Universidade do Minho
Departamento de Informática

SENSORIZAÇÃO E AMBIENTE

MESTRADO EM ENGENHARIA INFORMÁTICA, 1º ANO - Perfil SI

Universidade do Minho
Departamento de Informática

Concepts and Platforms

Agenda

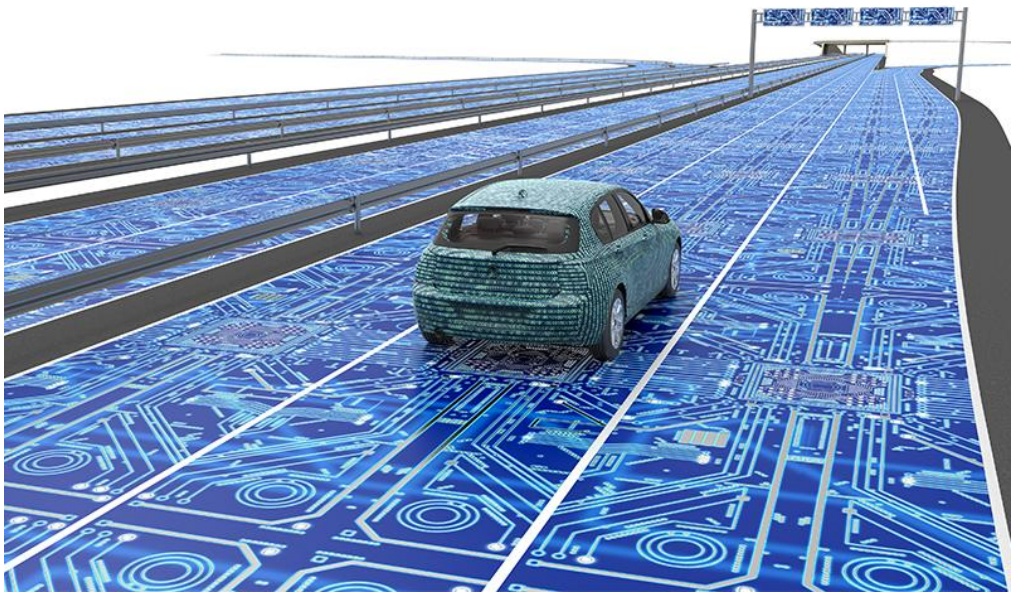
- Concepts
  - Ambient Intelligence (AmI)
  - Internet of Things (IoT)
  - Internet of People (IoP)
  - Smart Cities
  - DIKW pyramid
  - Sensorization
- Platforms
  - Adafruit IO
  - Firebase
  - IFTT
  - MQTT
  - Adafruit IO + IFTTT
  - Adafruit IO + Java
  - Adafruit IO + JavaScript
  - Adafruit IO + Arduino
- Hands On

Universidade do Minho
Departamento de Informática

Concepts

Ambient Intelligence (AmI)

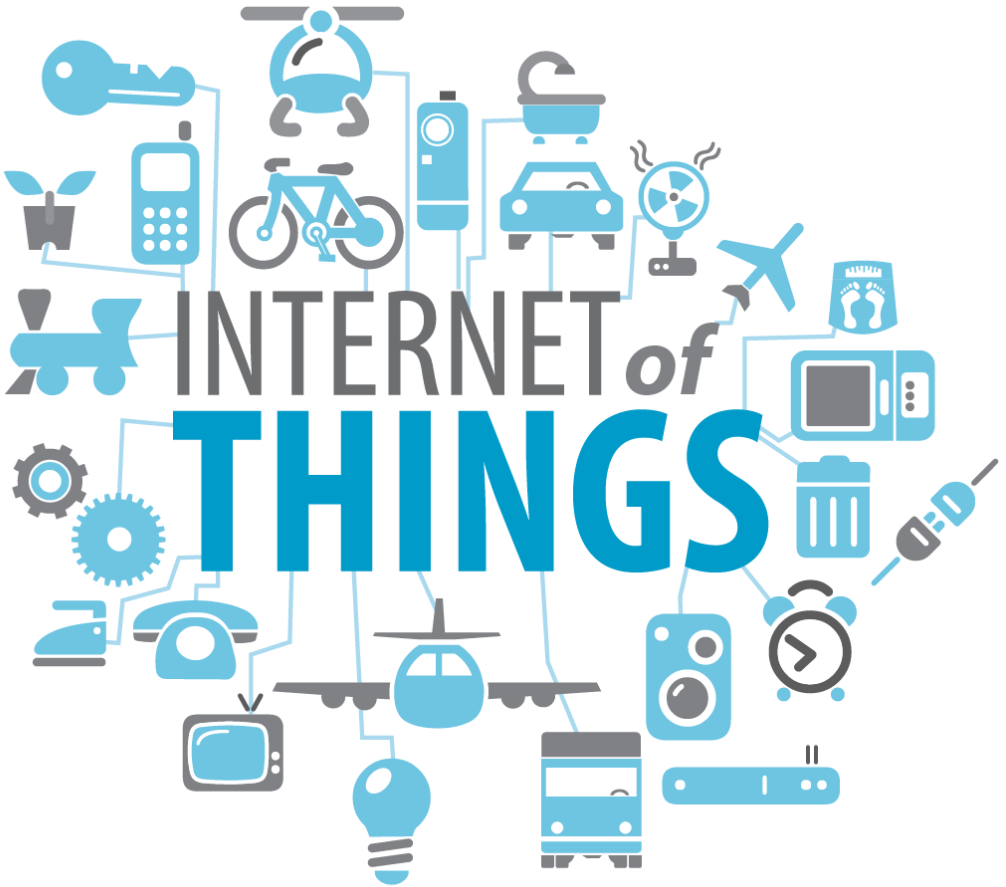
- **Ambient Intelligence** refers to environments that are **sensitive** and **responsive** to the **presence of people** in a non-intrusive manner. As devices grow smaller, connected and more integrated with the environment, the technology disappears into our surroundings.



Ambient Intelligence

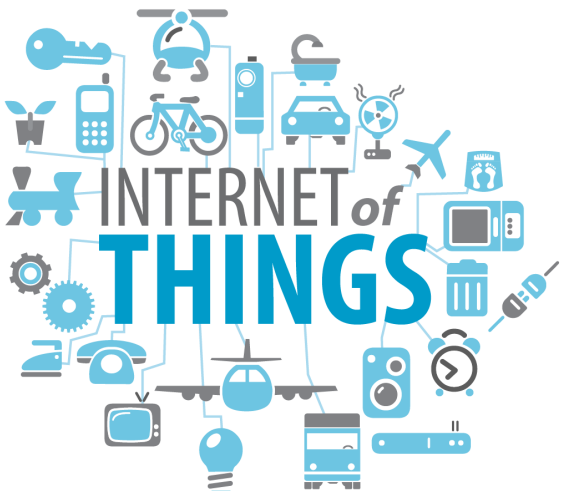
- Data Acquisition
  - Sensors
  - Services
  - Data processing
- Reasoning
  - Data Modeling
  - Machine Learning
  - Decision Models
- Actuation
  - Notifications
  - Interactions
  - Actions

Internet of Things (IoT)



Internet of Things (IoT)

- An open and comprehensive **network of intelligent objects** that have the capacity to auto-organize, share information, data and resources, reacting and acting in face of situations, and changes in the environment.



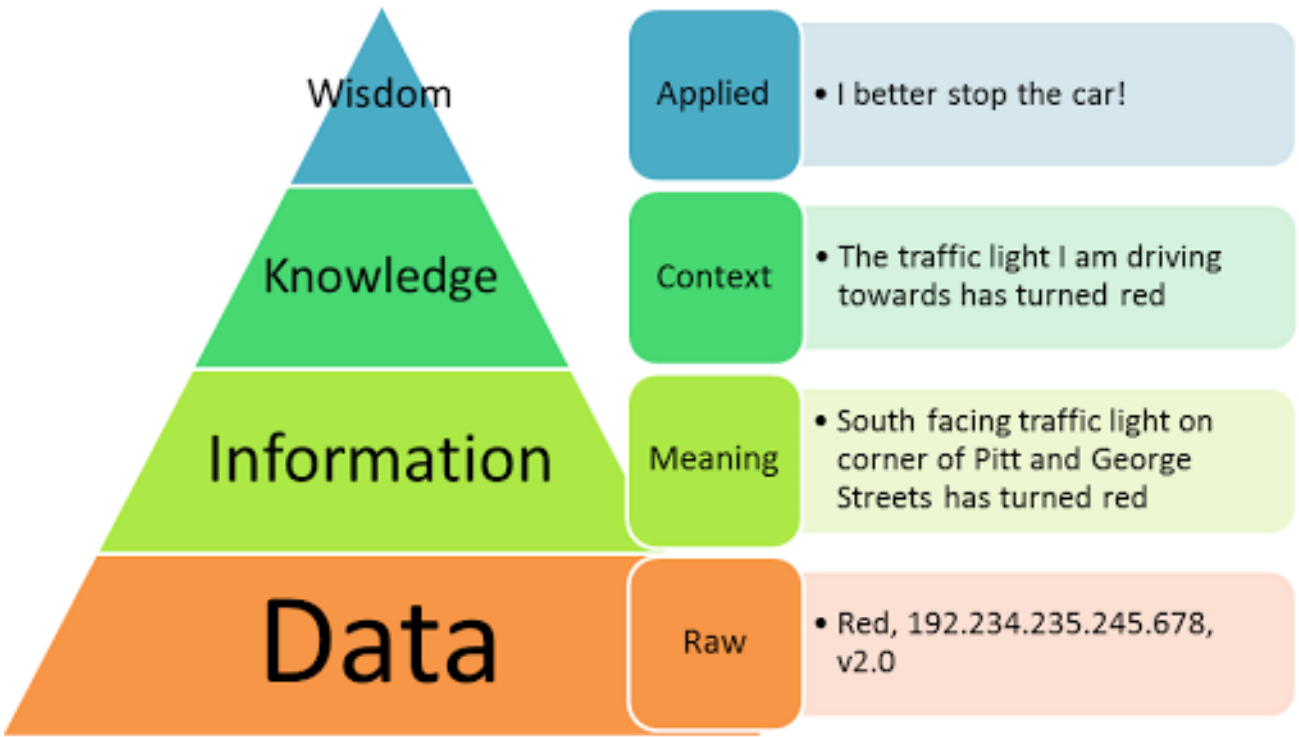
Internet of People (IoP)

Internet of People (IoP)

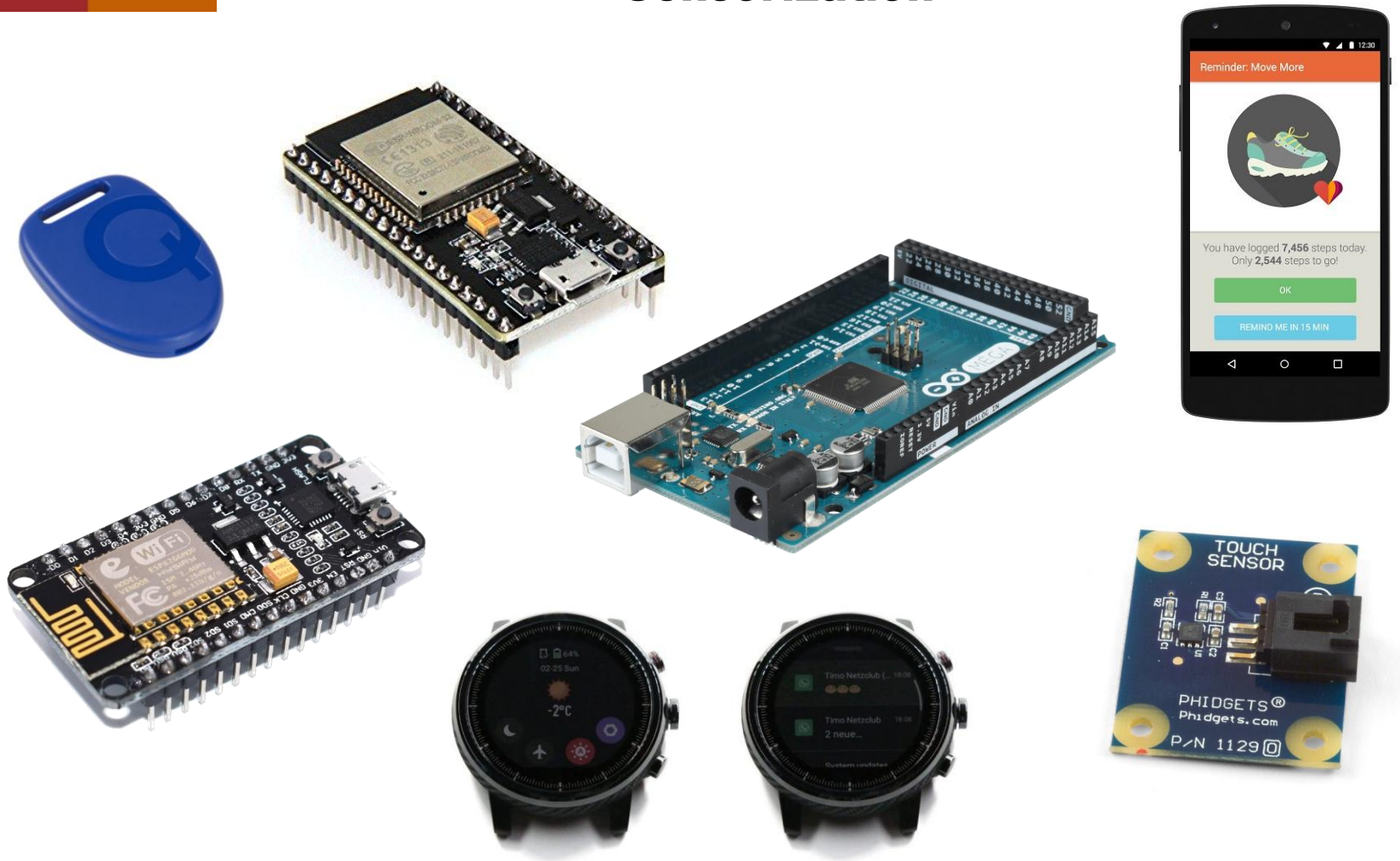
- A dynamic global **network** where **things and people** communicate and understand each other; where everyone and everything can sense the other and the world, and act on such knowledge and information, aiming to enhance **people's quality of life**.

Smart Cities

Data Pyramid (DIKW pyramid)



Sensorization



Sensorization

- A **sensor** is something that is able to **percept phenomena that is being observed** and translate **its state**.

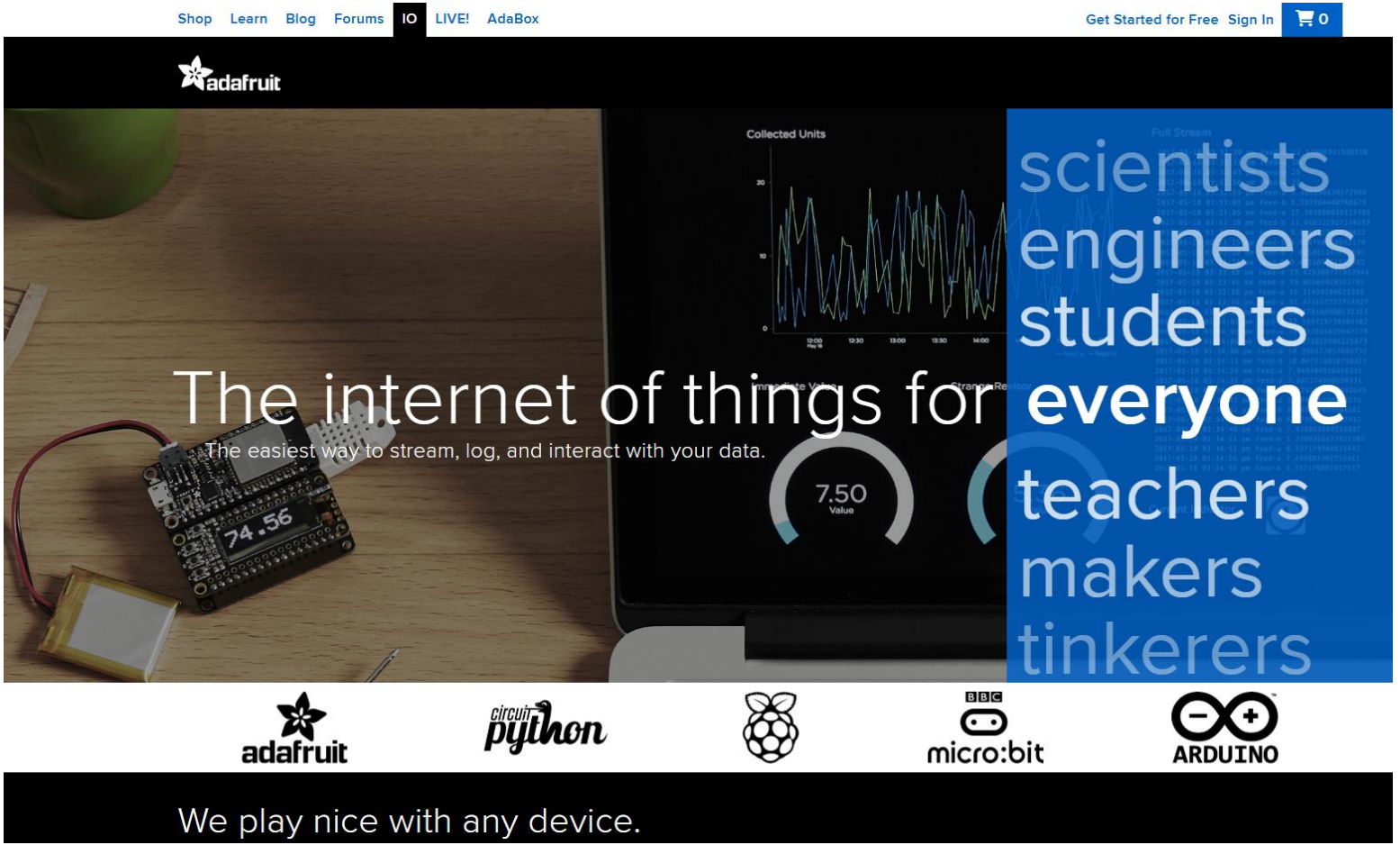
- Traditionally sensors were **physical** and observed physical phenomena, but sensors may also be **virtual**:
  - Access to web API
  - Mathematical formulae

- Currently, data fusion can also infer virtual assets such as:
  - Emotions
  - Wellbeing
  - Sustainability
  - Happiness

Universidade do Minho
Departamento de Informática

Platforms

Adafruit IO



https://io.adafruit.com/

Adafruit IO

- Adafruit.io is a **cloud service**

- It's meant primarily for **storing** and then **retrieving data** (using **feeds**)
    - It can also display our data in real-time

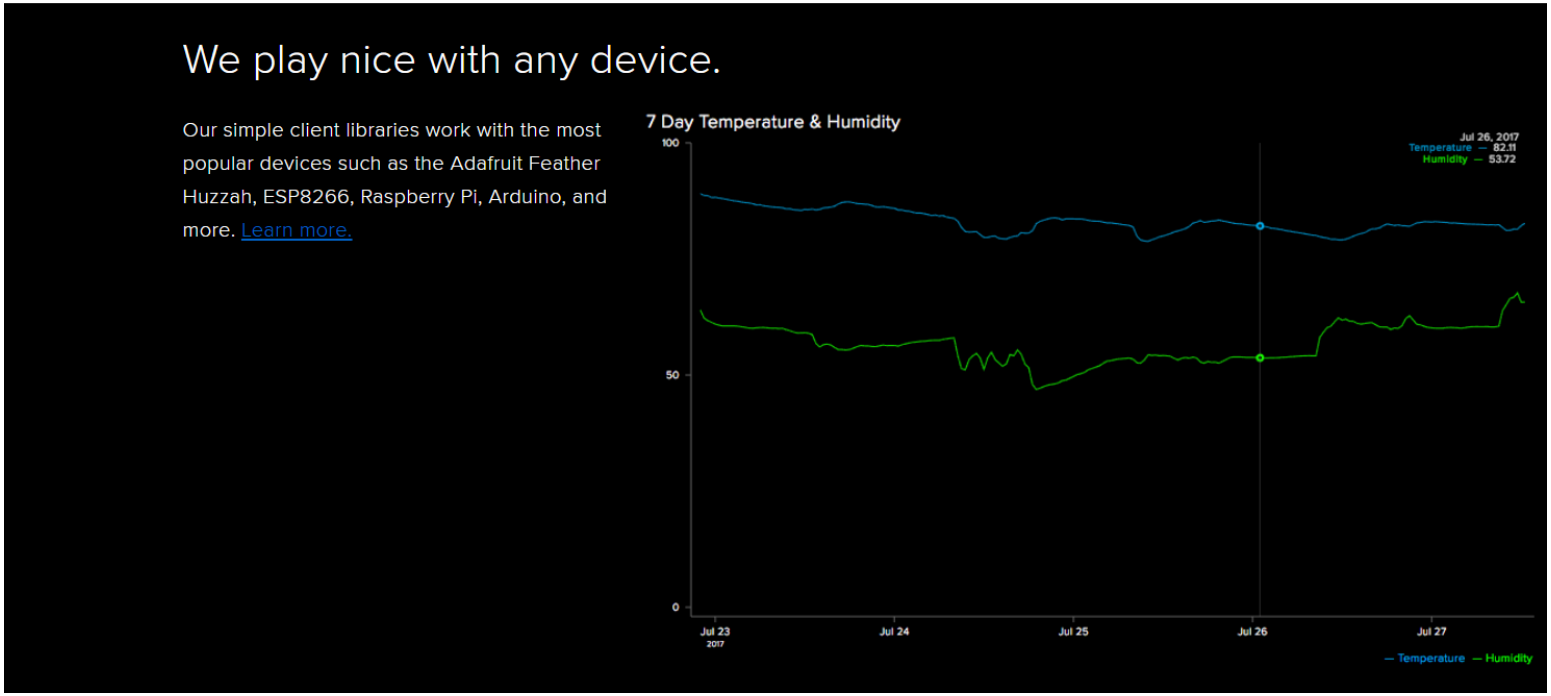
- **Feeds** are the **core** of Adafruit IO holding both the uploaded data (data pushed to Adafruit IO by sensors) and its meta-data

- Nice and intuitive **dashboards** integrated into Adafruit IO

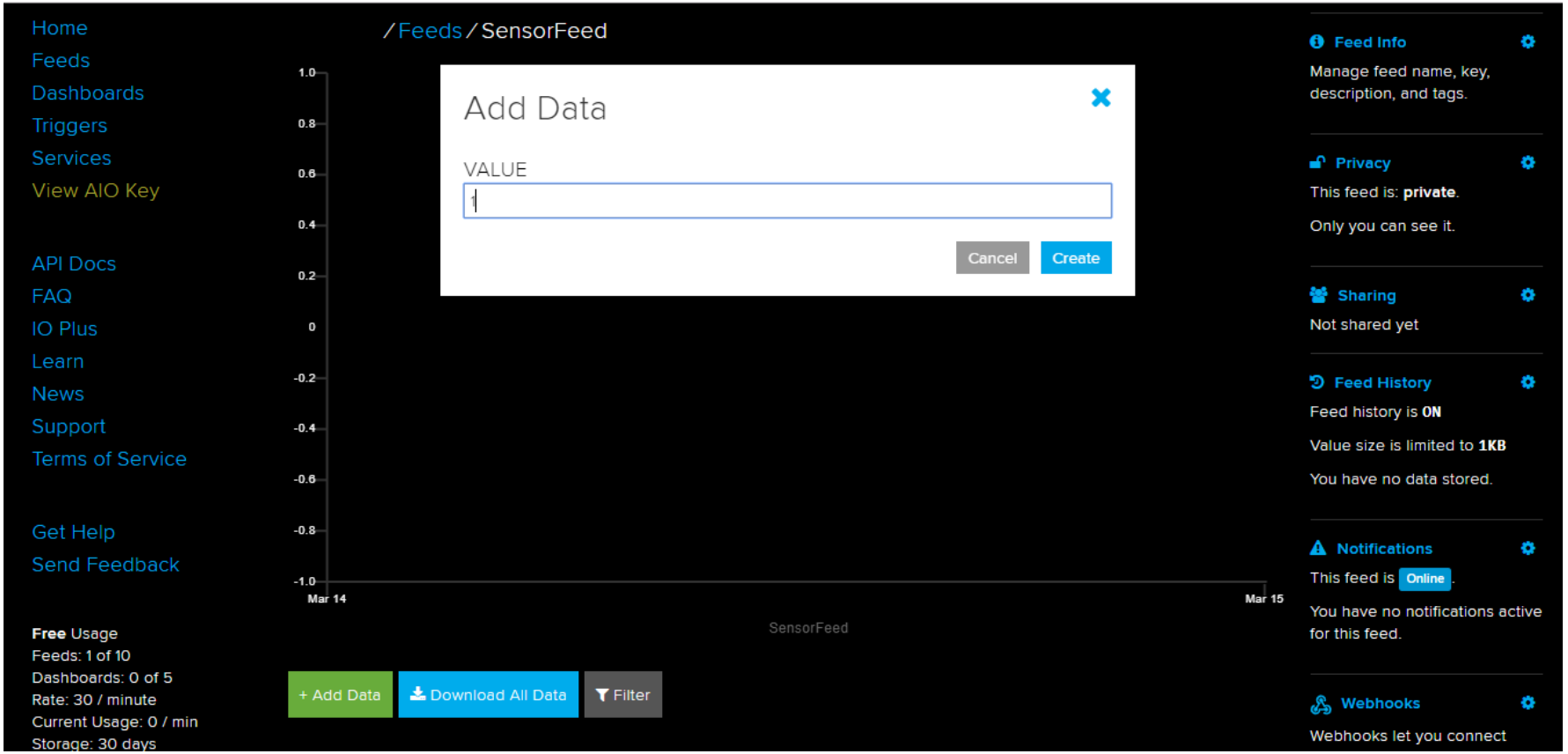
- Allows to define **triggers** to **control and react to the data** (ex.: triggers to email when a temperature sensor gets too hot)

- Integration with IFTTT

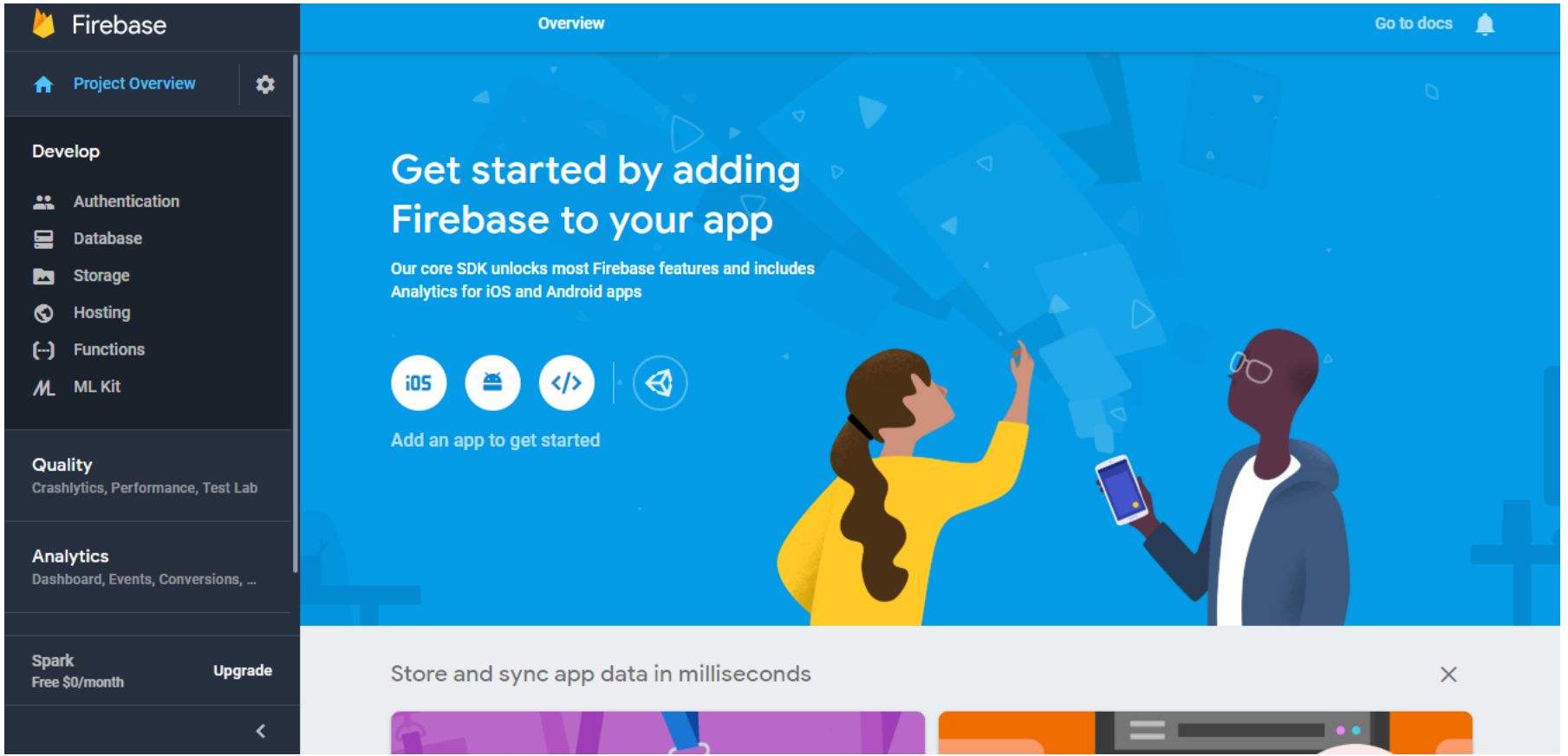
Adafruit IO


We play nice with any device.
Our simple client libraries work with the most popular devices such as the Adafruit Feather Huzzah, ESP8266, Raspberry Pi, Arduino, and more. Learn more.
7 Day Temperature & Humidity
Jul 26, 2017
Temperature — 82.11
Humidity — 53.72
100
50
0
Jul 23
2017
Jul 24
Jul 25
Jul 26
Jul 27
— Temperature — Humidity

Adafruit IO


Home
Feeds
Dashboards
Triggers
Services
View AIO Key
API Docs
FAQ
IO Plus
Learn
News
Support
Terms of Service
Get Help
Send Feedback
Free Usage
Feeds: 1 of 10
Dashboards: 0 of 5
Rate: 30 / minute
Current Usage: 0 / min
Storage: 30 days
/Feeds/SensorFeed
Add Data
VALUE
1
Cancel
Create
1.0
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1.0
Mar 14
Mar 15
SensorFeed
+ Add Data
Download All Data
Filter
Feed Info
Manage feed name, key, description, and tags.
Privacy
This feed is: private.
Only you can see it.
Sharing
Not shared yet
Feed History
Feed history is ON
Value size is limited to 1KB
You have no data stored.
Notifications
This feed is Online.
You have no notifications active for this feed.
Webhooks
Webhooks let you connect

Firebase



https://console.firebase.google.com/

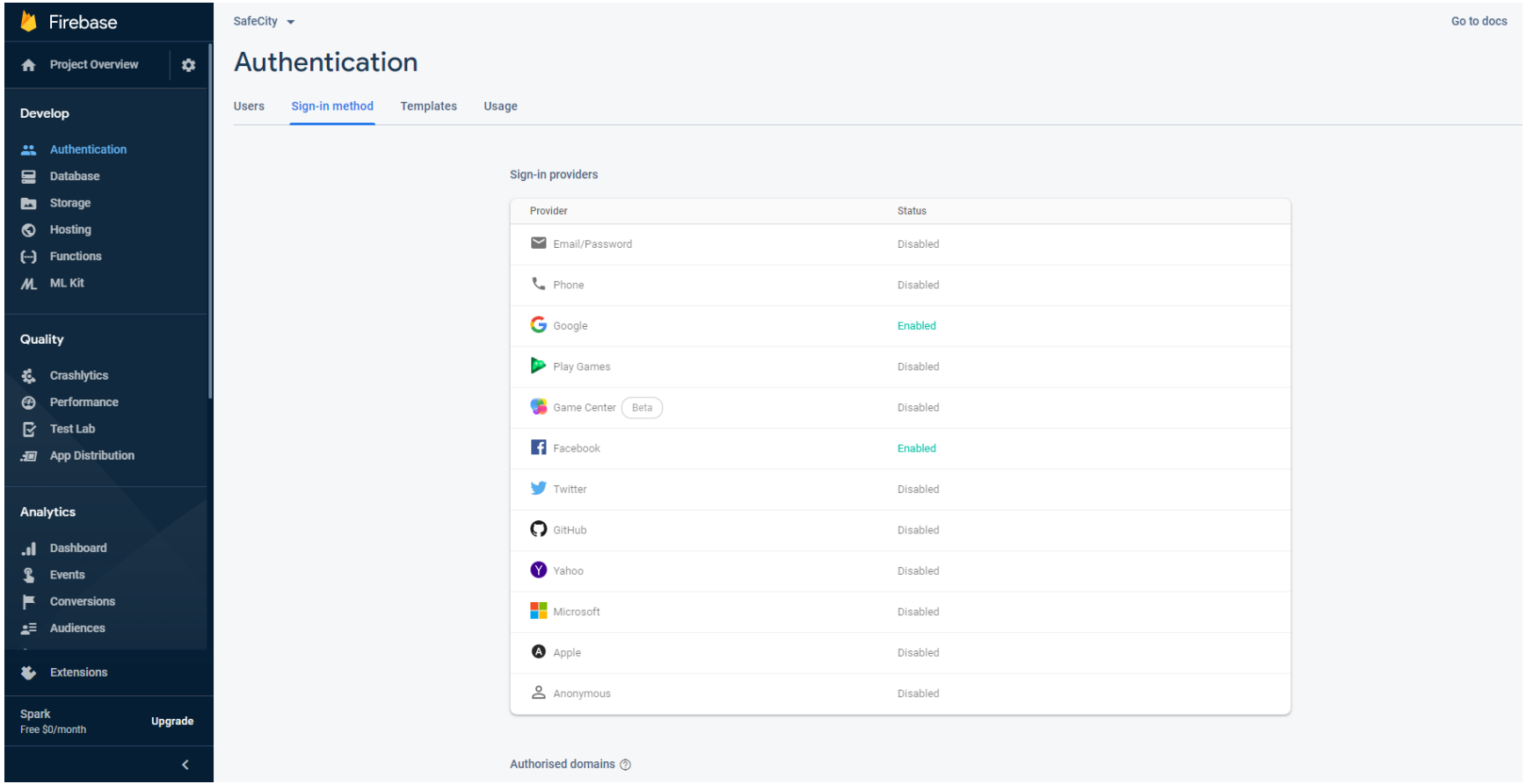
Firebase

- All the tools we need to build a successful app
  - For example, they can handle all the authentication process for us (it can take up to months to set up our own authentication system)

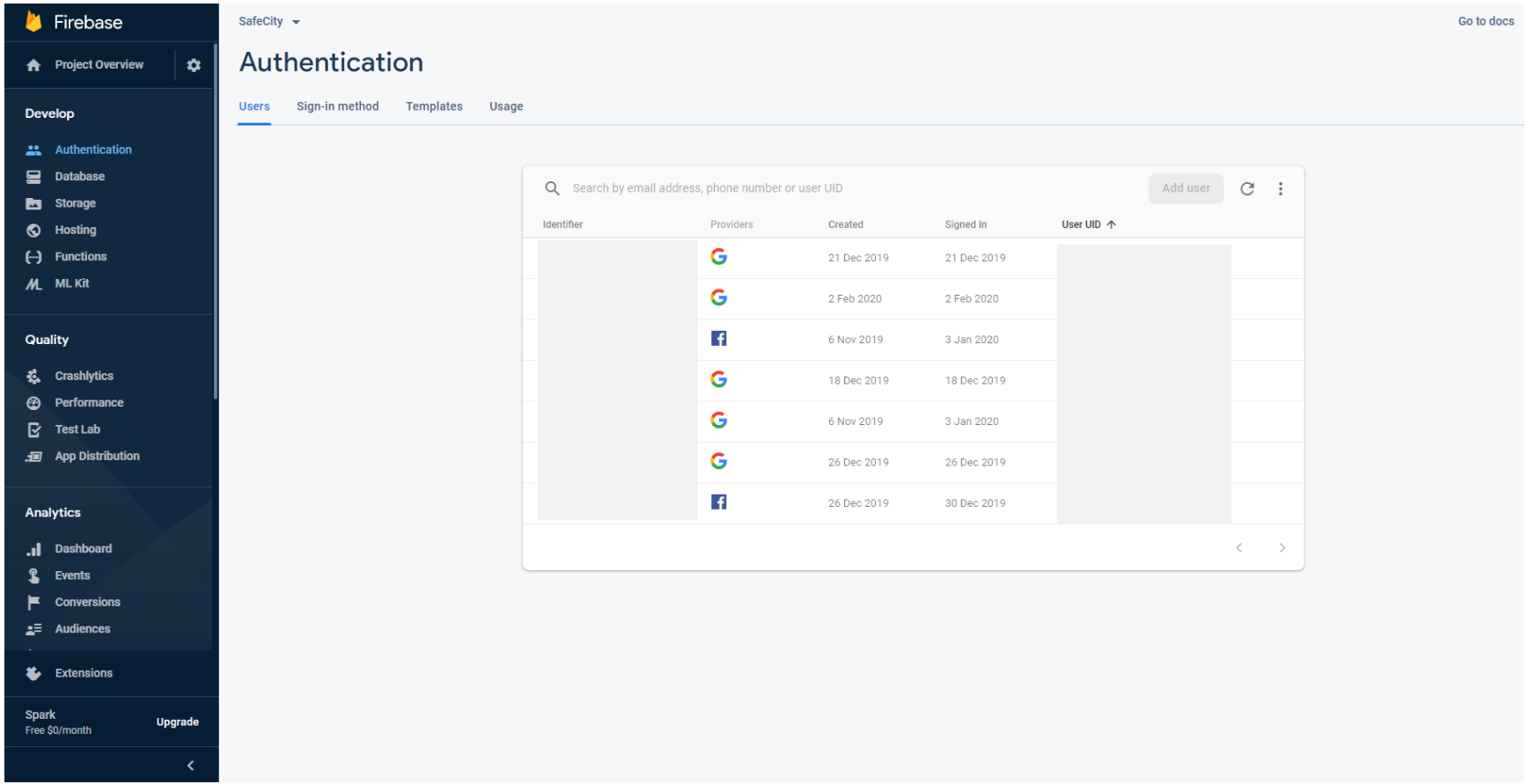
- Handles backend, database, user engagement, scalability, etc.

- **Mobile** and **web app development platform** that provides developers with a set of tools and services to help them develop high-quality apps

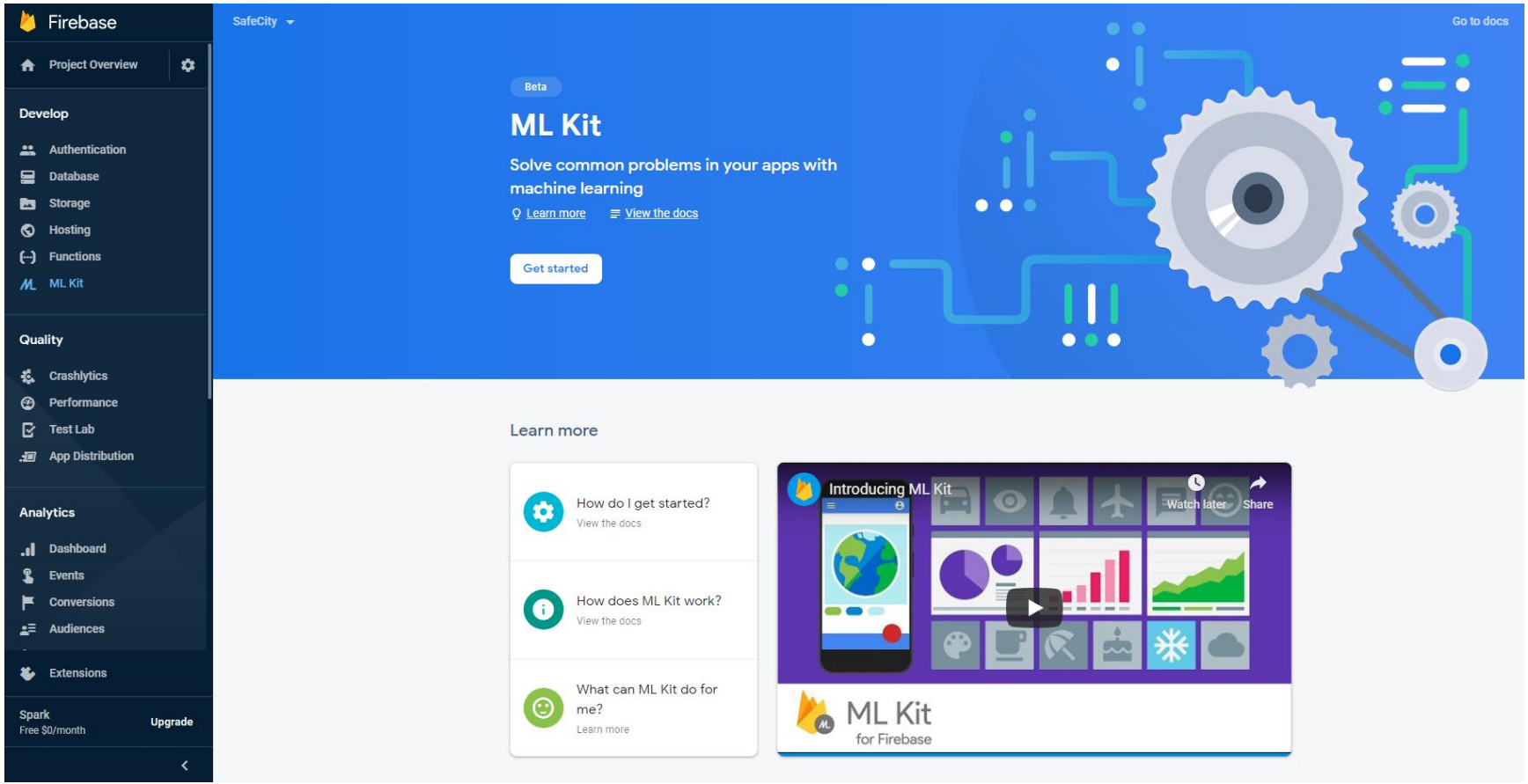
Firebase


Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics
Performance
Test Lab
App Distribution
Analytics
Dashboard
Events
Conversions
Audiences
Extensions
Spark
Free $0/month
Upgrade
SafeCity
Go to docs
Authentication
Users
Sign-in method
Templates
Usage
Sign-in providers
Provider
Status
Email/Password
Disabled
Phone
Disabled
Google
Enabled
Play Games
Disabled
Game Center
Beta
Disabled
Facebook
Enabled
Twitter
Disabled
GitHub
Disabled
Yahoo
Disabled
Microsoft
Disabled
Apple
Disabled
Anonymous
Disabled
Authorised domains

Firebase


Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics
Performance
Test Lab
App Distribution
Analytics
Dashboard
Events
Conversions
Audiences
Extensions
Spark
Free $0/month
Upgrade
SafeCity
Go to docs
Authentication
Users
Sign-in method
Templates
Usage
Search by email address, phone number or user UID
Add user
Identifier
Providers
Created
Signed In
User UID
21 Dec 2019
21 Dec 2019
2 Feb 2020
2 Feb 2020
6 Nov 2019
3 Jan 2020
18 Dec 2019
18 Dec 2019
6 Nov 2019
3 Jan 2020
26 Dec 2019
26 Dec 2019
26 Dec 2019
30 Dec 2019

Firebase


Firebase
SafeCity
Go to docs
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics
Performance
Test Lab
App Distribution
Analytics
Dashboard
Events
Conversions
Audiences
Extensions
Spark
Free $0/month
Upgrade
Beta
ML Kit
Solve common problems in your apps with machine learning
Learn more
View the docs
Get started
Learn more
How do I get started?
View the docs
How does ML Kit work?
View the docs
What can ML Kit do for me?
Learn more
Introducing ML Kit
Watch later
Share
ML Kit
for Firebase

Firebase

Firebase | Products | Use Cases | Pricing | Docs | Support | Search | English

Overview | Guides | Reference | Samples | Libraries

Guides

- Get started with Firebase
- Manage your Firebase projects
- Prototype and test with Emulator Suite β
- Analytics
- Extensions β

DEVELOP

- Authentication
- Realtime Database
- Cloud Firestore
- Storage
- Hosting
- Cloud Functions
- ML Kit β
  - Introduction
  - Vision
    - Recognize text
    - Detect faces
    - Scan barcodes
    - Label images
    - Detect and track objects

How does it work?

ML Kit makes it easy to apply ML techniques in your apps by bringing Google's ML technologies, such as the Google Cloud Vision API, TensorFlow Lite, and the Android Neural Networks API together in a single SDK. Whether you need the power of cloud-based processing, the real-time capabilities of mobile-optimized on-device models, or the flexibility of custom TensorFlow Lite models, ML Kit makes it possible with just a few lines of code.

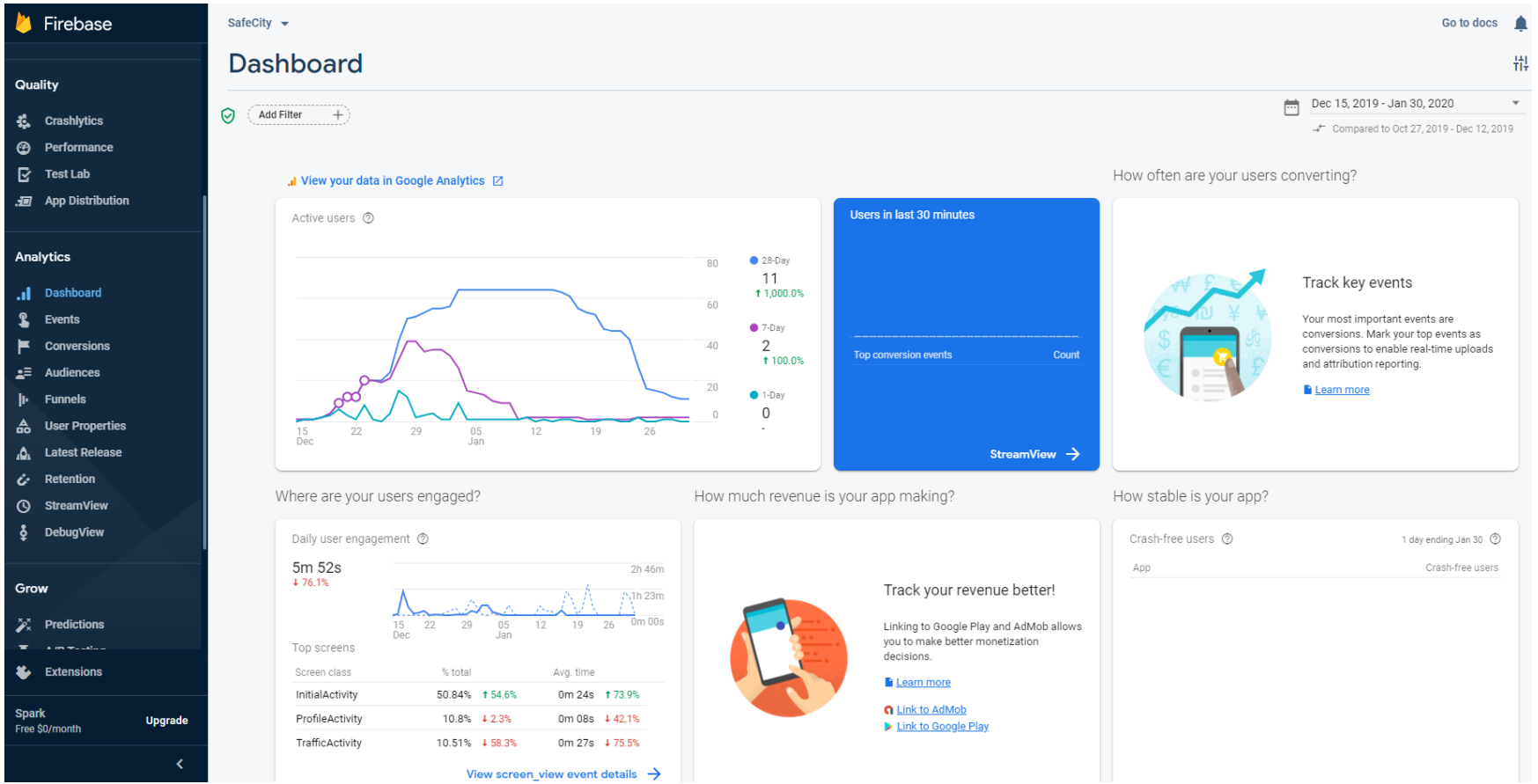
What features are available on device or in the cloud?

Feature	On-device	Cloud
Text recognition	✓	✓
Face detection	✓	
Barcode scanning	✓	
Image labeling	✓	✓
Object detection & tracking	✓	
Landmark recognition		✓
Language identification	✓	
Translation	✓	
Smart Reply	✓	
AutoML model inference	✓	
Custom model inference	✓	

★ Use of ML Kit to access Cloud ML functionality is subject to the Google Cloud Platform License Agreement and Service Specific Terms, and billed accordingly. For billing information, see the Firebase Pricing page.

Contents

Firebase

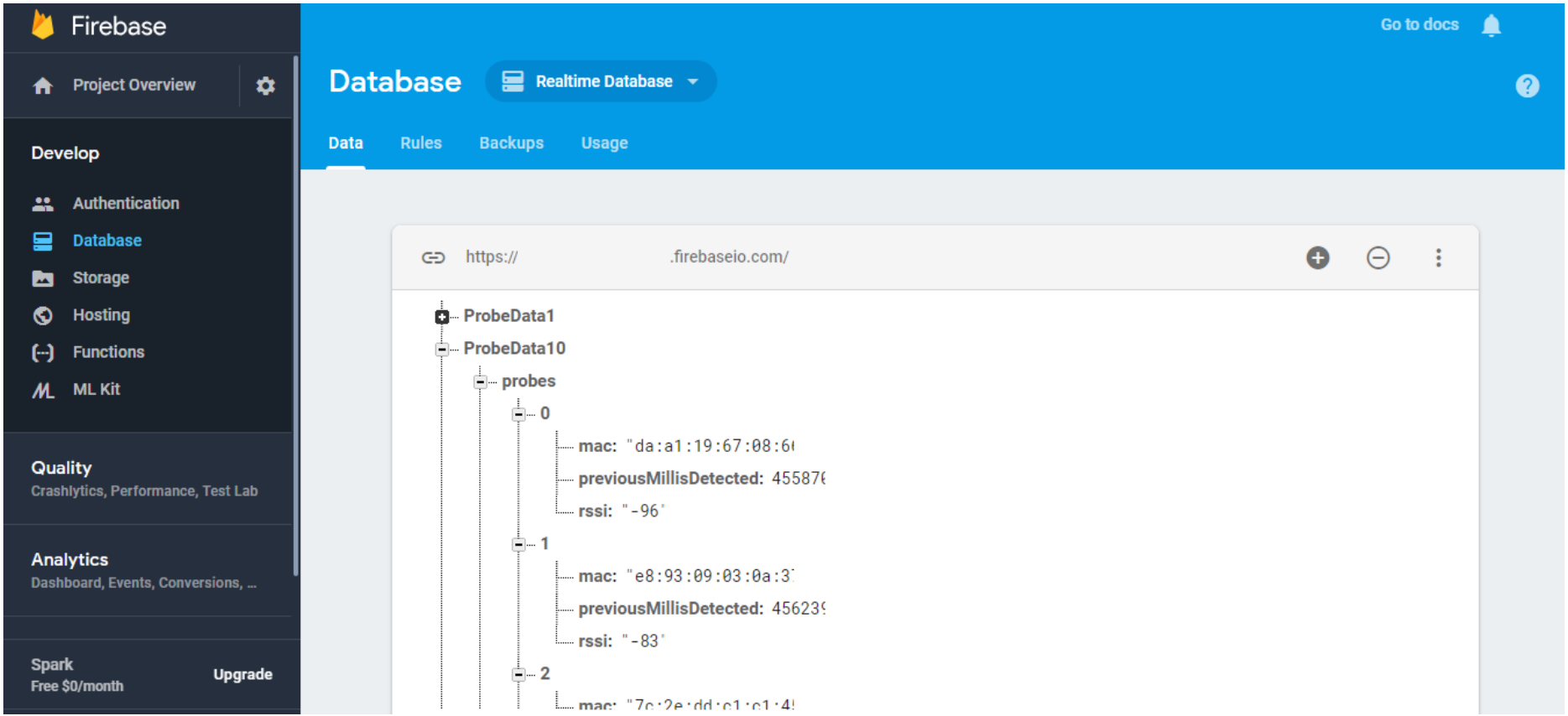

Firebase
SafeCity
Go to docs
Dashboard
Quality
Crashlytics
Performance
Test Lab
App Distribution
Analytics
Dashboard
Events
Conversions
Audiences
Funnels
User Properties
Latest Release
Retention
StreamView
DebugView
Grow
Predictions
Extensions
Spark
Free $0/month
Upgrade
Add Filter
Dec 15, 2019 - Jan 30, 2020
Compared to Oct 27, 2019 - Dec 12, 2019
View your data in Google Analytics
How often are your users converting?
Active users
28-Day
11
1,000.0%
7-Day
2
100.0%
1-Day
0
Users in last 30 minutes
Top conversion events
Count
StreamView
Track key events
Your most important events are conversions. Mark your top events as conversions to enable real-time uploads and attribution reporting.
Learn more
Where are your users engaged?
How much revenue is your app making?
How stable is your app?
Daily user engagement
5m 52s
76.1%
Top screens
Screen class
% total
Avg. time
InitialActivity
50.84%
54.6%
0m 24s
73.9%
ProfileActivity
10.8%
2.3%
0m 08s
42.1%
TrafficActivity
10.51%
58.3%
0m 27s
75.5%
View screen_view event details
Track your revenue better!
Linking to Google Play and AdMob allows you to make better monetization decisions.
Learn more
Link to AdMob
Link to Google Play
Crash-free users
1 day ending Jan 30
App
Crash-free users

Firebase Database

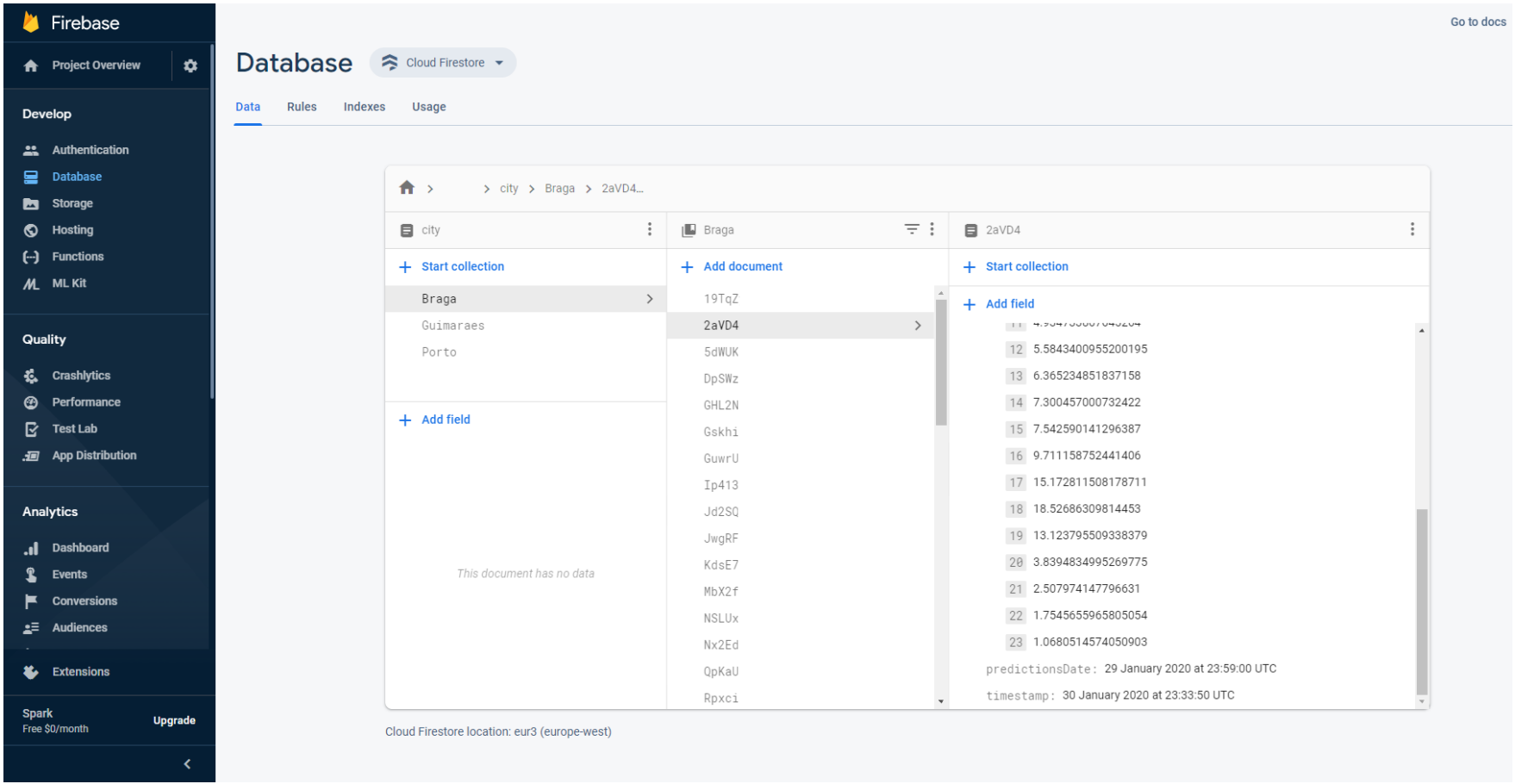
- **Firebase Realtime Database** is a cloud-hosted NoSQL database that let store and sync the sensors data in real-time
  - It is one big JSON object that the developers can manage in real-time

- **Cloud Firestore** (yet another database) takes a more structured approach
  - It is the main Firebase database
  - Faster queries and performance than the real-time database
  - However, it still lacks some important features (such as data exporting)

Firebase Database



Firebase Database



Firebase Database

```
fun getUserUID(): String {
    return FirebaseAuth.getInstance().currentUser?.uid ?: "-99"
}

fun readCitiesTrafficFirebaseForAlerts(collection:String, subCollection: String, document: String, context: Context?){
    if(!isDbStarted)
        startConnection()
    db.collection(collection).document(subCollection).collection(document).orderBy("timestamp", Query.Direction.DESCENDING).limit(1).get()
        .addOnSuccessListener { queryDocumentSnapshots ->
            if (queryDocumentSnapshots != null) {
                val allDocs = queryDocumentSnapshots.documents
                if(allDocs.size > 0){
                    for(doc in allDocs) {
                        doc.toObject(FirebaseTraffic::class.java)?.let {
                            if(context != null)
                                launchTrafficAlertNotification(context, it)
                        }
                    }
                }
            }
        }
}
```

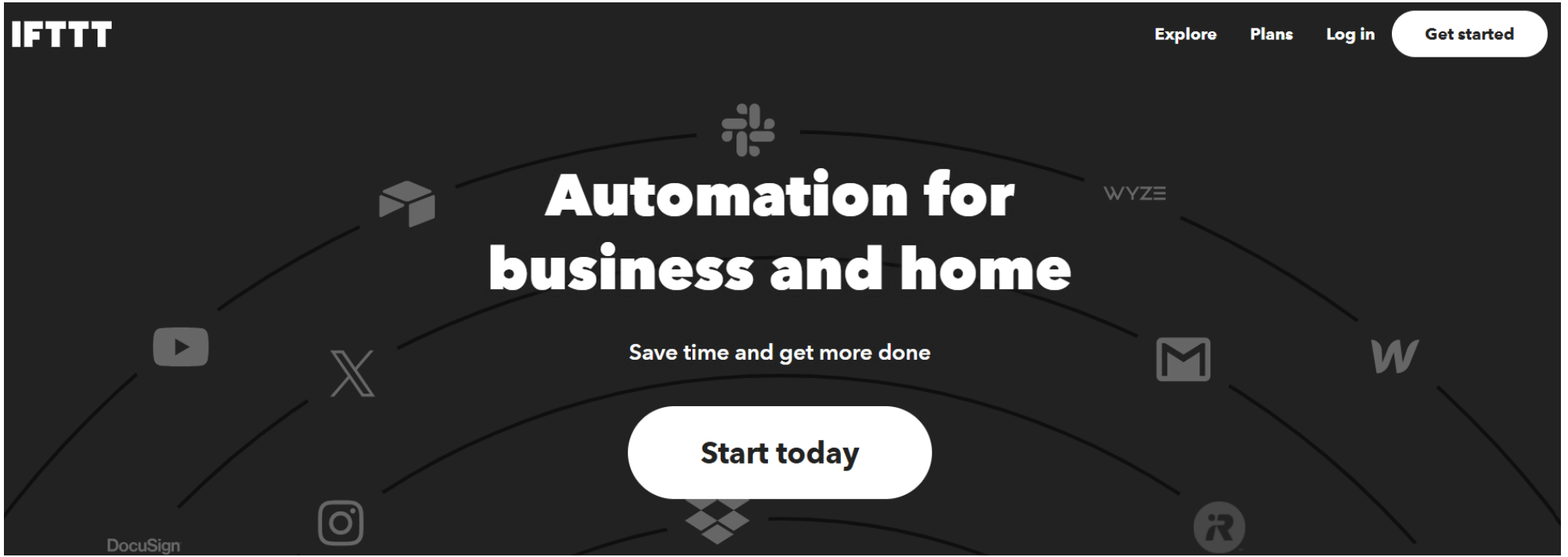
https://firebase.google.com/docs/firestore/quickstart

Firebase Database

```
service cloud.firestore {
  match /databases/{database}/documents {
    // A read rule can be divided into get and list rules
    match /cities/{city} {
      // Applies to single document read requests
      allow get: if <condition>;

      // Applies to queries and collection read requests
      allow list: if <condition>;
    }

    // A write rule can be divided into create, update, and delete rules
    match /cities/{city} {
      // Applies to writes to nonexistent documents
      allow create: if <condition>;

      // Applies to writes to existing documents
      allow update: if <condition>;

      // Applies to delete operations
      allow delete: if <condition>;
    }
  }
}
```

https://firebase.google.com/docs/firestore/security/rules-structure

If This Then That (IFTTT)

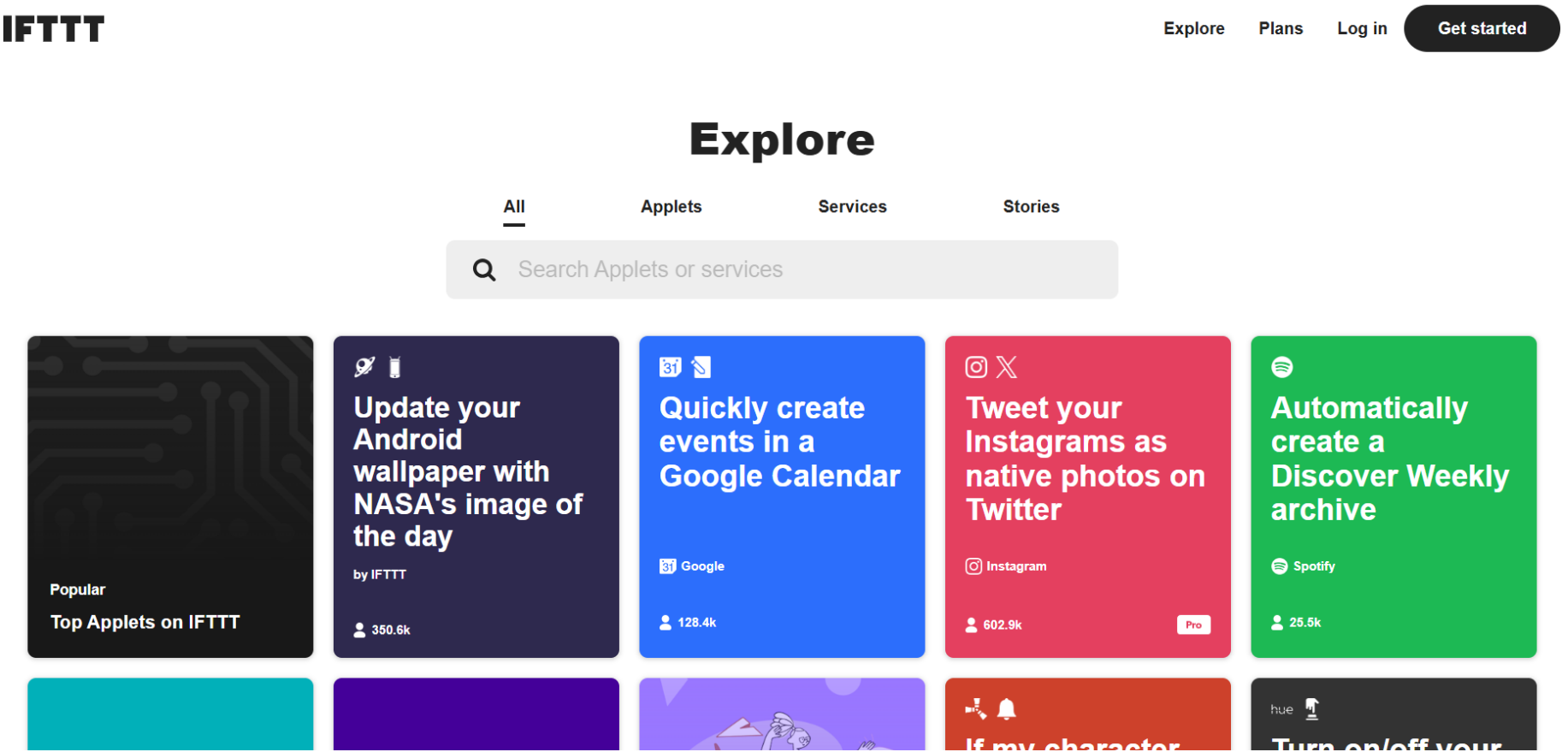


https://ifttt.com/

IFTTT

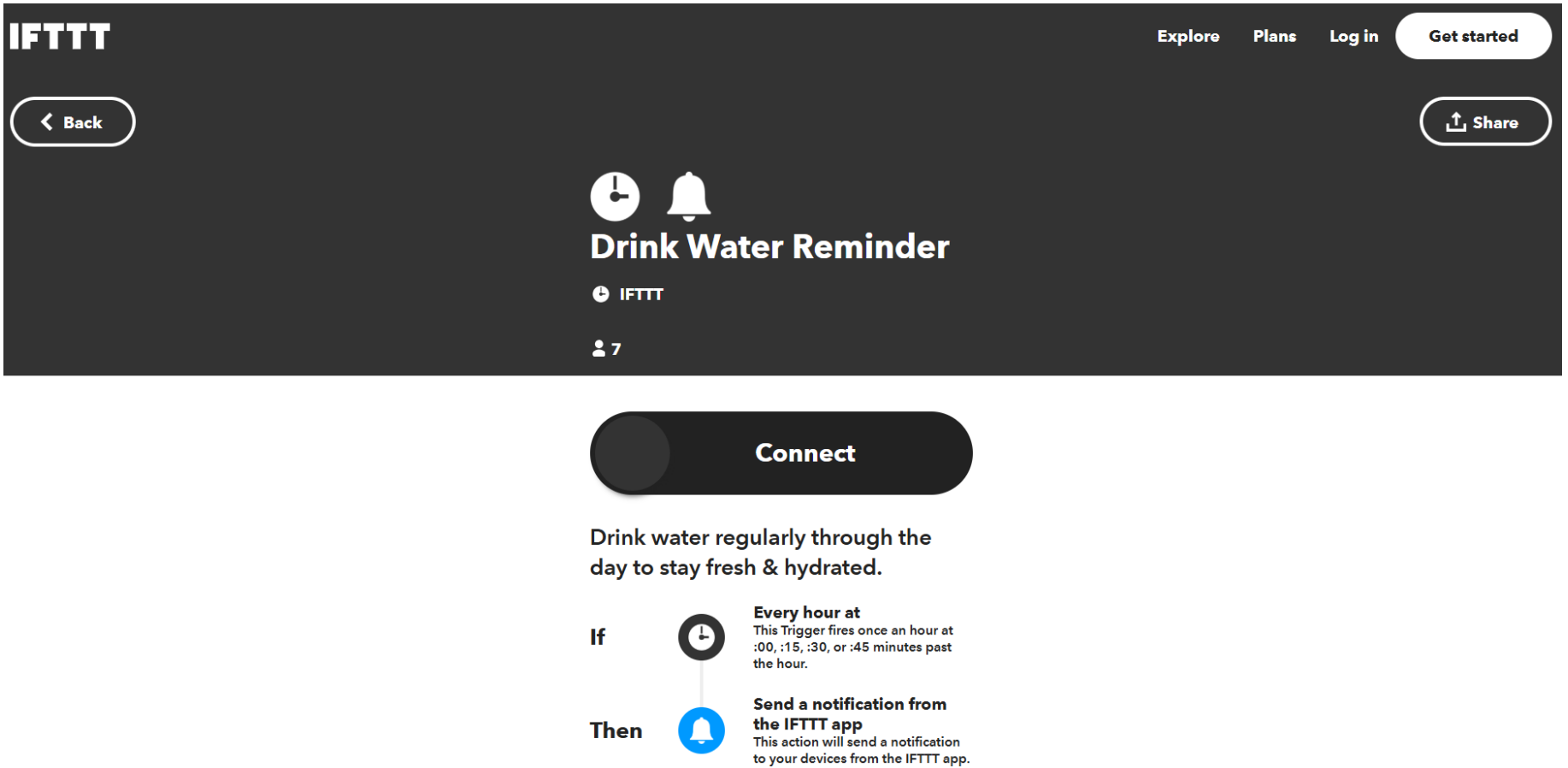
- A service to create chains of **conditional statements**, called **applets**
- It is **triggered** by **changes** that occur **within other web services**
- After triggered, it executes an **actionable service** in the platform
- Besides the **web-based application**, it runs on **iOS** and **Android**
- Alternatives:
  - Zapier
  - Microsoft Power Automate
  - AutomationAnywhere

IFTTT



https://ifttt.com/explore

IFTTT



https://ifttt.com/applets/QwrmUPC4-drink-water-reminder

Message Queue Telemetry Transport (MQTT)

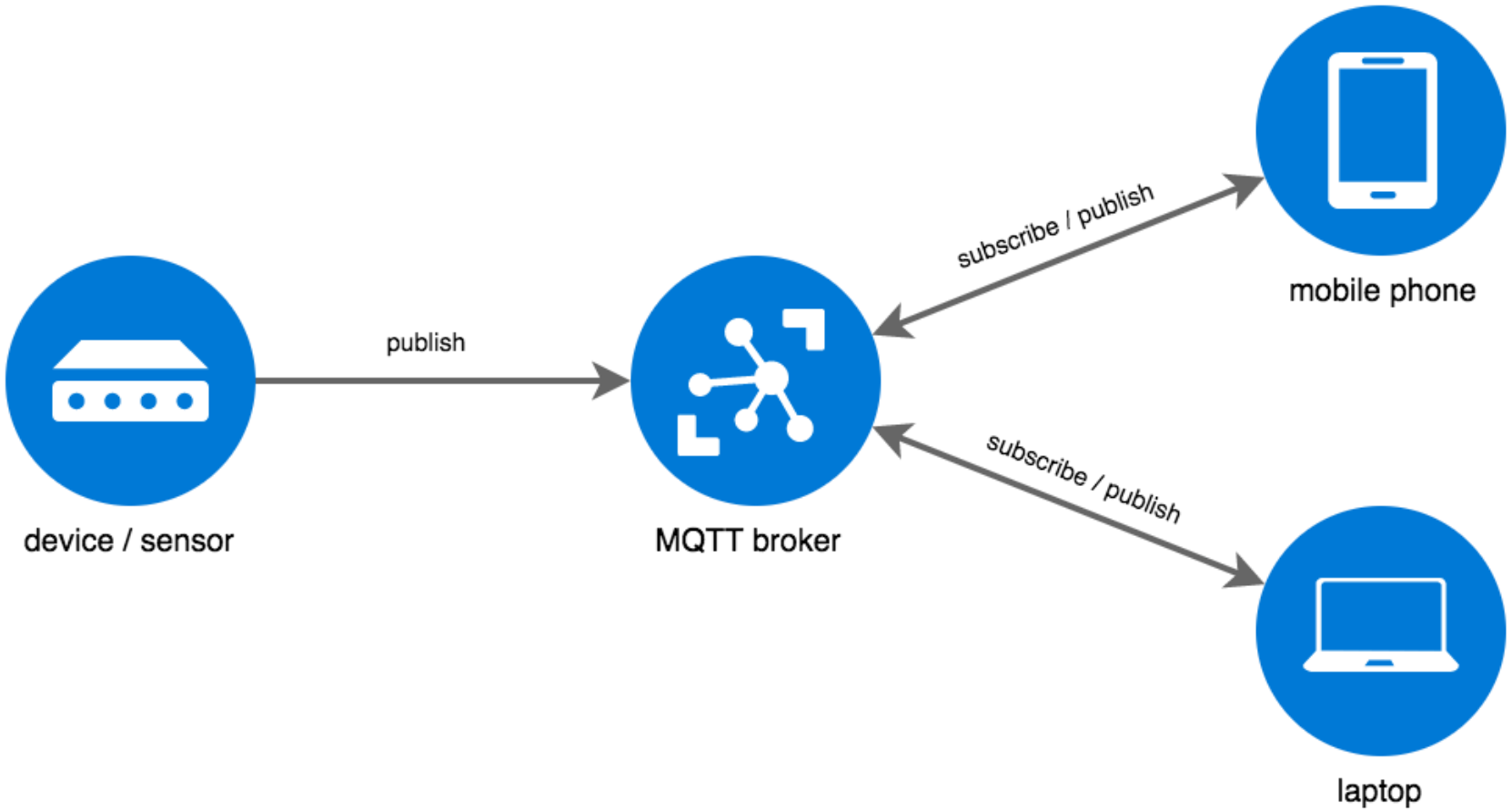
- MQTT is a Machine-to-Machine/IoT connectivity protocol
  - It was originally developed out of IBM's pervasive computing team

- It is a **publish/subscribe** extremely simple and **lightweight messaging protocol**, designed for constrained devices and low-bandwidth, high-latency networks

- **MQTT messages** are **sent to feeds** in an **MQTT broker** (such as Adafruit IO), which then distributes them through the devices that subscribed those feeds

- Lightweight message protocol
  - Connecting to a server only takes about 80 bytes
  - Push data from server to device is about 20 bytes

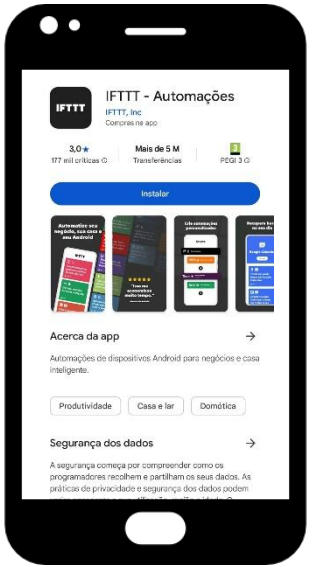
MQTT


publish
subscribe / publish
subscribe / publish
device / sensor
MQTT broker
mobile phone
laptop

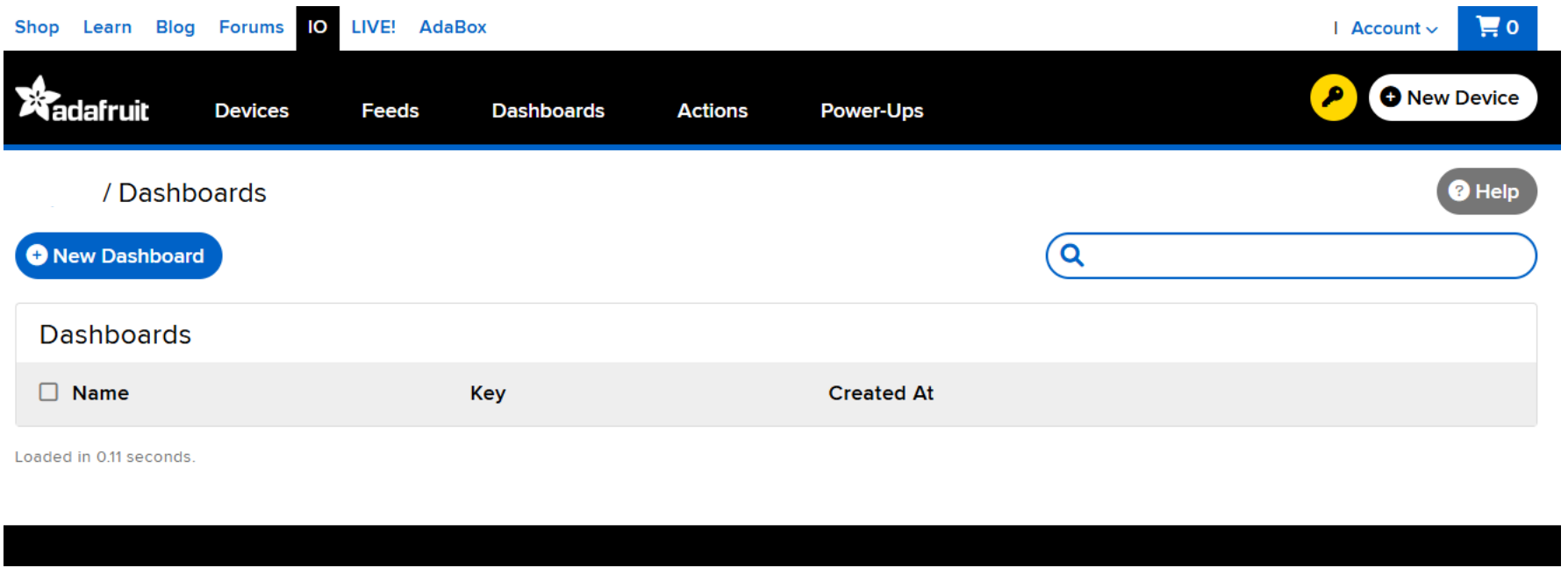
Adafruit IO + IFTTT

Adafruit IO + IFTTT

- How to:
  - Develop an **IFTTT applet** that **reacts to values in Adafruit IO feeds**
  - The goal is to **monitor sensor values** sent to Adafruit IO feeds and **take actions** in specific context
- To complete this you need:
  - An **IFTTT account**
  - An **Adafruit IO account**
  - A smartphone with the **IFTTT application installed**



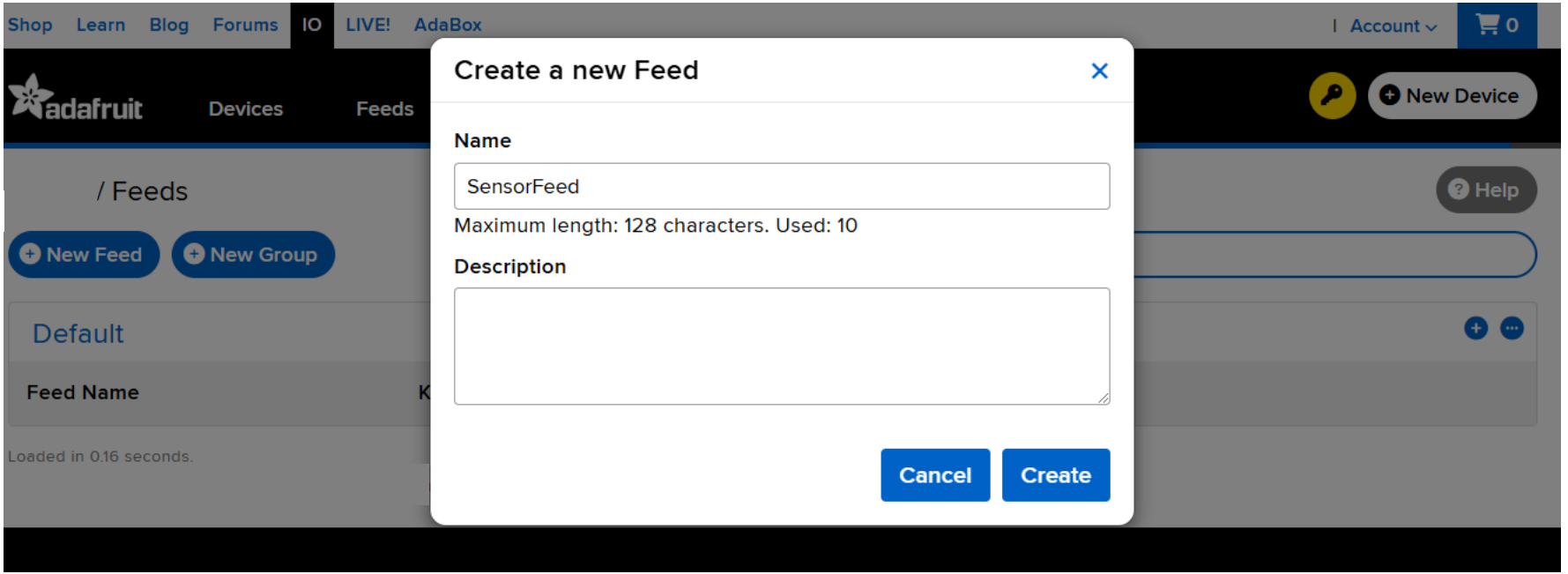
Adafruit IO + IFTTT



https://io.adafruit.com/.../dashboards

Adafruit IO + IFTTT

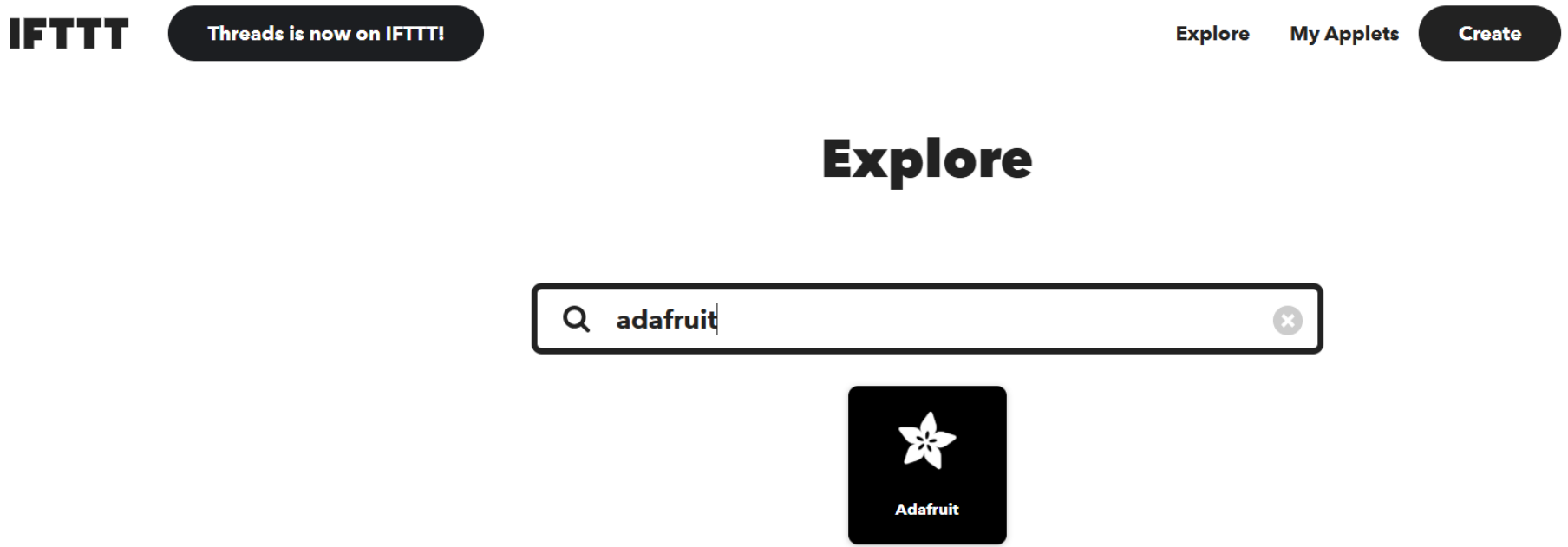
- Create a new feed named **SensorFeed** in Adafruit IO



https://io.adafruit.com/.../feeds

Adafruit IO + IFTTT

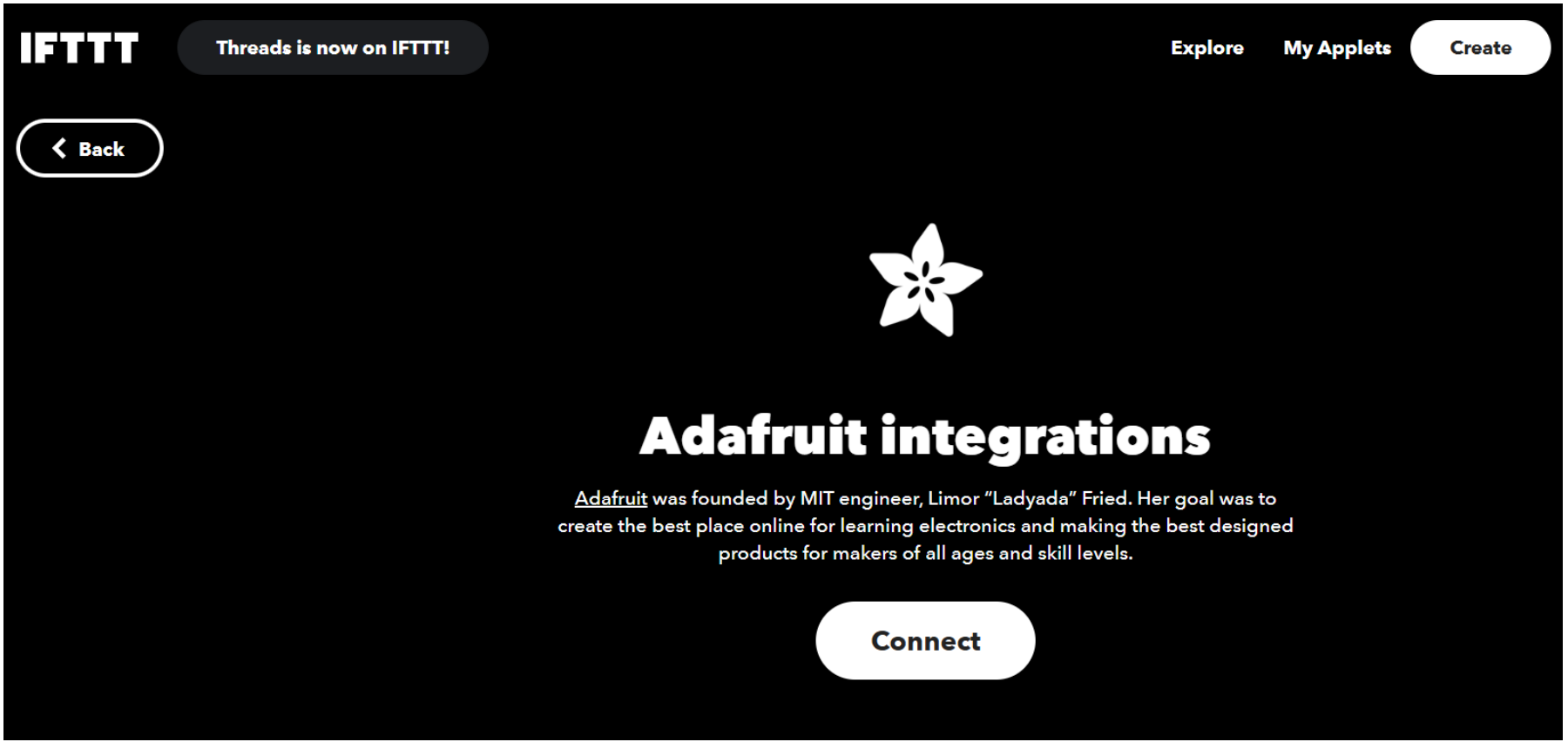
- Login into IFTTT and **search for the Adafruit** service



https://ifttt.com/search/query/adafruit

Adafruit IO + IFTTT

- **Connect** IFTTT to the Adafruit platform



Adafruit IO + IFTTT

- **Authorize** IFTTT access

Shop Learn Blog Forums IO LIVE! AdaBox | Account 0

adafruit My Adafruit My Wishlists Account Settings

Account Settings

Account

Profile
Security & Privacy
Subscriptions
Services
Change Password
Accessibility

Shop

Saved Addresses
Payment Methods

Authorize IFTTT

The application IFTTT is requesting the following information for your account. Would you like to grant access?

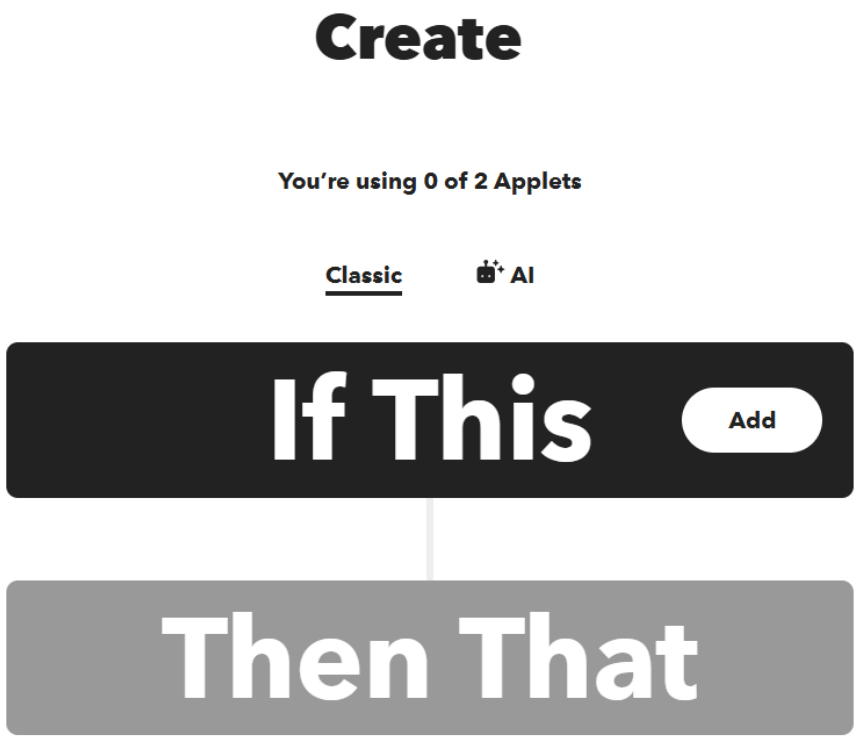
Name
username
Adafruit IO Dashboard URL
Read and write to your feed data

By granting access, you'll be able to connect your Adafruit account to IFTTT, and enable any integrations that have been provided.

CANCEL AUTHORIZE

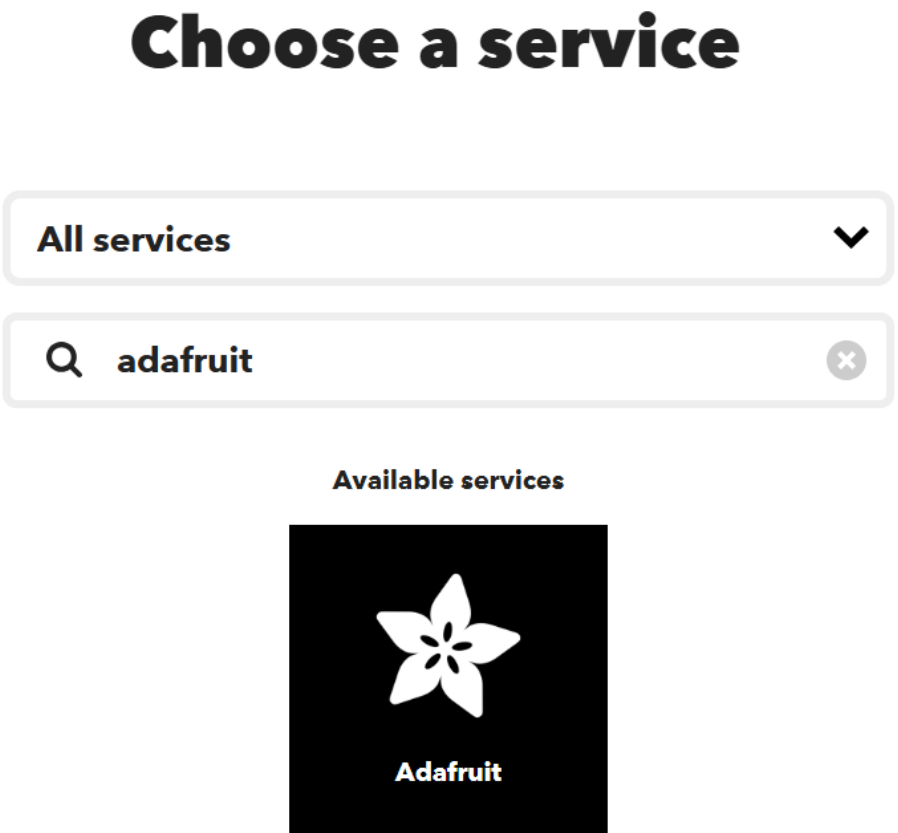
Adafruit IO + IFTTT

- On the IFTTT platform lets **create a new applet (If This)**



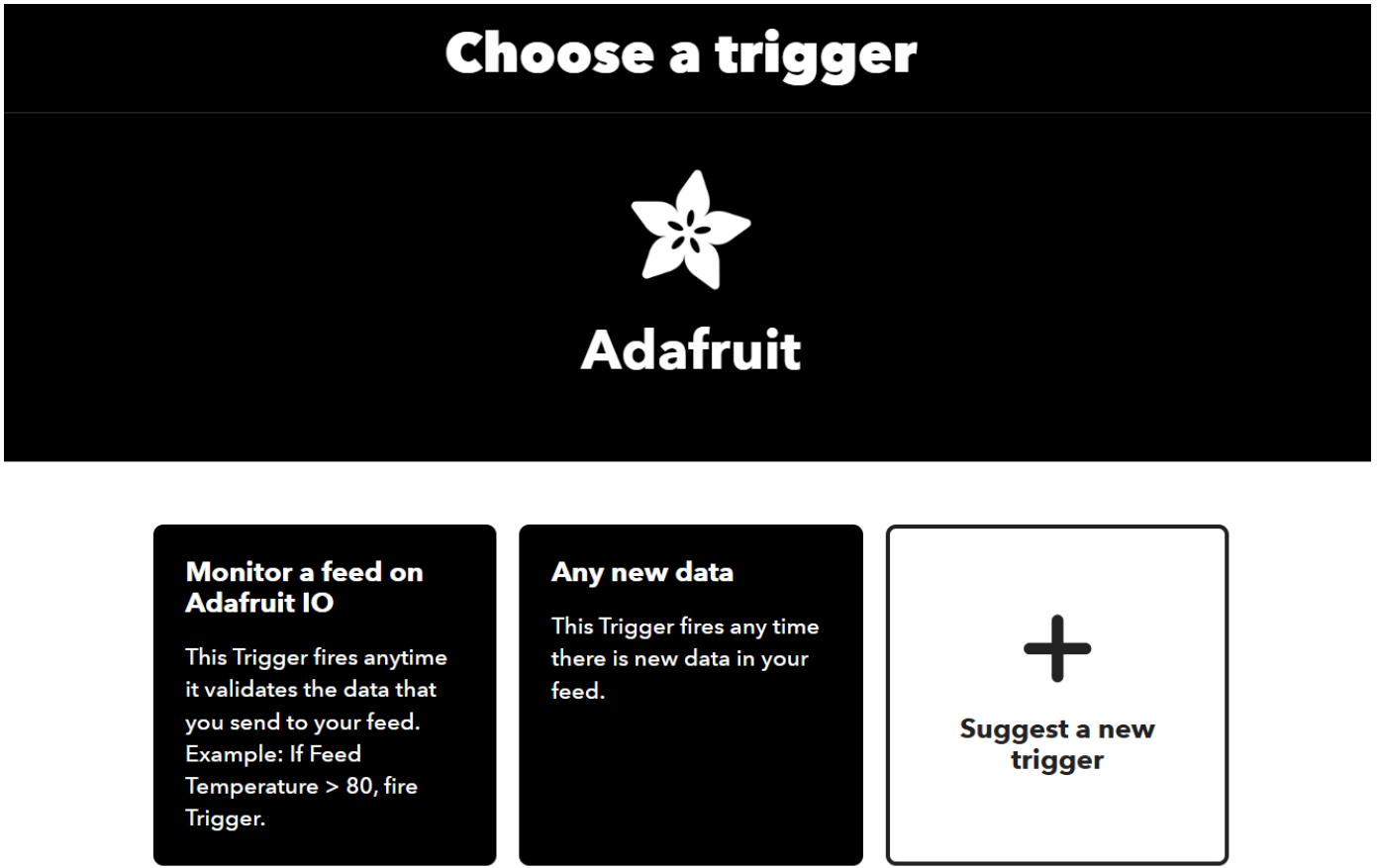
Adafruit IO + IFTTT

- Search for the **Adafruit** service



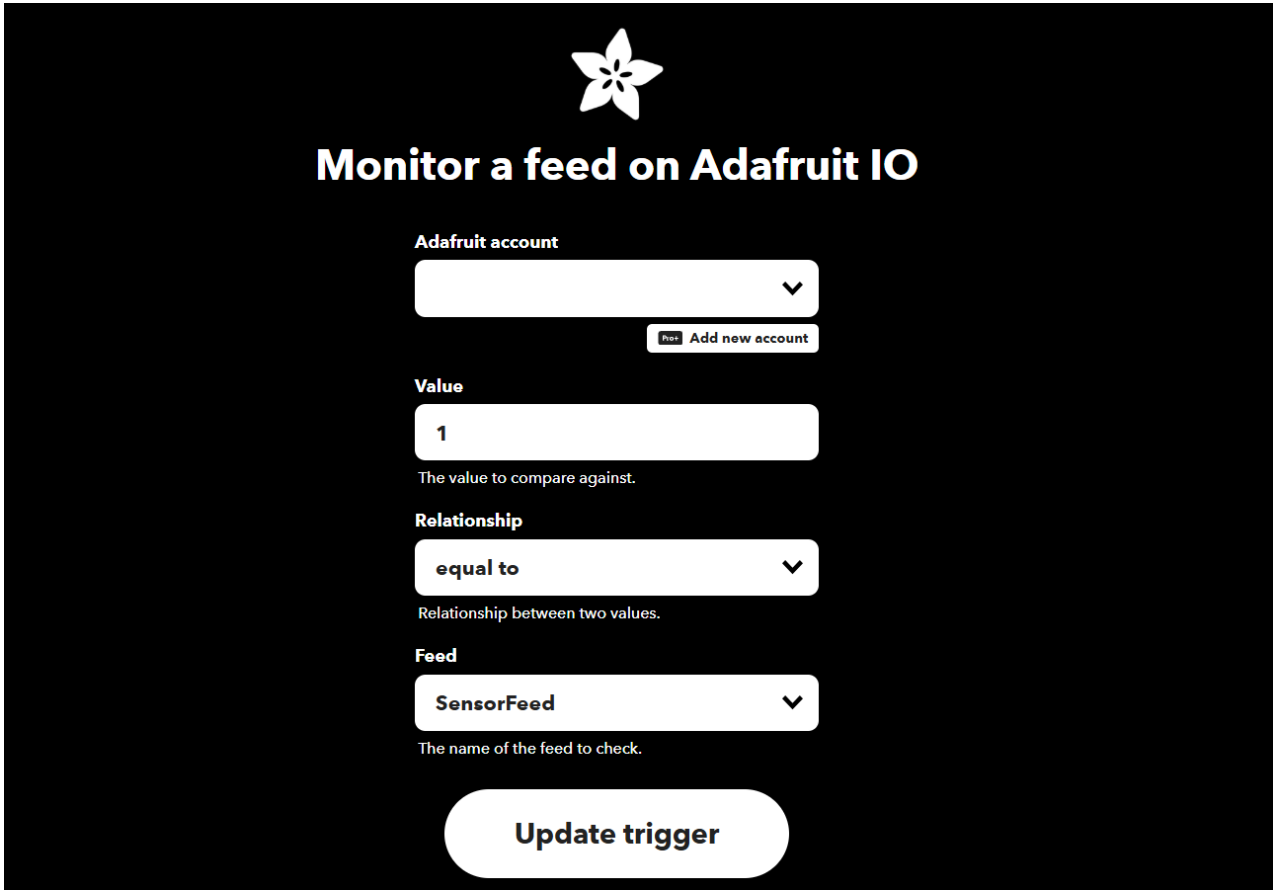
Adafruit IO + IFTTT

- Let's **monitor a feed** and fire a trigger if some condition is met



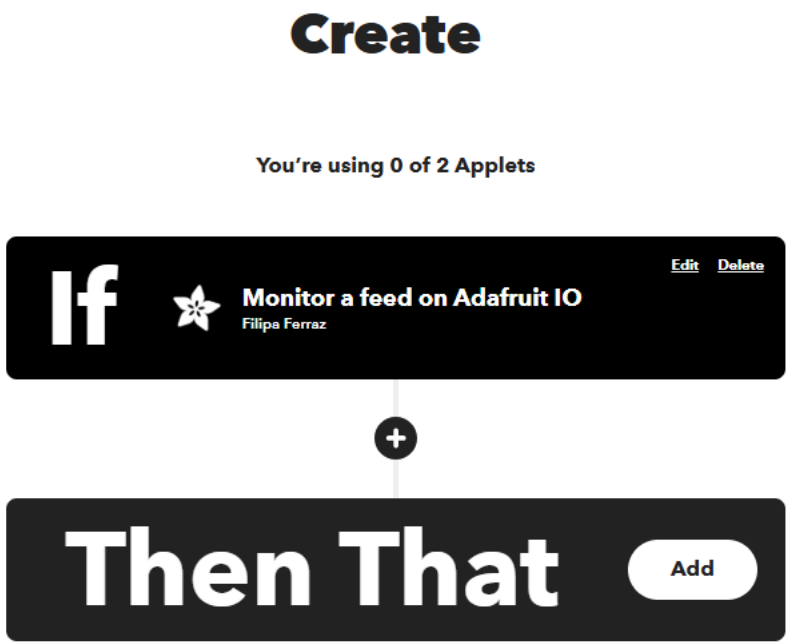
Adafruit IO + IFTTT

- Let's monitor **SensorFeed** and fire a **trigger** if the value of **1** is **sent to the feed**



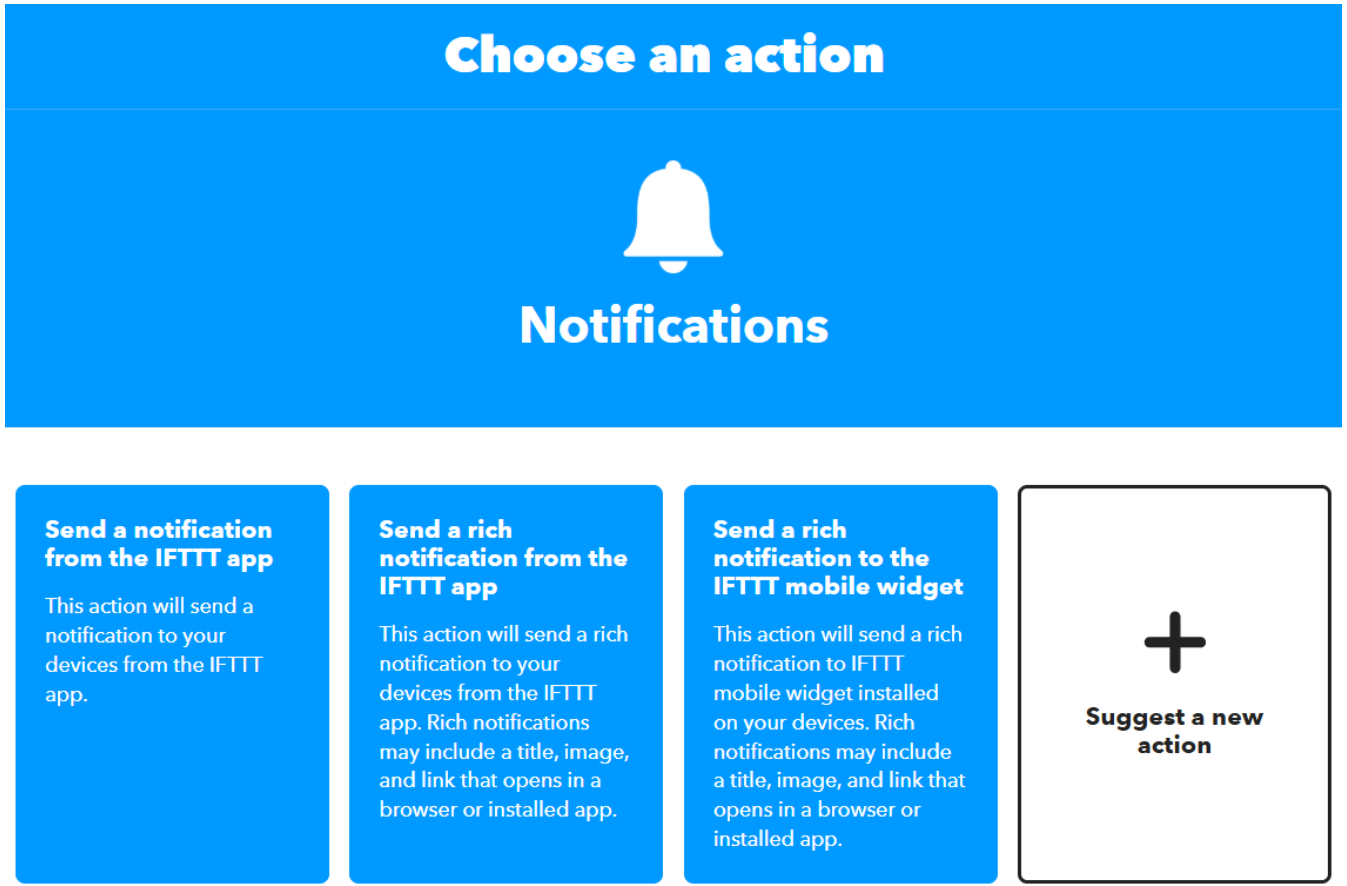
Adafruit IO + IFTTT

- If this > **Then That**



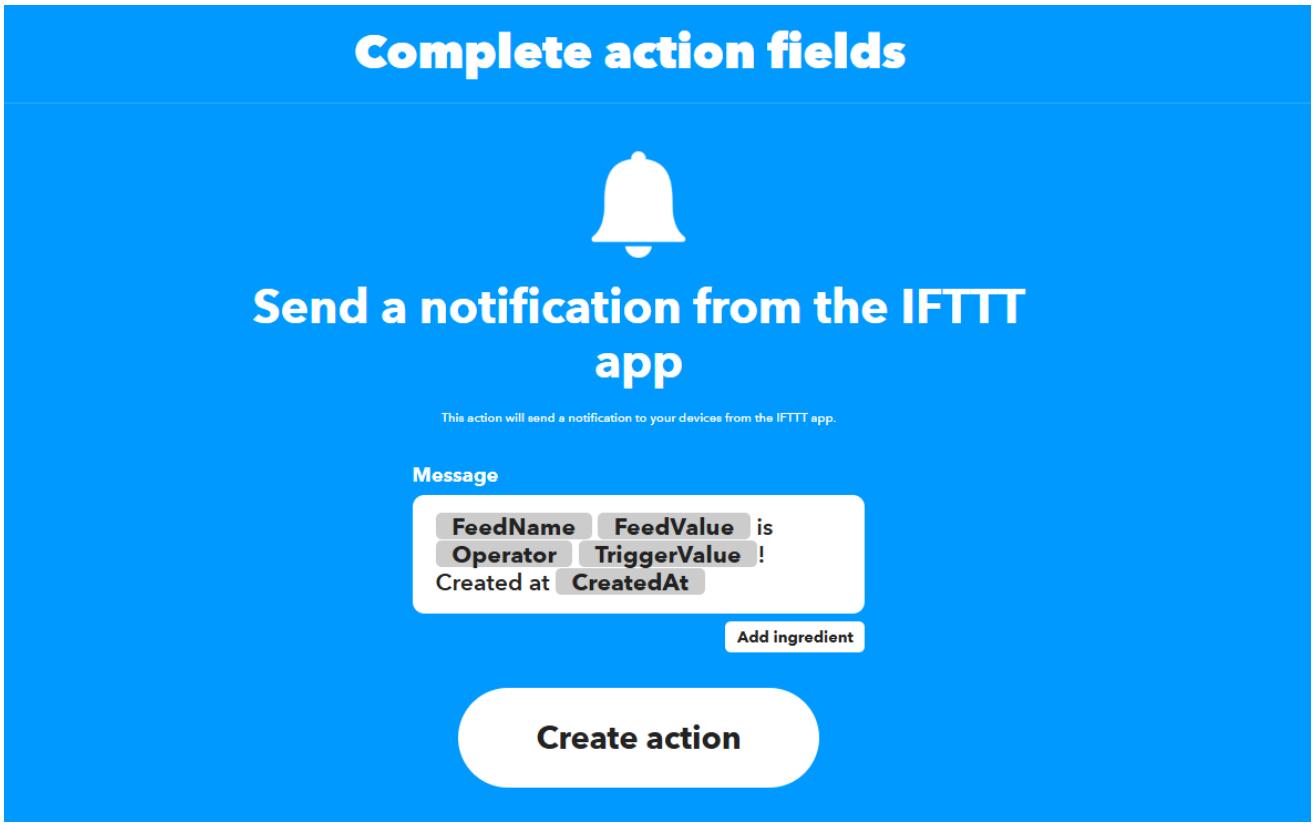
Adafruit IO + IFTTT

- Search for the **Notifications** service and select the option to **send a notification** from the **IFTTT app**



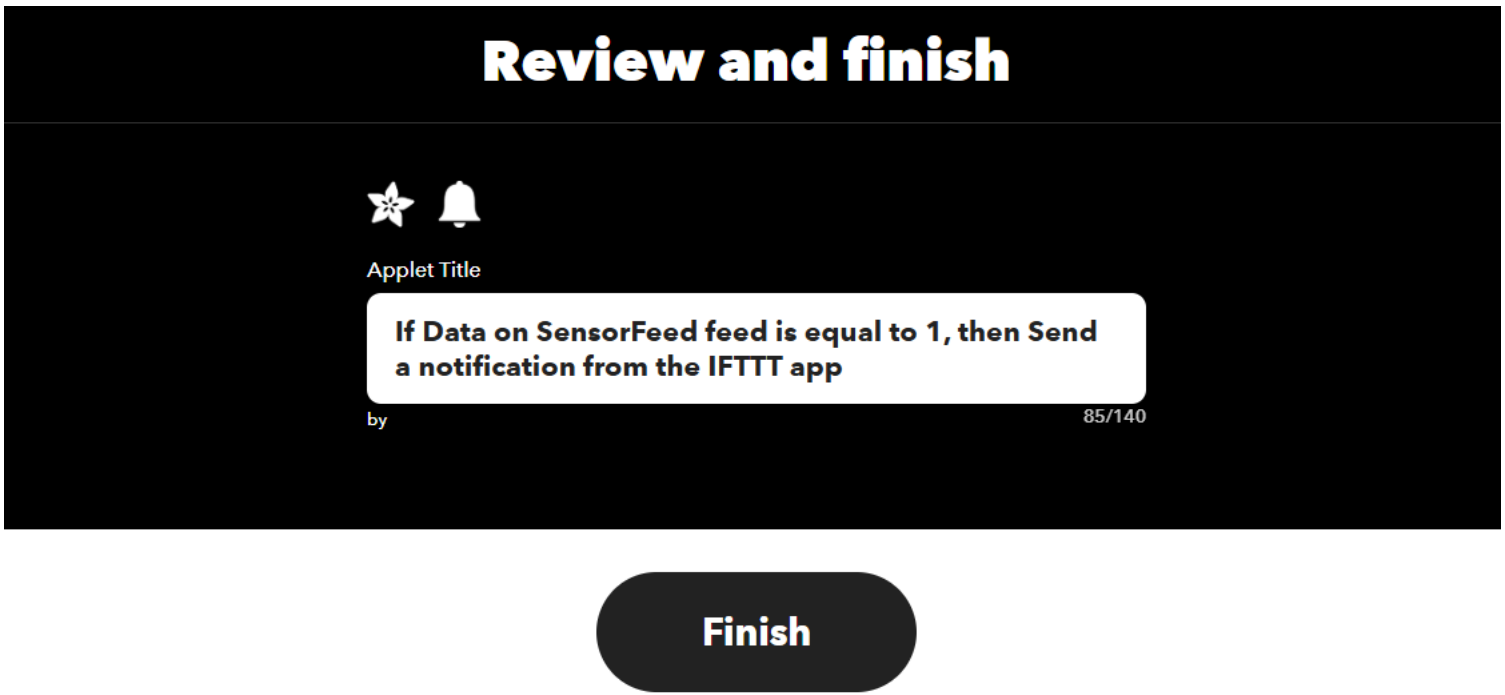
Adafruit IO + IFTTT

- Connect and create a **custom notification** to display in the device **where the IFTTT app is installed**. You can **obtain values** from the feed by **adding ingredients**



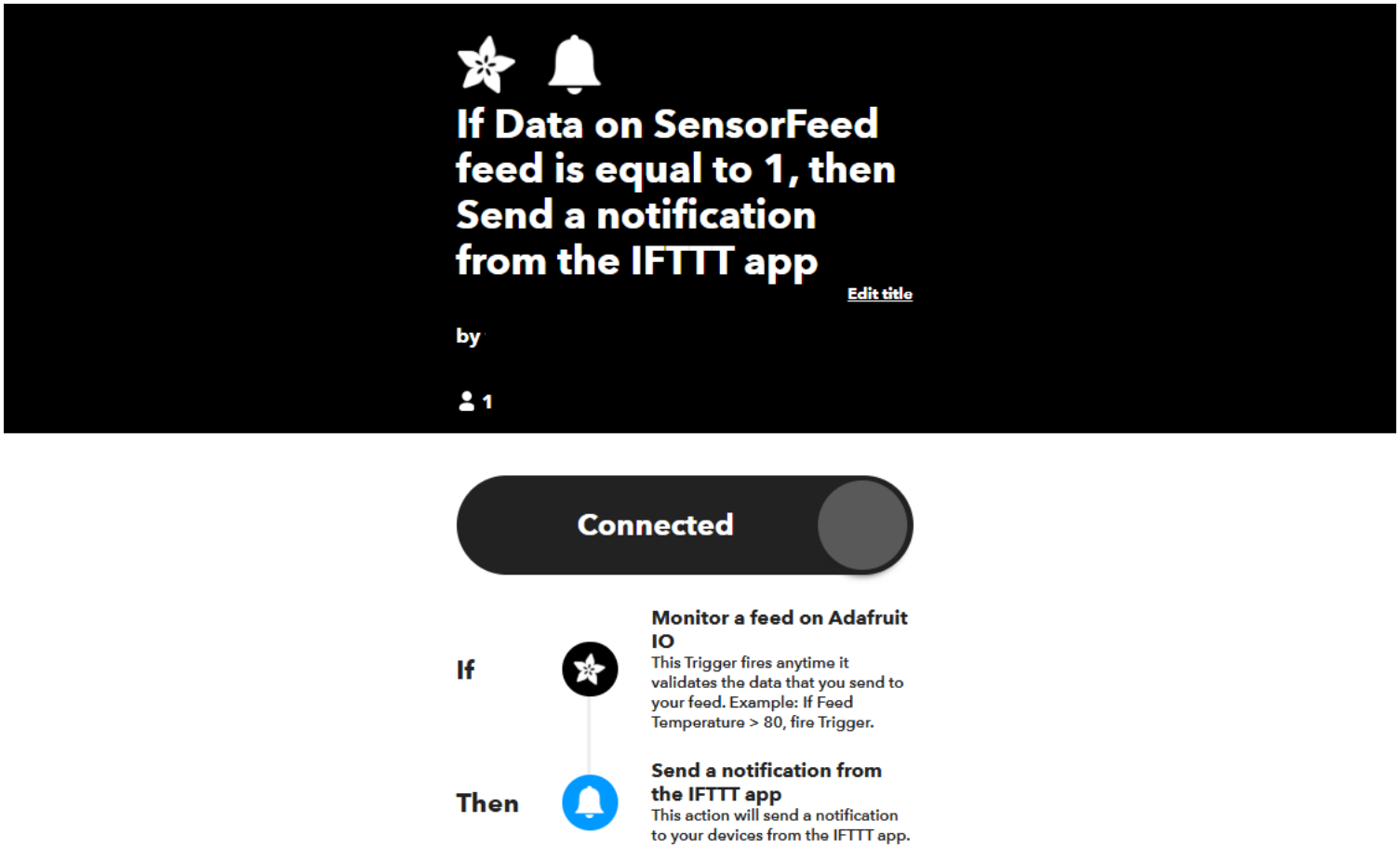
Adafruit IO + IFTTT

- **Review** the created rule and **confirm** it



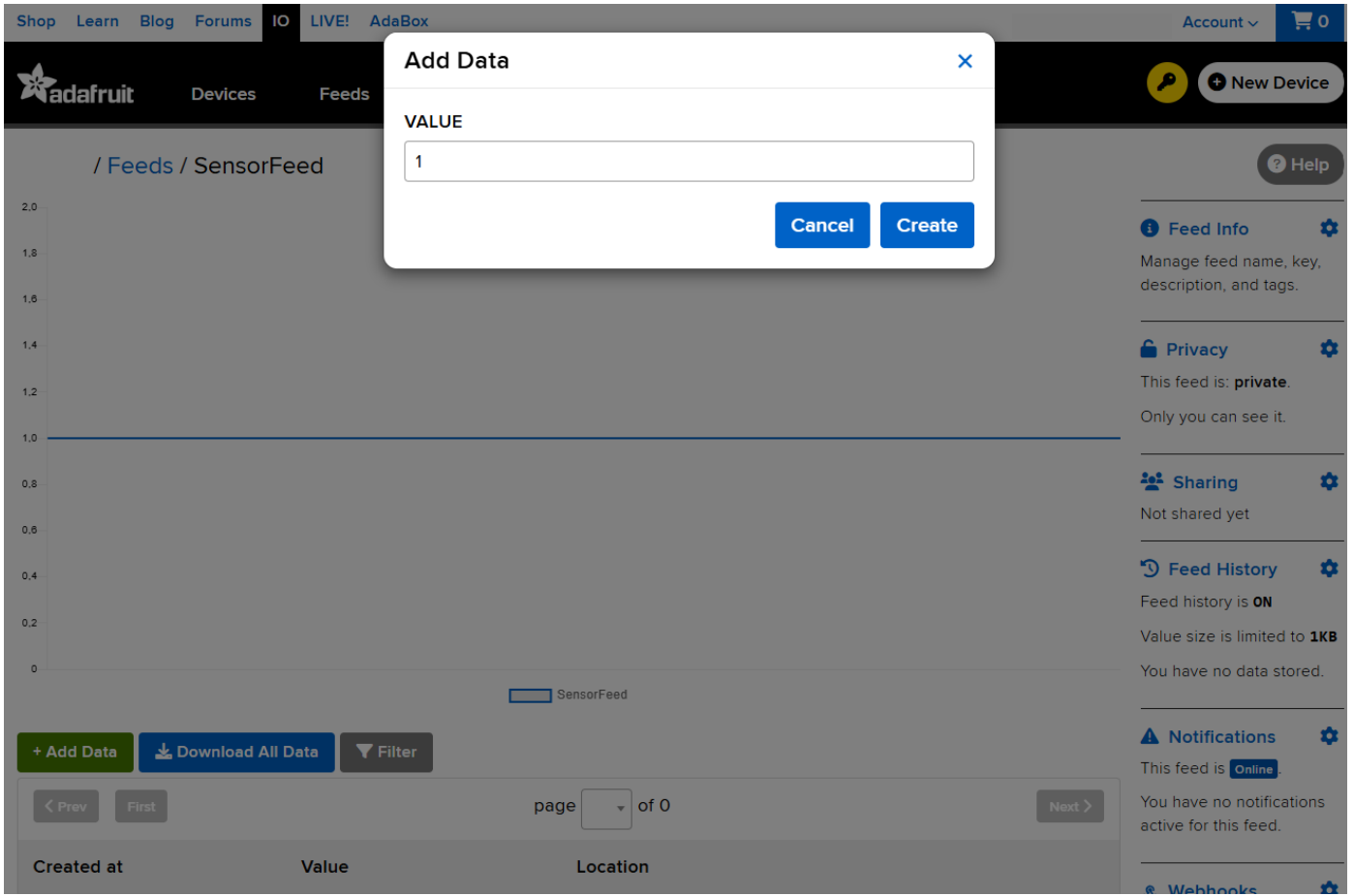
Adafruit IO + IFTTT

- **Activate** it



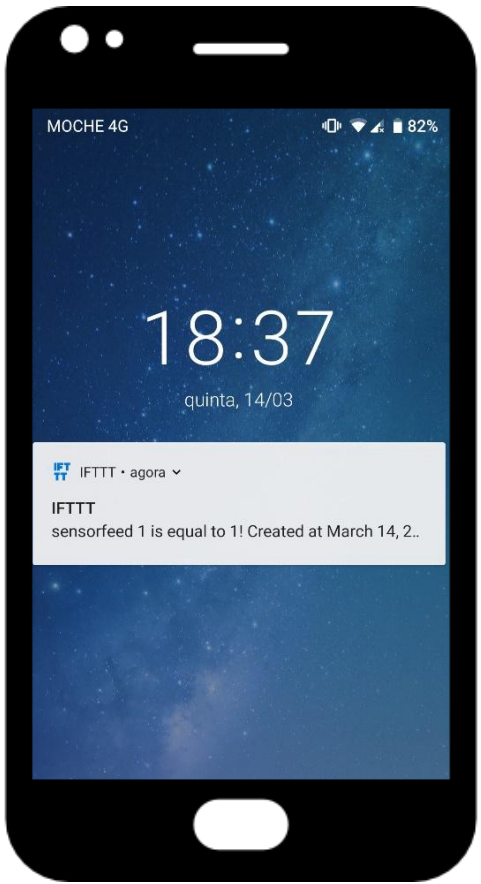
Adafruit IO + IFTTT

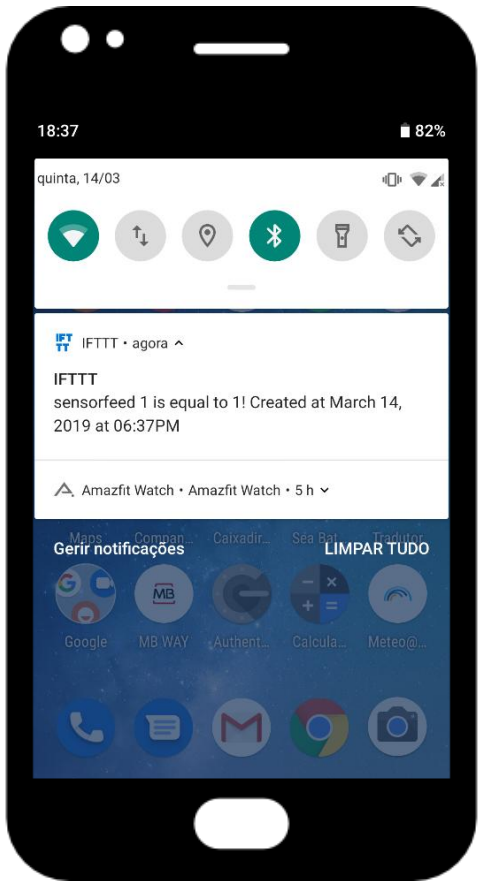
- Go to Adafruit IO and **add a new data** point with **value of 1** to the **SensorFeed**



Adafruit IO + IFTTT

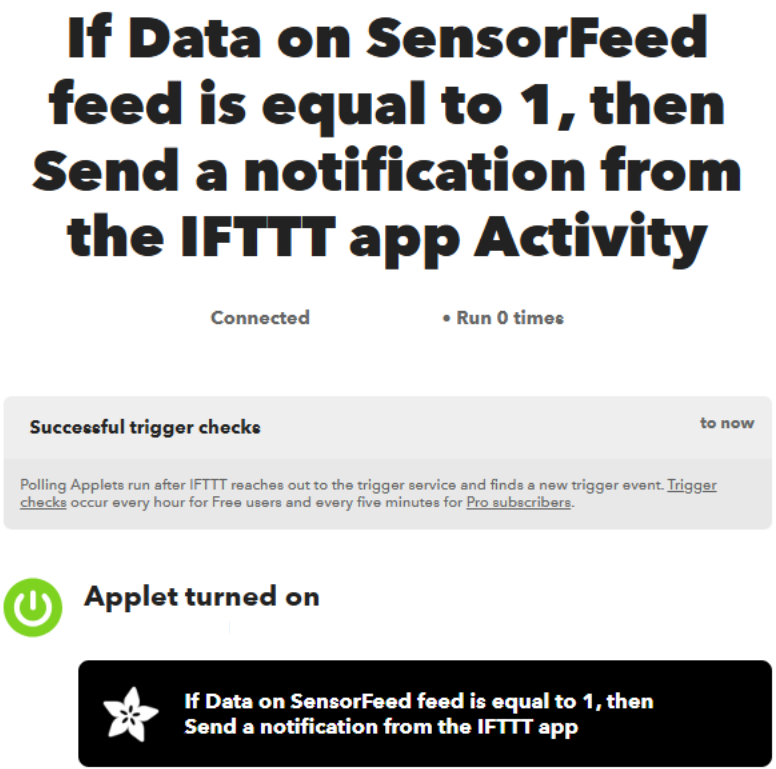
- **Notification received** in the smartphone





Adafruit IO + IFTTT

- If **not**, you may need to **enable push notifications**



Adafruit IO + Java

Adafruit IO + Java

- How to:
  - Implement a simple **Java program** to test interaction between **Adafruit IO** and **Java**
    - It will be used to publish a value to the previously created feed named **SensorFeed**
  - **Eclipse Paho project** provides an open-source client **implementation of the MQTT protocol** aimed at emerging applications for the **Internet of Things**
- Useful links:
  - https://projects.eclipse.org/projects/iot.paho
  - http://wiki.eclipse.org/Paho

Adafruit IO + Java

```
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;

public class MQTT_Test{

    private final static String ADAFRUIT_USERNAME = "YOUR_AIO_USERNAME";
    private final static String ADAFRUIT_AIO_KEY  = "YOUR_AIO_KEY";

    public static void main(String[] args) {

        String topic                          = ADAFRUIT_USERNAME + "/feeds/sensorfeed";
        String msg_content          = "Hello from java (not the island)!";
        int qos                               = 1; //QoS: 0 - at most once, 1 - at least once, 2 - exactly once
        String broker               = "tcp://io.adafruit.com:1883"; //Adafruit IO broker
        String client_id            = "JavaSample";
        MemoryPersistence persistence = new MemoryPersistence();
        …
```

Adafruit IO + Java

```java
try {
    MqttClient mqtt_client = new MqttClient(broker, client_id, persistence);

    MqttConnectOptions connOpts = new MqttConnectOptions();
    connOpts.setCleanSession(true);
    connOpts.setUserName(ADAFRUIT_USERNAME);
    connOpts.setPassword(ADAFRUIT_AIO_KEY.toCharArray());

    System.out.println("Connecting to broker: " + broker);
    mqtt_client.connect(connOpts);
    System.out.println("Connected. Publishing message: " + msg_content);

    MqttMessage message = new MqttMessage(msg_content.getBytes());
    message.setQos(qos);
    mqtt_client.publish(topic, message);
    System.out.println("Message published");
    mqtt_client.disconnect();
    System.out.println("Disconnected");
    System.exit(0);
}
```

Adafruit IO + Java

```
        …
        catch(MqttException me) {
            System.out.println("reason: " + me.getReasonCode());
            System.out.println("msg: " + me.getMessage());
            System.out.println("loc: " + me.getLocalizedMessage());
            System.out.println("cause: " + me.getCause());
            System.out.println("excep: " + me);
            me.printStackTrace();
        }
    }

}
```

Adafruit IO + JavaScript

Adafruit IO + JavaScript

- How to:
  - Implement a simple **web page** to test **interaction** between **Adafruit IO** and **JavaScript**
    - It will be used to **publish** a value to the previously created **SensorFeed**
    - It will be used to **subscribe** to the feed and update the page on a received message
  - **Eclipse Paho project** also provides an **open-source JS client** implementing the MQTT protocol
- Useful links:
  - https://projects.eclipse.org/projects/iot.paho
  - http://wiki.eclipse.org/Paho
  - https://io.adafruit.com/api/docs/

Adafruit IO + JavaScript

```
<html>
    <head>
        <title>Adafruit IO + JS</title>
    </head>
    <body>
        <h1> Test it! </h1>
        <button onclick="publish()">Publish</button>

        <script src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.js"
                  type="text/javascript"></script>
        <script>
            //create a client instance
            client = new Paho.MQTT.Client("io.adafruit.com", Number(443), "JS_Client");

            //set callback handlers
            client.onConnectionLost = onConnectionLost;
            client.onMessageArrived = onMessageArrived;
            //connect the client
            client.connect({onSuccess:onConnect, userName:"YOUR_AIO_USERNAME",
                    password:"YOUR_AIO_KEY", useSSL:true, mqttVersion:4});
```

Adafruit IO + JavaScript

```
//called when the client connects
function onConnect() {
    console.log("onConnect");
    //subscribe
    client.subscribe("YOUR_AIO_USERNAME/feeds/sensorfeed");
}

function publish(){
    //send message
    message = new Paho.MQTT.Message("Hello from JS!");
    message.destinationName = "YOUR_AIO_USERNAME/feeds/sensorfeed";
    client.send(message);
}

//called when the client loses its connection
function onConnectionLost(responseObject) {
    if (responseObject.errorCode !== 0) {
        console.log("onConnectionLost:" + responseObject.errorMessage);
    }
}
```

Adafruit IO + JavaScript

```
            //called when a message arrives
            function onMessageArrived(message) {
                console.log("onMessageArrived:" + message.payloadString);
                var h1 = document.createElement("h1");
                h1.appendChild(document.createTextNode(message.payloadString));
                document.body.appendChild(h1);
            }
        </script>

    </body>
</html>
```

Adafruit IO + JavaScript

Test it!

Publish

Elements Console Sources Netw

top Filter

onConnect

Console

Test it!

Publish

Hello from JS!

Elements Console Sources Network Performance Memory Application Security Audits AdBlock EditThisCookie

top Filter Default levels

onConnect hello_adafruit.html:24

onMessageArrived:Hello from JS! hello_adafruit.html:45

Console

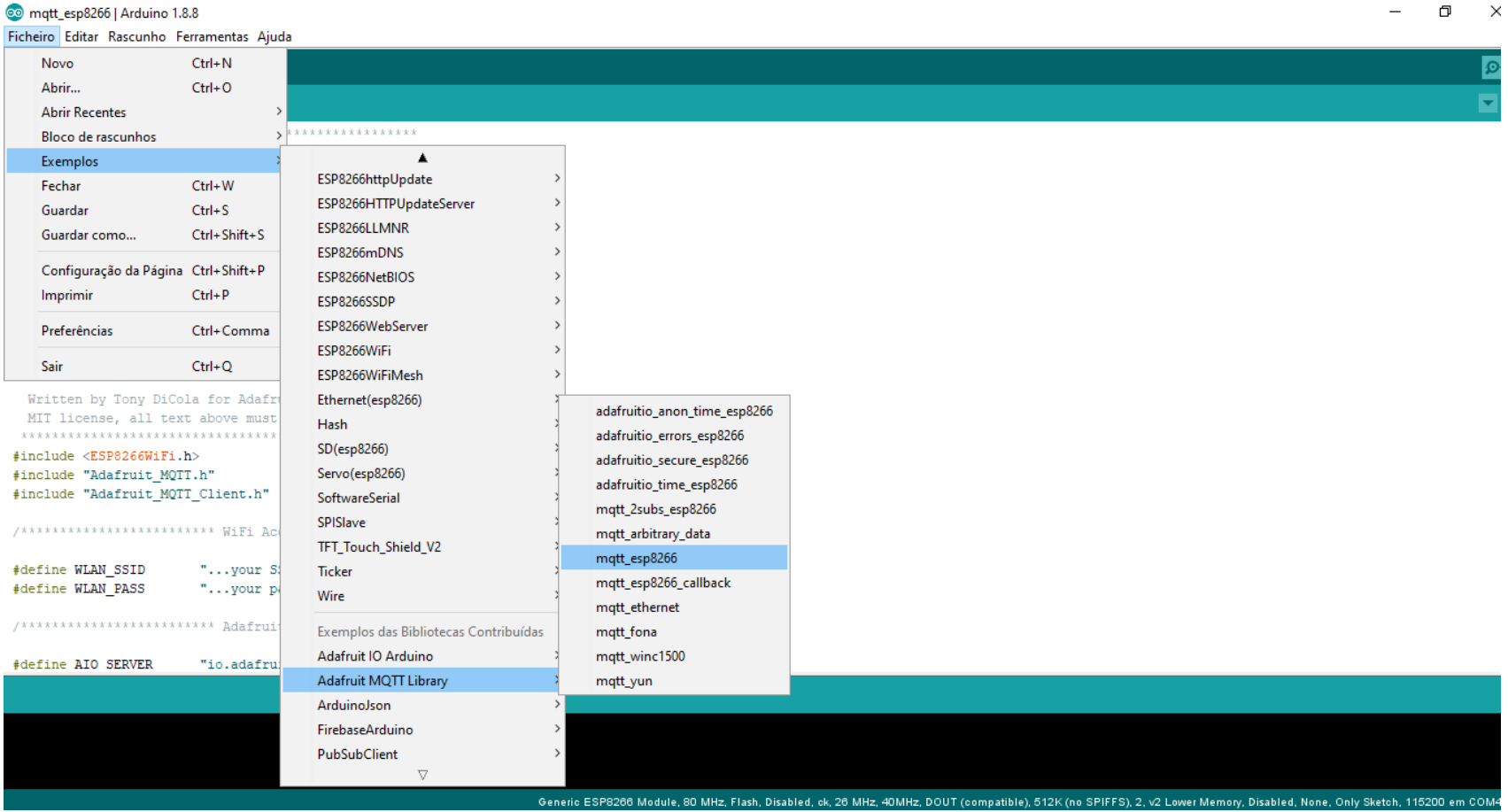
Adafruit IO + Arduino

Adafruit IO + Arduino

- How to:
  - Implement a **simple sketch** to test **interaction** between **Adafruit IO** and **Arduino** boards
    - It will be used to **publish** a value to the previously created SensorFeed
    - It will be used to **subscribe** to the feed
  - Let's use the **Adafruit MQTT** library
    - There are others such as the **PubSubClient MQTT** Library
- Useful links:
  - https://learn.adafruit.com/welcome-to-adafruit-io/arduino-and-adafruit-io
  - https://learn.adafruit.com/mqtt-adafruit-io-and-you/intro-to-adafruit-mqtt
  - https://github.com/adafruit/Adafruit_MQTT_Library

Adafruit IO + Arduino

- Let's use **mqtt_esp8266** example from the **Adafruit MQTT Library** to get started



Adafruit IO + Arduino

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h“

/************************* WiFi Access Point *********************************/
#define WLAN_SSID                   "SSID"
#define WLAN_PASS       "password"

/************************* Adafruit.io Setup *********************************/
#define AIO_SERVER                  "io.adafruit.com"
#define AIO_SERVERPORT  1883                    //use 8883 for SSL
#define AIO_USERNAME    "YOUR_AIO_USERNAME"
#define AIO_KEY         "YOUR_AIO_KEY"

/************ Global State (you don't need to change this!) ******************/
//create an ESP8266 WiFiClient class to connect to the MQTT server
WiFiClient client;
//or use WiFiFlientSecure for SSL -> WiFiClientSecure client

//setup the MQTT client class by passing in the WiFi client, MQTT server and login details
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
```

Adafruit IO + Arduino

```
/****************************** Feeds ***************************************/
Adafruit_MQTT_Publish sensorfeed_publish = Adafruit_MQTT_Publish(&mqtt,
                                                                 AIO_USERNAME
"/feeds/sensorfeed");
Adafruit_MQTT_Subscribe sensorfeed_subscribe = Adafruit_MQTT_Subscribe(&mqtt,
                                                                 AIO_USERNAME
"/feeds/sensorfeed");
/*************************** Sketch Code ************************************/
void MQTT_connect();

void setup() {
    Serial.begin(115200); //set the data rate in bits per second (baud) for serial data transmission
    Serial.println(F("*** Adafruit MQTT demo for SensorFeed ***")); //write to serial

    Serial.print("Connecting to "); Serial.println(WLAN_SSID);
    WiFi.begin(WLAN_SSID, WLAN_PASS); //connect to WiFi access point
    while (WiFi.status() != WL_CONNECTED) {
        delay(500); Serial.print(".");
    }
    Serial.println("WiFi connected"); Serial.print("IP address: "); Serial.println(WiFi.localIP());

    mqtt.subscribe(&sensorfeed_subscribe); //setup MQTT subscription for SensorFeed feed
}
```

Adafruit IO + Arduino

```
uint32_t x=0;

void loop() {

    MQTT_connect(); //ensure the connection to the MQTT server is alive. See MQTT_connect()
definition

    Adafruit_MQTT_Subscribe *subscription;
    while ((subscription = mqtt.readSubscription(5000))) {  //wait for incoming subscrip packets
subloop
        if (subscription == &sensorfeed_subscribe) {
            Serial.print(F("Got: ")); Serial.println((char *)sensorfeed_subscribe.lastread);
        }
    }

    Serial.print(F("\nSending sensor x val ")); Serial.print(x); Serial.println("...");
    if (! sensorfeed_publish.publish(x++)) {  //lets publish stuff to the SensorFeed
        Serial.println(F("Failed"));
    } else {
        Serial.println(F("OK!"));
    }

}
```

Adafruit IO + Arduino

```
//function to connect and reconnect as necessary to the MQTT server
void MQTT_connect() {
    if (mqtt.connected()) {  //return if already connected
        return;
    }

    Serial.println("Connecting to MQTT... ");

    int8_t ret;
    uint8_t retries = 3;
    while ((ret = mqtt.connect()) != 0) { //connect will return 0 for connected
        Serial.println(mqtt.connectErrorString(ret)); Serial.println("Retrying MQTT connection in 5
seconds");
        mqtt.disconnect();
        delay(5000);  //wait 5 seconds to retry
        retries--;
        if (retries == 0) {
            while (1);  //basically die and wait for the Watchdog Timer to reset us
        }
    }
    Serial.println("MQTT Connected!!!");
}
```

Universidade do Minho
Departamento de Informática

Hands On

Hands On

- **For even student numbers**
  - Implement an **IFTTT applet** that **reacts** to values in **Adafruit IO feeds**

- **For odd student numbers**
  - Implement a **Java (or Web) program** to **subscribe** and **publish** to **Adafruit IO feeds**

Hands On

Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\data\PythonWorkspace\dev

Editor - C:\data\PythonWorkspace\dev\meanshift_algorithm.py

deep-network.py | meanshift_algorithm.py

```
class Mean_Shift:
    def __init__(self, radius=None, radius_normalize_step = 150):
        self.radius = radius
        self.radius_normalize_step = radius_normalize_step

    def fit(self, data):

        if self.radius == None:
            all_data_centroid  = np.average(data, axis=0)
            all_data_norm = np.linalg.norm(all_data_centroid)
            self.radius = all_data_norm/self.radius_normalize_step

        centroids = {}

        #initialize centroids
        for i in range(len(data)):
            centroids[i] = data[i]

        weights = [i for i in range(self.radius_normalize_

        while True:
            new_centroids = []
            for i in centroids:
                in_range = []
                centroid = centroids[i]

                for featureset in data:
                    distance = np.linalg.norm(featureset-centroid)
                    if distance == 0:
                        distance = 0.0000000001
                    weight_index = int(distance/self.radius)
                    if weight_index > self.radius_normalize_step-1:
                        weight_index = self.radius_normalize_step-1
                    to_add = (weights[weight_index]**2)*[featureset]
                    in_range += to_add

                new_centroid = np.average(in_range, axis=0)
```

Variable explorer

Name	Type	Size	Value
batch_size	int	1	100
mnist	contrib.learn.python.learn.datasets.base.Datasets	3	Datasets object of…
n_classes	int	1	10
n_nodes_hl1	int	1	500
n_nodes_hl2	int	1	500
n_nodes_hl3	int	1	500

Variable explorer | File explorer | Help

IPython console

Console 1/A

```
See `tf.nn.softmax_cross_entropy_with_logits_v2`.

Epoch 0 completed out of 10 loss: 1666037.4677734375
Epoch 1 completed out of 10 loss: 377809.3128890991
Epoch 2 completed out of 10 loss: 201302.4857263565
Epoch 3 completed out of 10 loss: 119427.91378033161
Epoch 4 completed out of 10 loss: 72651.25679710507
Epoch 5 completed out of 10 loss: 45327.621502393486
Epoch 6 completed out of 10 loss: 31955.17812934518
Epoch 7 completed out of 10 loss: 23664.356108333137
Epoch 8 completed out of 10 loss: 18248.740643078025
Epoch 9 completed out of 10 loss: 19962.00085876091
Accuracy: 0.9511

In [2]:
```

IPython console | History log

HANDS ON