilio-python

→sudo apt-get install python-gpiozero

→sudo pip install AWSIoTPythonSDK