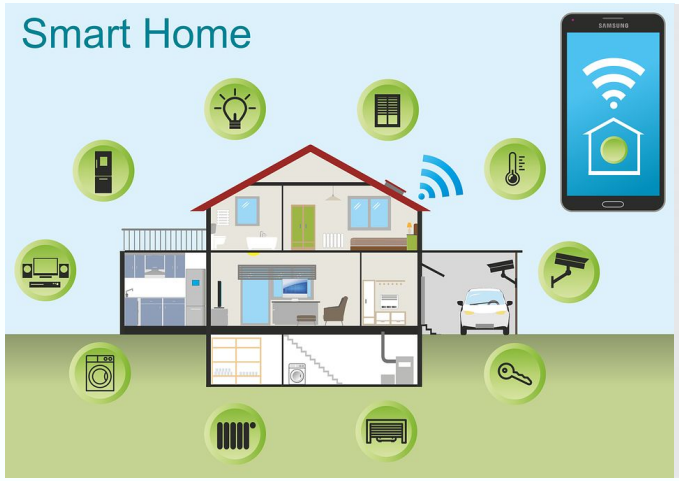
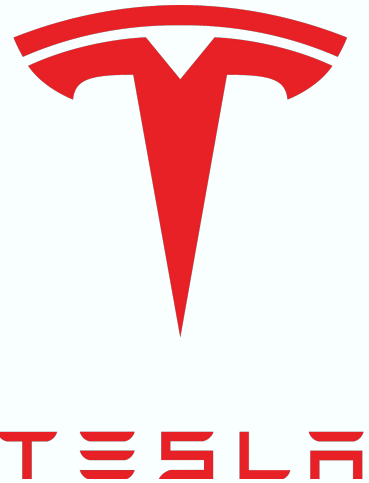
Anshori - @ans4175
Iqbal - @miqbal
Dimas - @dimas.sumartoyo

DSW Workshop IoT

eFishery

What is IoT?
It’s “Internet of Things”



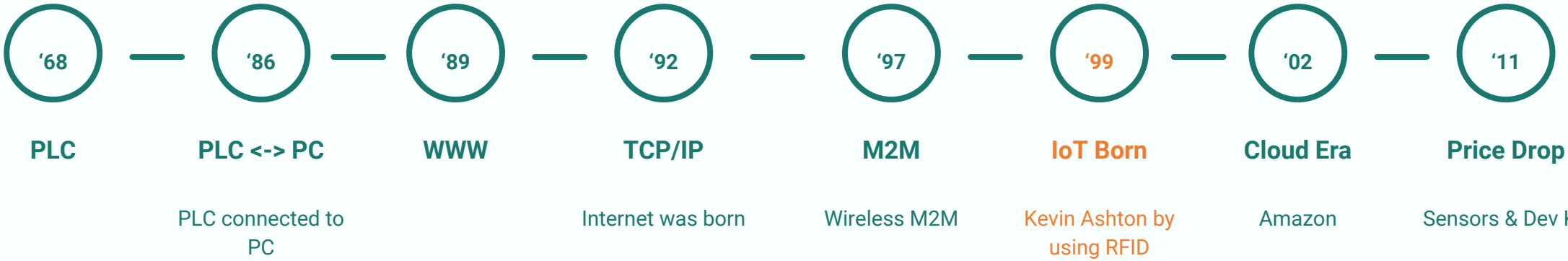


@internetofshit

eFishery

IoT Timeline

Before we had internet of things (IoT), there were **sensornets**….



eFishery

THE MARKET IS EXPECTED TO BE ASTRONOMICAL

2015:
- 10 billion connected things
- $1.9 billion from IoT services

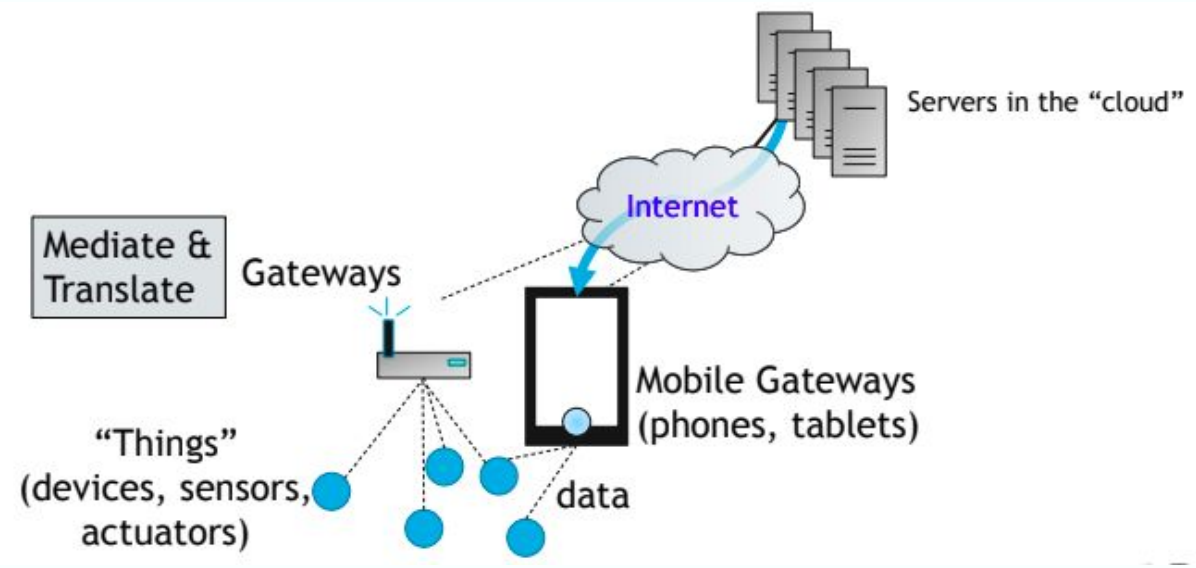
2020:
- ABI: 250,000 connected cars
- IDC: $7 billion from IoT services
- Gartner: $300 billion from IoT products
- IDC: Global IoT market $7.1 trillion
- ABI: 40 billion IoT devices

2035
- GE: $10-15 trillion added to GDP
- Cisco: $19 trillion
- ABI: 450 million IoTcars

http://www.eetimes.com/author.asp?doc_id=1326597

eFishery

Buzzwords Simplified



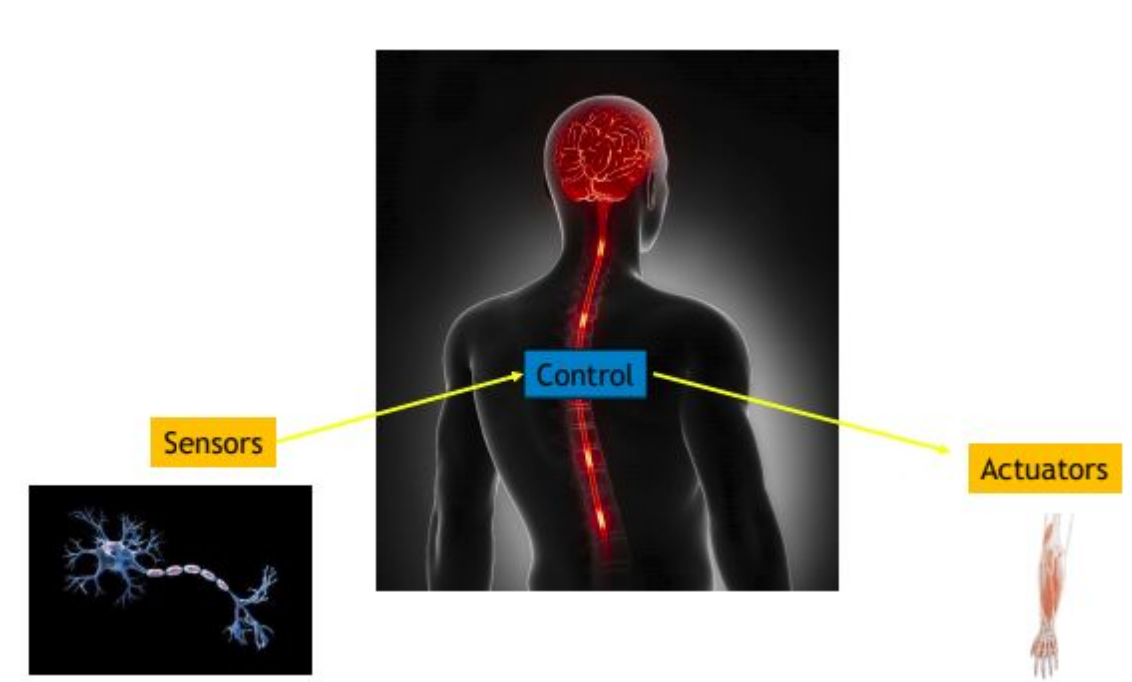
Connected Things
plus
Internet
and
Context

eFishery

THINGS

eFishery

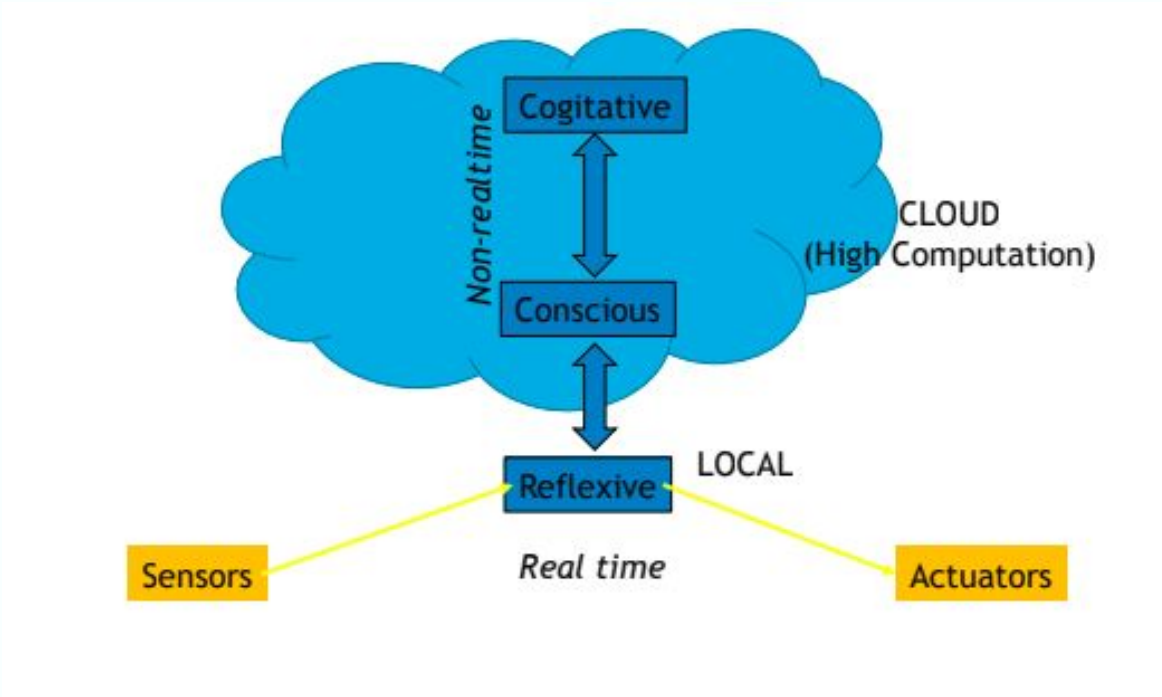
Things



Things are like
extension
of
human body

eFishery

Things



Non-Realtime Action
(**Cloud Computing**)

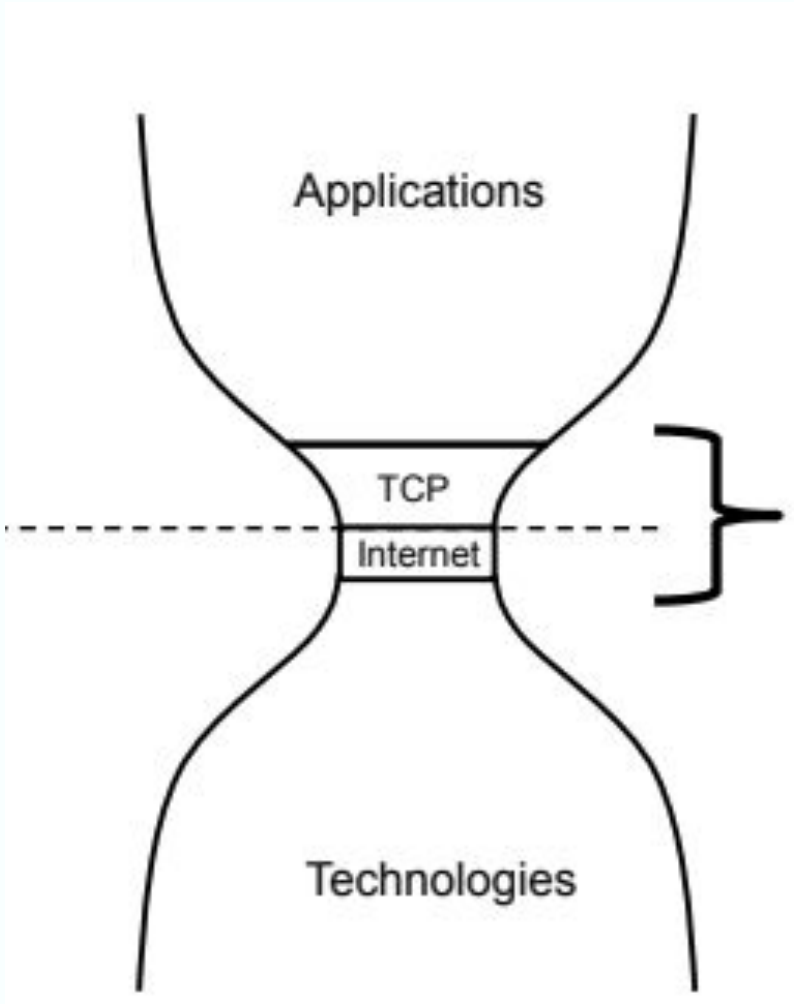
Realtime Action
(**Fog/Edge Computing**)

eFishery

INTERNET

eFishery

Internet



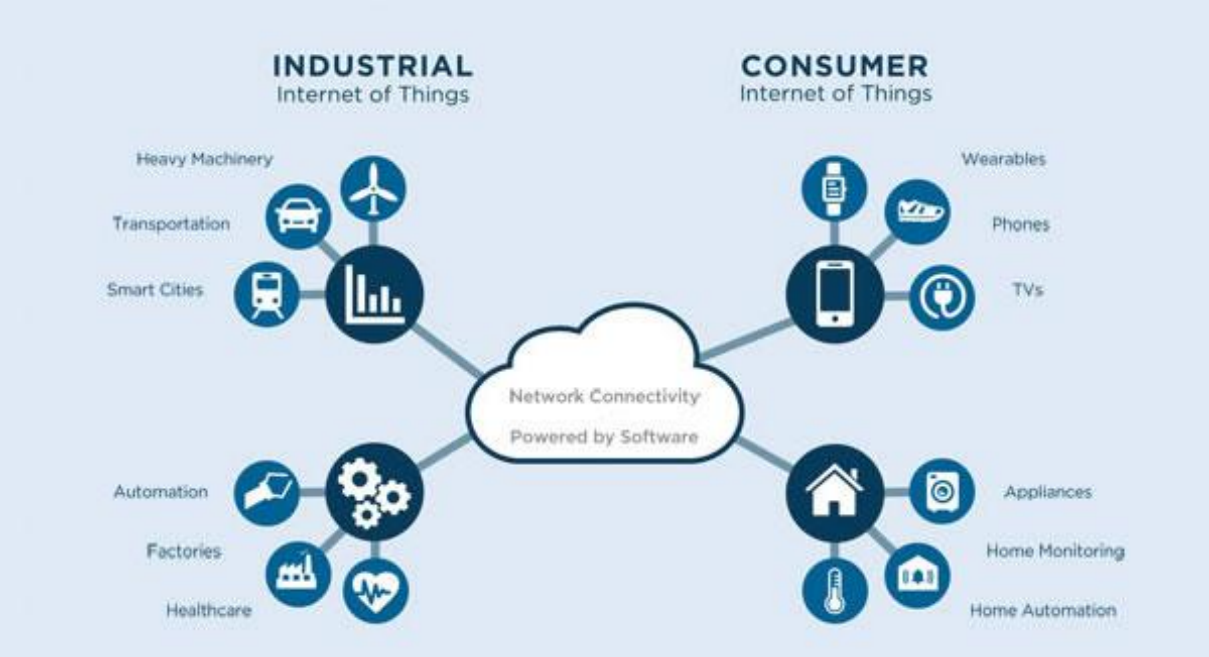
Take all good practices from **Internet Tech**

eFishery

PRESENTS

eFishery

Market & Use Cases



INDUSTRIAL much realizable
but
CONSUMER will be massive

eFishery

Local Context

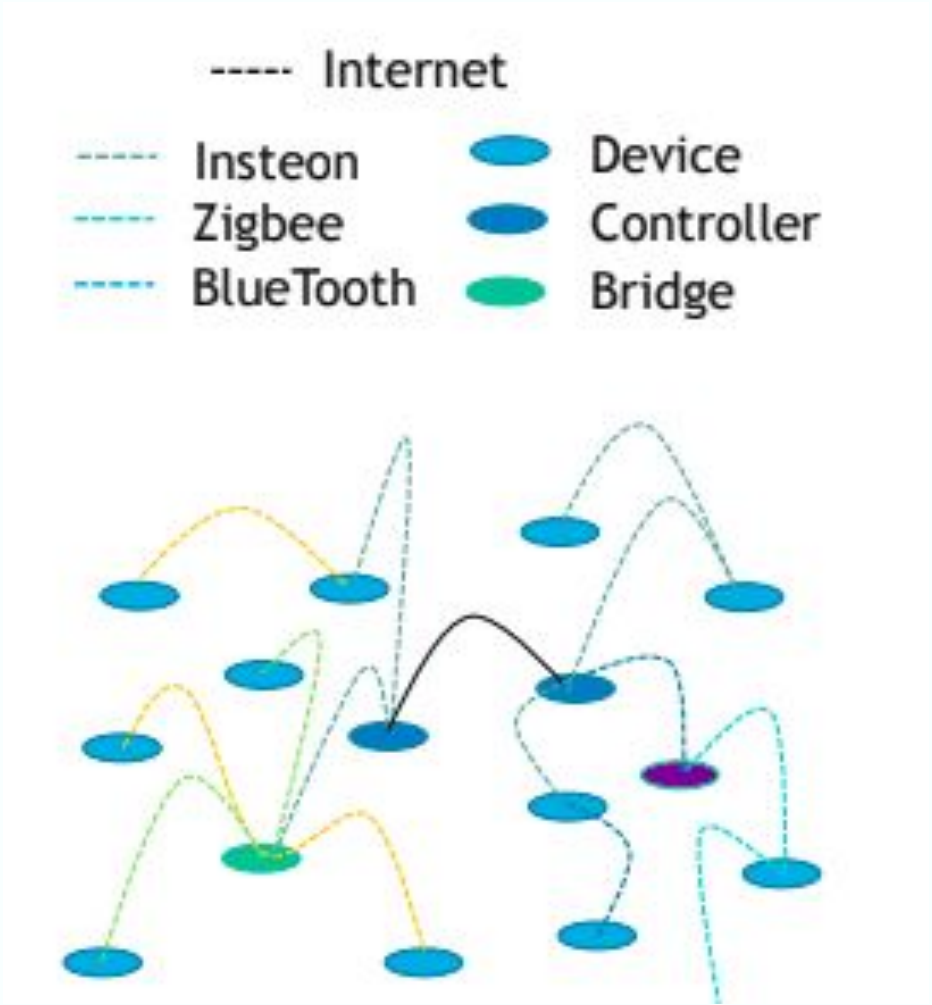
Product Design 101

People & Culture

Internet Coverage

eFishery

Interoperability Problems



Closed Ecosystems
on ground

eFishery

Standardization

- Regulation
  - Protocol
  - Bandwidth
  - Platform and Data
  - TKDN

Mass adoption will eventually be **STANDARD**

eFishery

Think Big Start Small

1. How to **Sense** and **Act** first

2. Talk about **Basic Connectivity** later

3. Prepare how to connect Things and Internet (**Locally** and **Globally**)

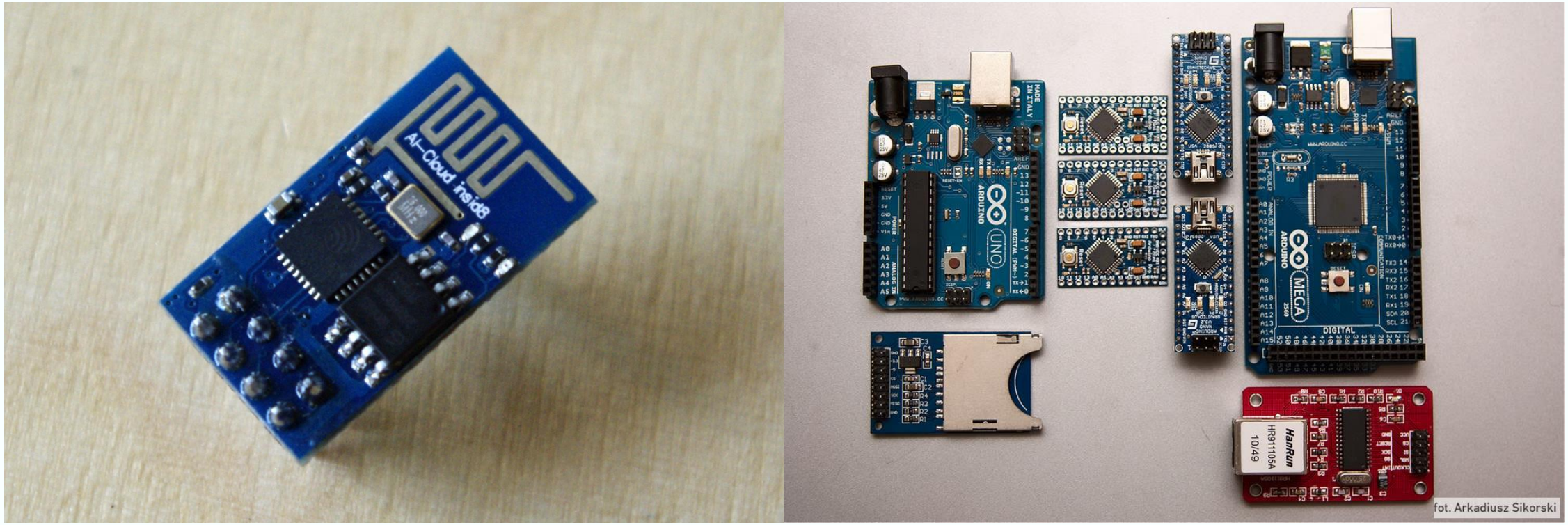
BASIC

eFishery

IoT Cloud Broker

eFishery

Devices and Sensors



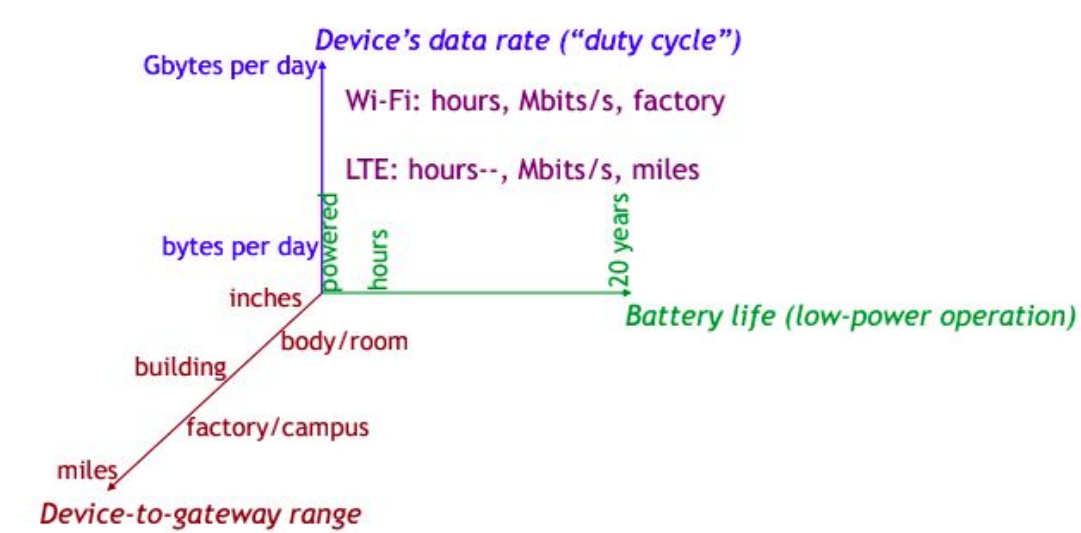
(a lot literally A LOT)

eFishery

Protocols

eFishery

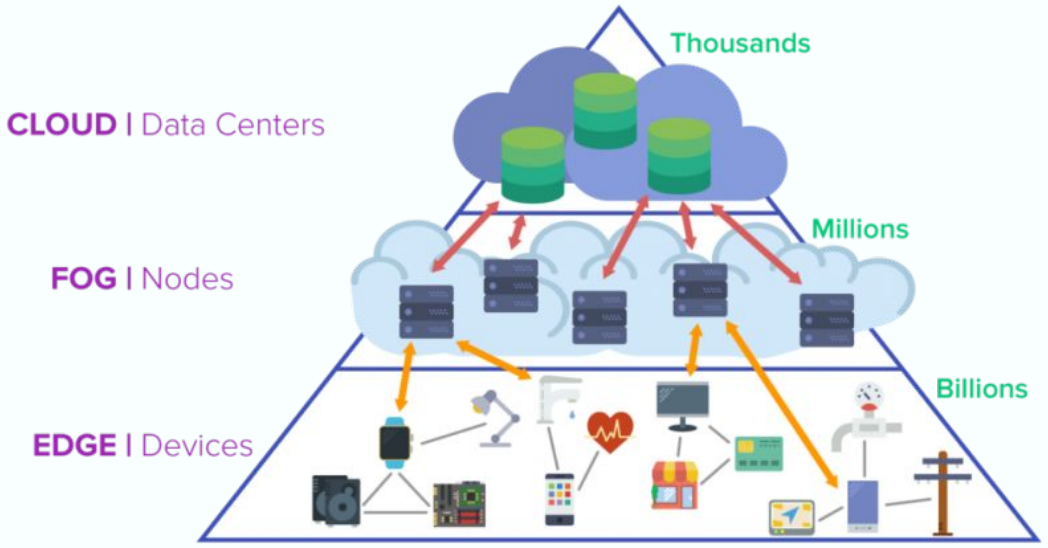
Protocols



Pick **two** out of **three**

eFishery

Fog or Edge Computing



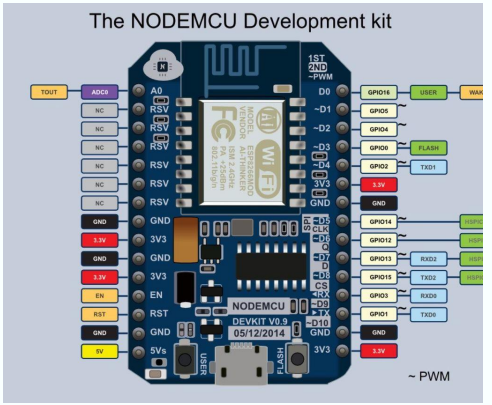
Intelligence on Gateway, it's **Fog**

Intelligence on Devices, it's **Edge**

eFishery

How To Start IoT Project

- Get a DevKit (Nodemcu ESP8266 / ESP32)



- Start using Framework (Sming OR Arduino)
- Use Open Source Broker, ex. Mosquitto (MQTT)
- ….rest is usual programming for web and cloud

eFishery

QUESTIONS?

eFishery

INTERMEDIATE

Install Arduino IDE

Download it here:
https://www.arduino.cc/en/Main/Software

Install Arduino IDE

Install toolchain ESP8266:

- Open Arduino IDE, open File > Preferences
- In Additional Board Manager URLs field enter
`http://arduino.esp8266.com/stable/package_esp8266com_index.json`
- Open Tools > Board > Boards Manager and wait until package indexing complete
- Search for `esp8266` and install it, or don't. Downloads will take a long long time but we can provide it!
- Select the board by choosing it from Tools > Board > WeMos D1 R2 & mini

eFishery

Install Libs

Install PubSubClient Library:

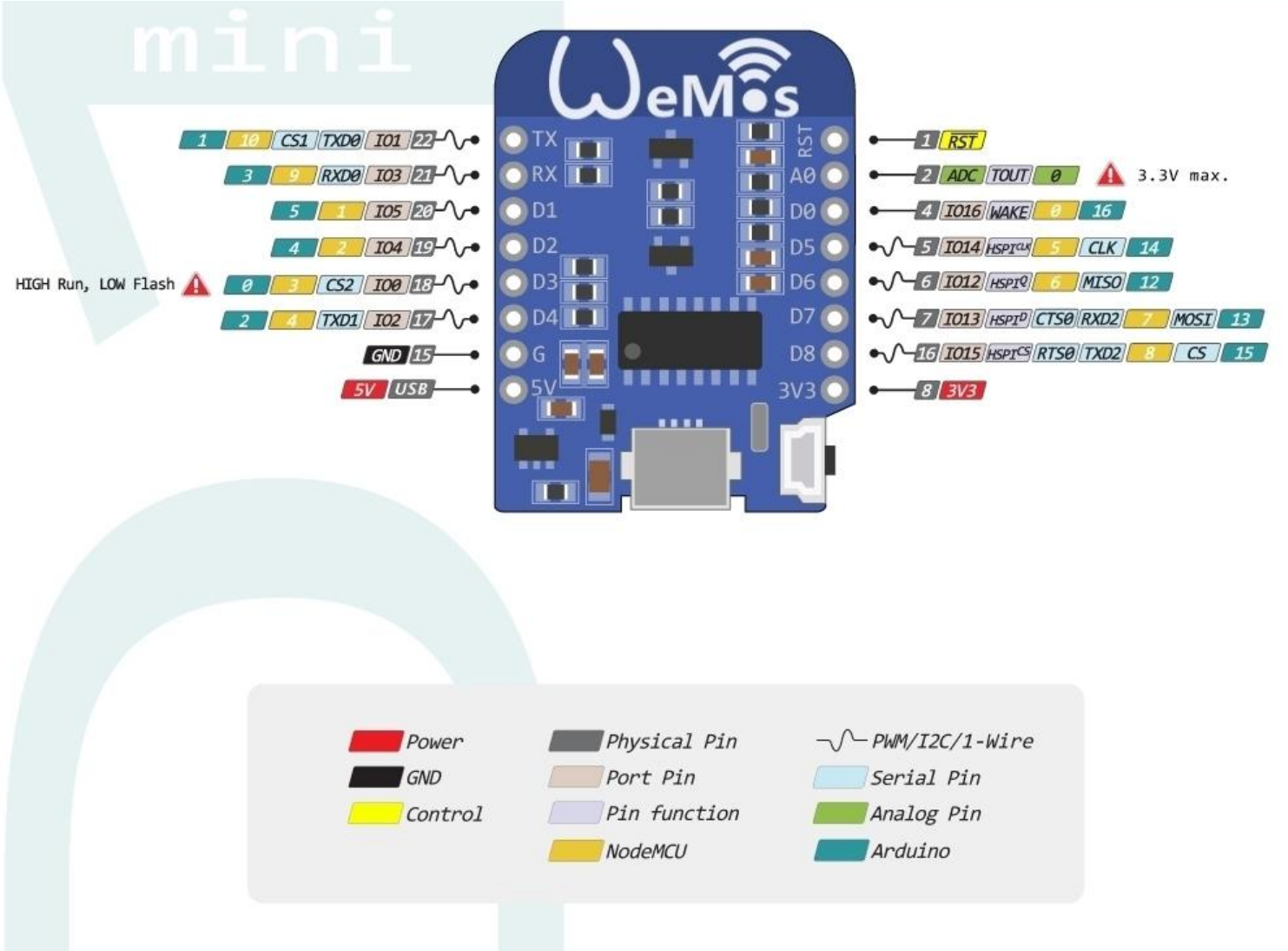
- Open Arduino IDE, open Sketch > Include Library > Manage Libraries
- Search for `PubSubClient` and install it.

Install ESP AsyncWebServer Library:

- Download zip file from: https://platformio.org/lib/show/1826/AsyncTCP/installation and https://platformio.org/lib/show/306/ESP%20Async%20WebServer/installation
- Open Arduino IDE, open Sketch > Include Library > Add .ZIP Library

eFishery

Wemos D1 Mini



eFishery

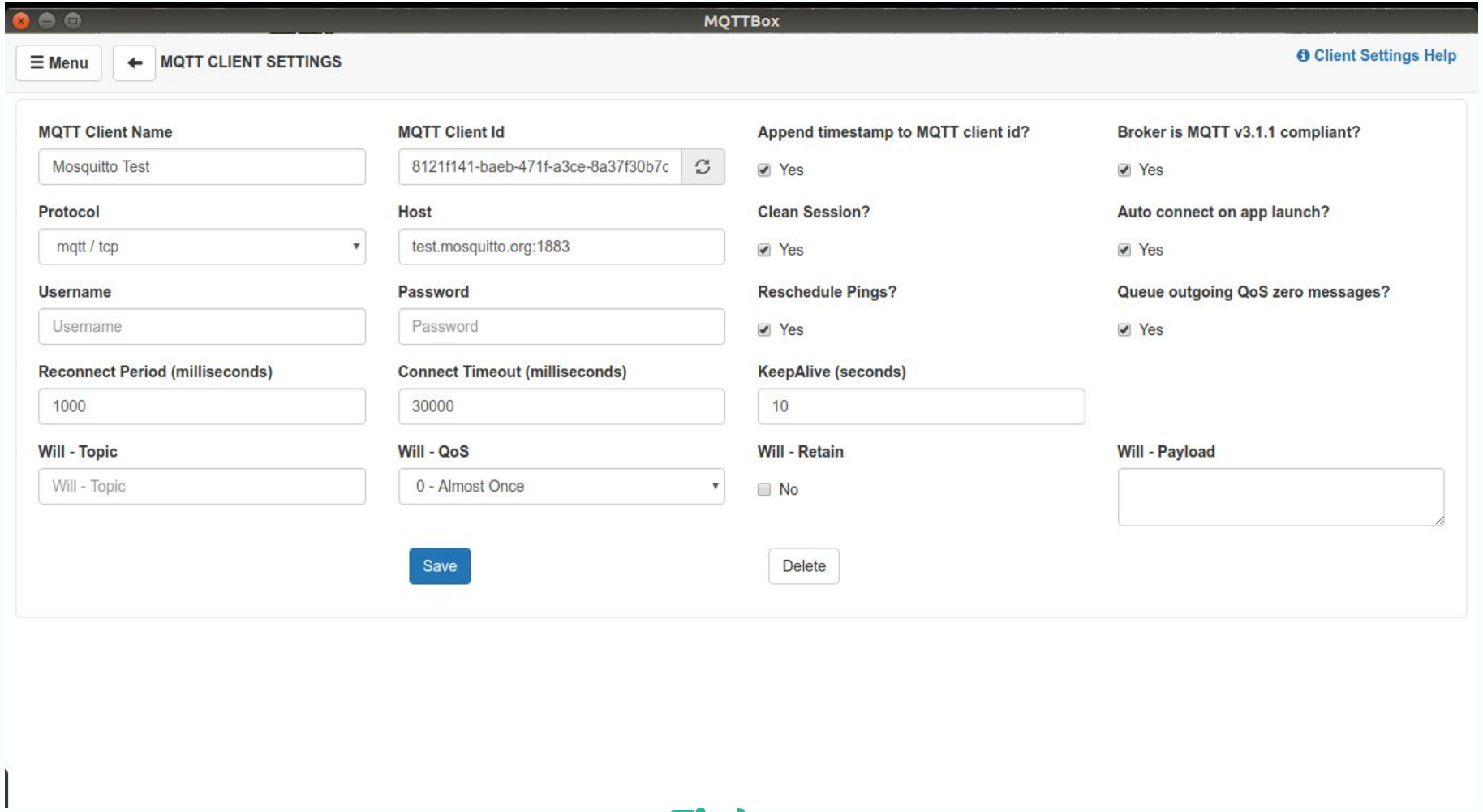
Setup MQTT Broker

Go to:
https://test.mosquitto.org/

Install MQTTBox

https://chrome.google.com/webstore/detail/mqttbox/kaajoficamnjijhkeomgfljpicifbkaf or
http://workswithweb.com/html/mqttbox/downloads.html

Install MQTTBox



eFishery

Connect ESP to MQTT Broker (WiFi)

- Source code: https://bitbucket.org/snippets/efishery/re87pk#file-Basic.cpp

eFishery

Connect ESP to MQTT Broker (WiFi)

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

#define WIFI_SSID "EDIT SSID WIFI"
#define WIFI_PASS "EDIT PASSWORD WIFI"

const char* mqtt_server = "test.mosquitto.org";

WiFiClient espClient;
PubSubClient client(espClient);
String clientId;

void setup() {
  Serial.begin(115200);
  delay(10);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

  // generate client id untuk koneksi mqtt
  randomSeed(micros());
  clientId = "DSWClient-";
  clientId += String(random(0xffff), HEX);
}
```

Connect ESP to MQTT Broker (WiFi)

```
void callback(char* topic, byte* payload, unsigned int length)
{
  //print message to serial
  Serial.print("Incoming Message (");
  Serial.print(topic);
  Serial.print("): ");
  for (int i = 0; i < length; i++)
  {
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

void setup_wifi()
{
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WIFI_SSID);

  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_SSID, WIFI_PASS);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

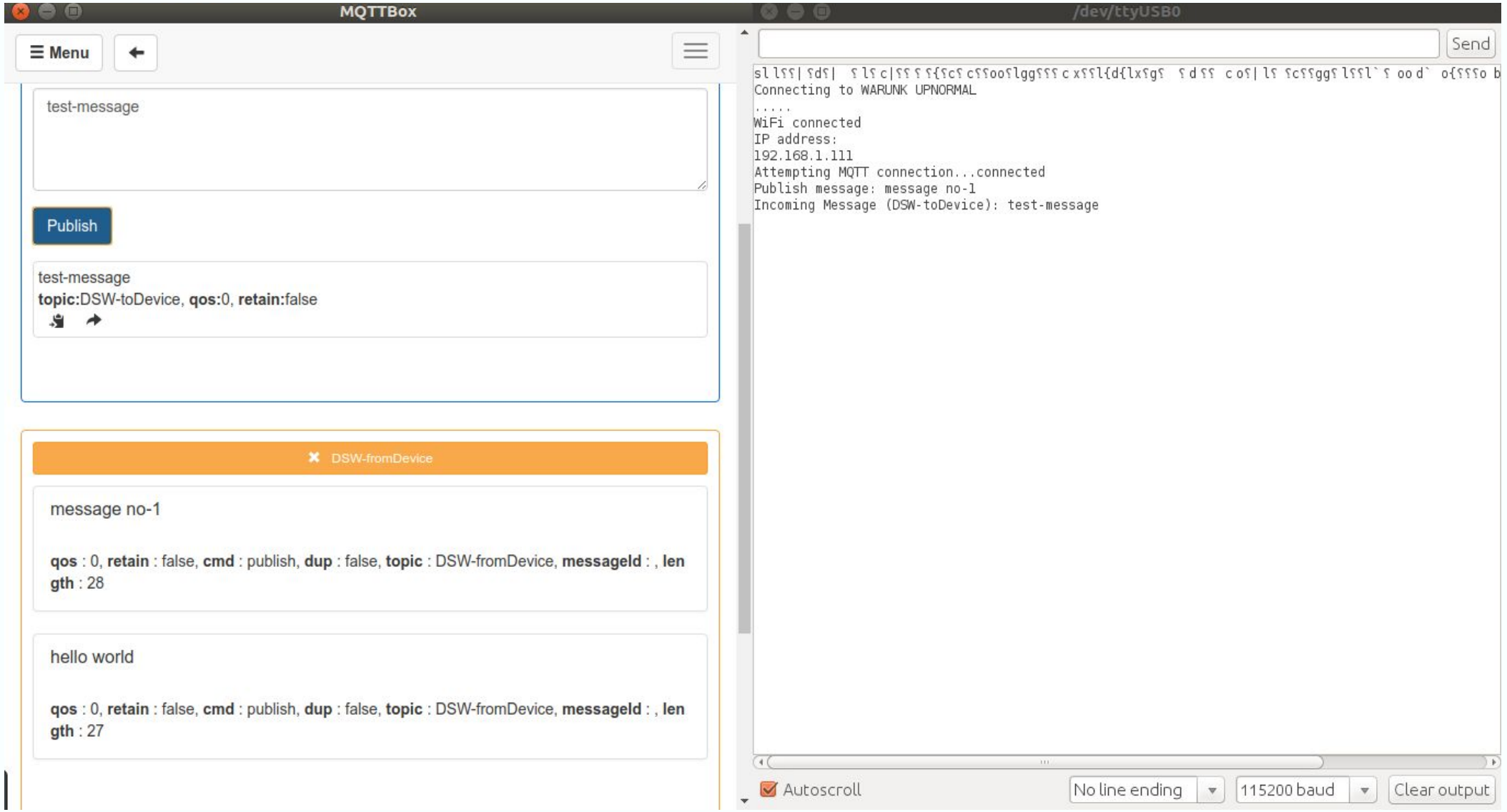
Connect ESP to MQTT Broker (WiFi)

```
void connectMqtt()
{
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");
    if (client.connect(clientId.c_str()))
    {
      Serial.println("connected");
      client.publish("DSW-fromDevice", "hello world");
      client.subscribe("DSW-toDevice");
    } else {
      Serial.print("failed, code=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

Connect ESP to MQTT Broker (WiFi)

```
long lastPubTime = 0;
int sequence = 0;
#define PUBLISH_INTERVAL_MS 30000 //30 detik

void loop()
{

  if (!client.connected())
  {
    connectMqtt();
  }
  client.loop();

  long now = millis();
  if (now - lastPubTime > PUBLISH_INTERVAL_MS)
  {
    lastPubTime = now;
    sequence++;

    String msg = "message no-";
    msg += sequence;

    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish("DSW-fromDevice", msg.c_str());
  }
}
```

Test Action!


MQTTBox
Menu
test-message
Publish
test-message
topic:DSW-toDevice, qos:0, retain:false
DSW-fromDevice
message no-1
qos : 0, retain : false, cmd : publish, dup : false, topic : DSW-fromDevice, messageId : , length : 28
hello world
qos : 0, retain : false, cmd : publish, dup : false, topic : DSW-fromDevice, messageId : , length : 27
/dev/ttyUSB0
Send
Connecting to WARUNK UPNORMAL
.....
WiFi connected
IP address:
192.168.1.111
Attempting MQTT connection...connected
Publish message: message no-1
Incoming Message (DSW-toDevice): test-message
Autoscroll
No line ending
115200 baud
Clear output

QUESTIONS?

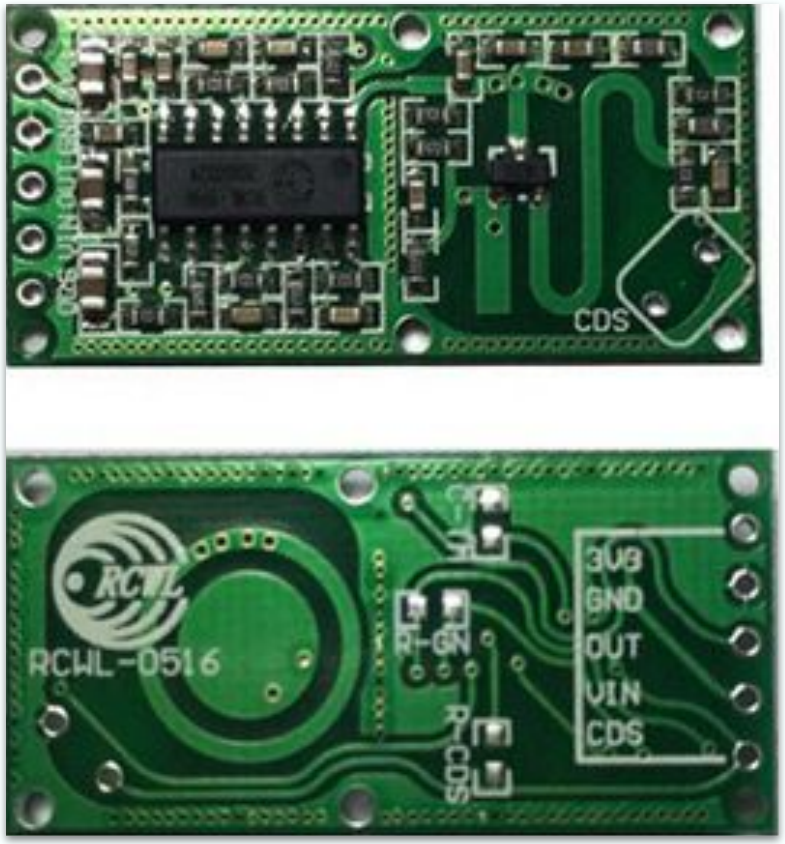
eFishery

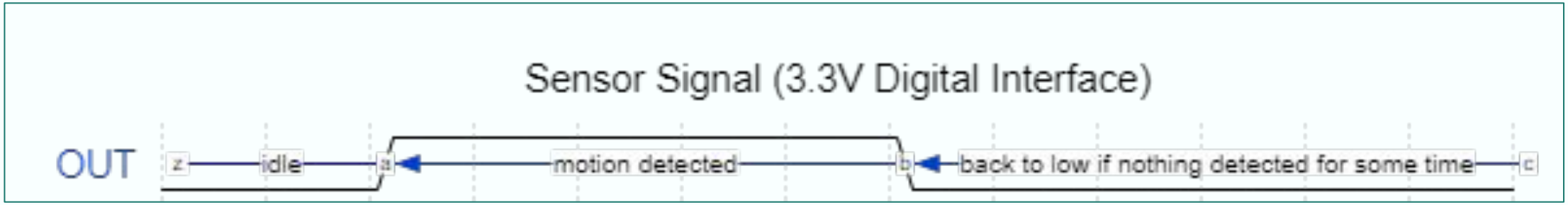
Intro to Small IoT Project

In this section towards, we will build a simple IoT use case example.
We will build lamp on/off switch using motion sensor.
We will also integrate our lamp’s switch using MQTT command.

Sensor: Microwave doppler effect radar human body motion detector

- Using RCWL-0516 Module
- Simplified version of radar
- Detecting body movement instead of speed and location



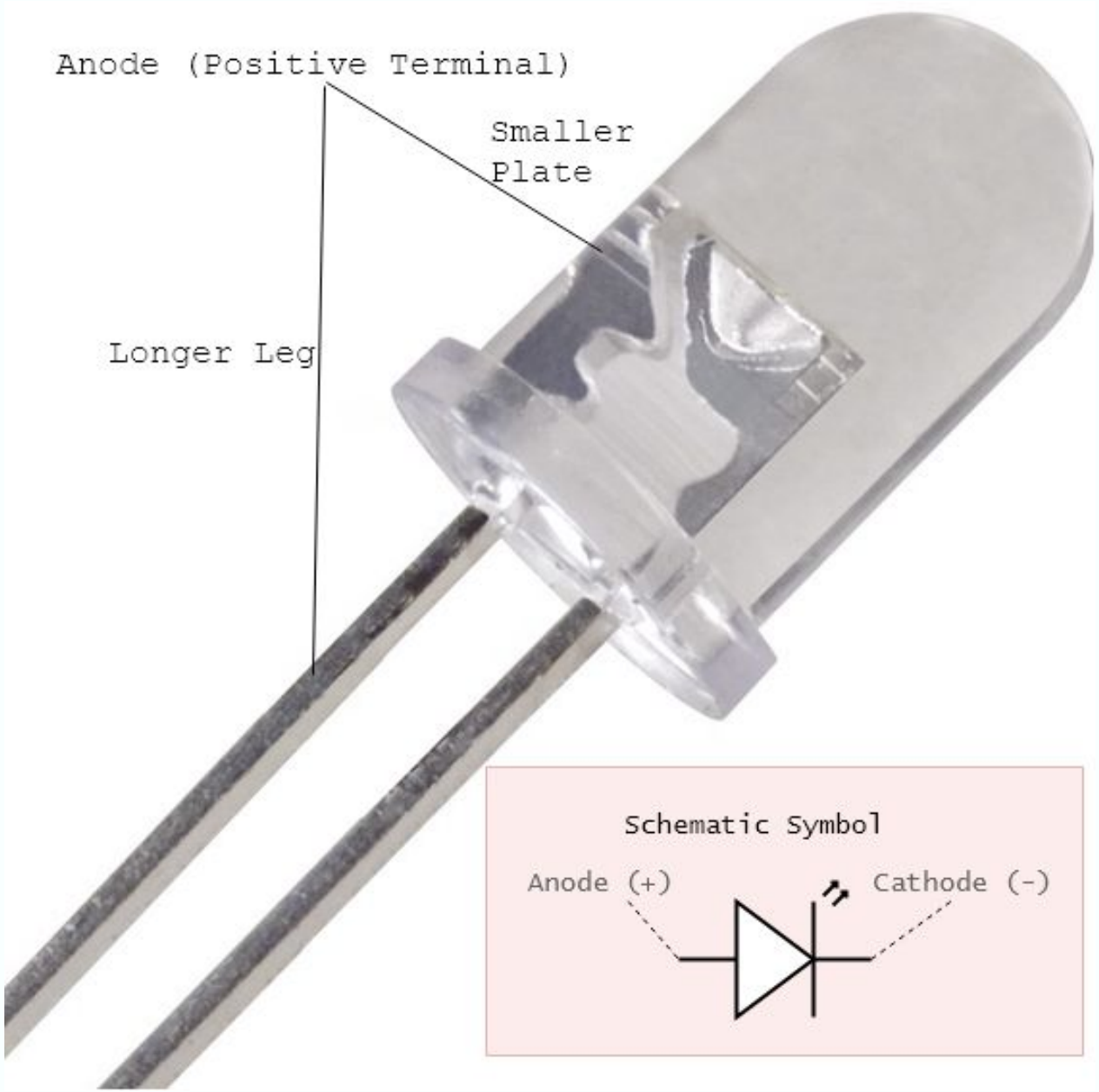


For comprehensive explanation, visit: https://github.com/jdesbonnet/RCWL-0516 (**warning:** Technical roadblock ahead!)

eFishery

Actuator: LED

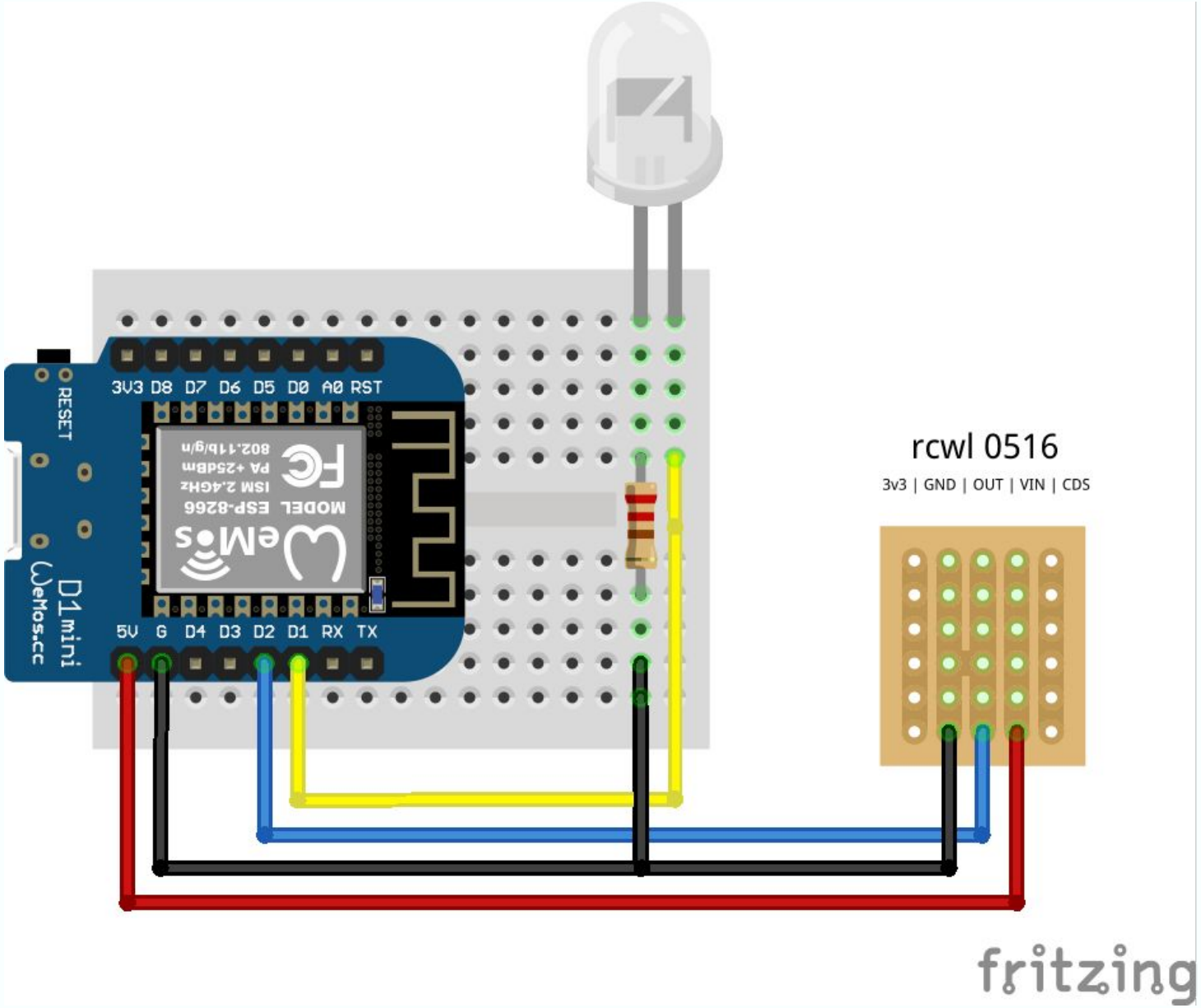
- A type of lamp
- Many shape and size, almost everywhere
- Used for:
  - Indicator
  - Sign
  - Display
  - lighting
- Watch the polarity



eFishery

Schematic Sensor-Actuator

- Wemos D1 Mini
- LED (5mm)
- Resistor 220Ω
- RCWL-0516
- Breadboard
- A bunch of jumper cable



eFishery

Wemos D1 Mini Wiring Connection

Wemos D1 Mini Pin	Peripheral
D1	LED Anode
D2	OUT (RCWL-0516)
5V	VIN (RCWL-0516)
GND	Common Ground

Assembly & Wiring

Let’s wire it all!

eFishery

Firmware Coding: Sensing & Actuating

- Source code: https://bitbucket.org/snippets/efishery/re87pk#file-Intermediate_I.cpp

Firmware Coding: Sensing & Actuating

```
//atur Konfigurasi Wifi
#define WIFI_SSID "EDIT SSID WIFI"
#define WIFI_PASS "EDIT PASSWORD WIFI"

//atur konfigurasi sensor (RCWL-0516) dan lampu
#define PIN_RCWL D2
#define PIN_LAMP D4
#define RCWL_ACTIVE HIGH
#define LAMP_ON HIGH
#define LAMP_OFF LOW

//atur konfigurasi MQTT server / broker
const char* mqttServer = "test.mosquitto.org";
int mqttPort = 1883;

//Berhubungan dengan koneksi MQTT
WiFiClient espClient;
PubSubClient client(espClient);
String clientId;

//Berhubungan dengan sensor
volatile bool sensorState;
bool lastState;
```

eFishery

Firmware Coding: Sensing & Actuating

```
void setupSensor()
{
  pinMode(PIN_RCWL, INPUT);
  pinMode(PIN_LAMP, OUTPUT);
  pinMode(D1, OUTPUT);
  digitalWrite(D1, LOW);
  digitalWrite(PIN_LAMP, LAMP_OFF);
  sensorState = digitalRead(PIN_RCWL);
  lastState = sensorState;
  attachInterrupt(digitalPinToInterrupt(PIN_RCWL), onSensorChange, CHANGE);
}

//Interrupt handler jika ada perubahan pada pin RCWL
void onSensorChange()
{
  sensorState = digitalRead(PIN_RCWL);
  if(sensorState == RCWL_ACTIVE)
  {
    digitalWrite(PIN_LAMP, LAMP_ON);
  } else {
    digitalWrite(PIN_LAMP, LAMP_OFF);
  }
}
```

eFishery

Firmware Coding: Sensing & Actuating

```
void loop()
{
  //Monitor koneksi MQTT
  if (!client.connected())
  {
    connectMqtt();
  }
  client.loop();

  //Cek interval waktu untuk publish message
  long now = millis();
  if (now - lastPubTime > PUBLISH_INTERVAL_MS)
  {
    lastPubTime = now;
    sequence++;

    String msg = "message no-";
    msg += sequence;

    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish("DSW-fromDevice", msg.c_str());
  }

  //Menampilkan pesan perubahan input dari sensor
  bool currentState = sensorState;  //baca dulu state secara lokal
  if(lastState != currentState)
  {
    lastState = currentState;
    if(currentState == RCWL_ACTIVE)
    {
      Serial.println("gerakan terdeteksi!");
    } else {
      Serial.println("Senyap");
    }
  }
}
```

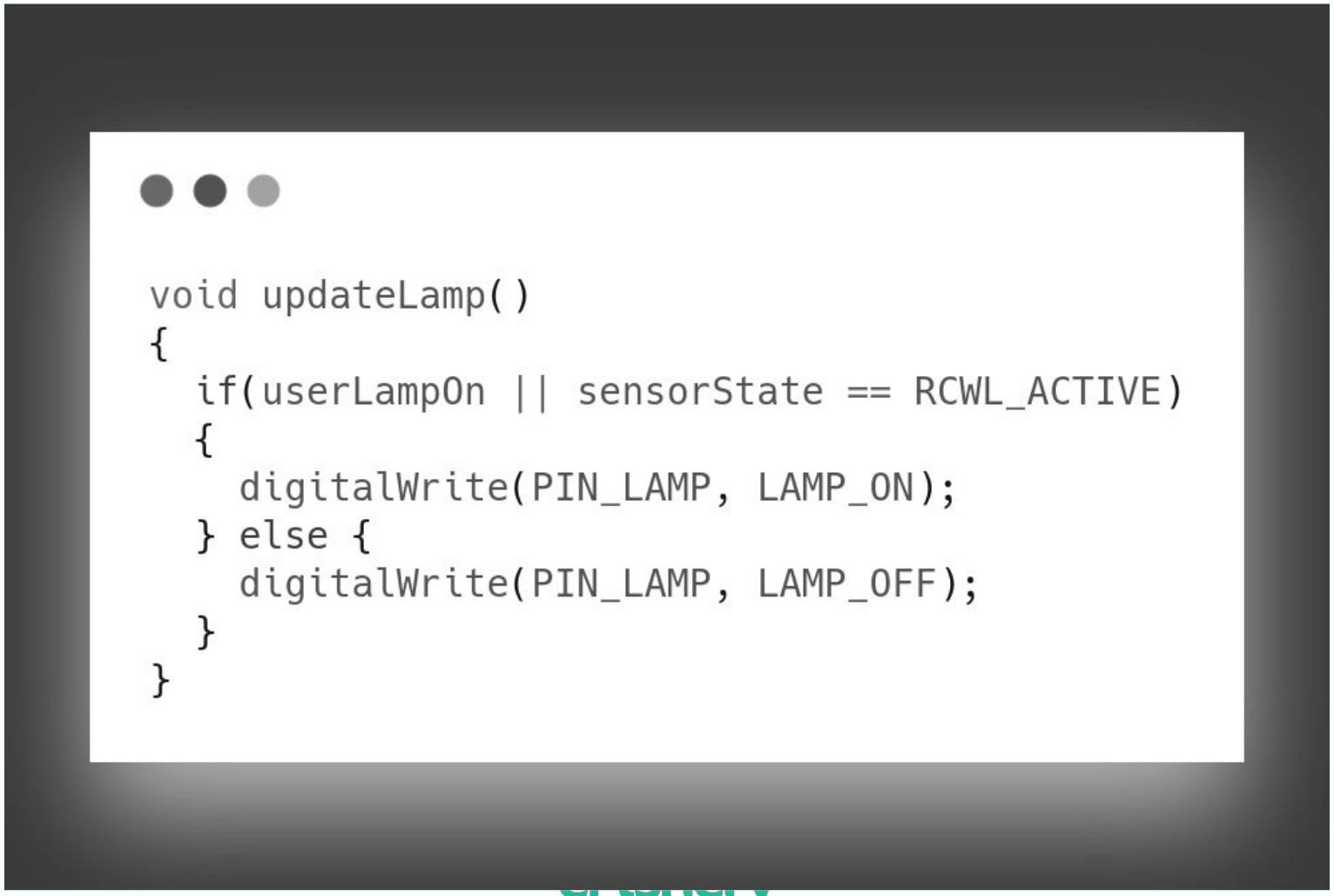
Firmware Coding: MQTT Handling

- Source code: https://bitbucket.org/snippets/efishery/re87pk#file-Intermediate_II.cpp

Firmware Coding: MQTT Handling

```
//Berhubungan dengan sensor dan lampu
volatile bool sensorState;
bool userLampOn; //state lampu dari user
bool lastState;
```

Firmware Coding: MQTT Handling



```
void updateLamp()
{
  if(userLampOn || sensorState == RCWL_ACTIVE)
  {
    digitalWrite(PIN_LAMP, LAMP_ON);
  } else {
    digitalWrite(PIN_LAMP, LAMP_OFF);
  }
}
```

Firmware Coding: MQTT Handling

```
//Interrupt handler jika ada perubahan pada pin RCWL
void onSensorChange()
{
  sensorState = digitalRead(PIN_RCWL);
  updateLamp();
}

void callback(char* topic, byte* payload, unsigned int length)
{
  //print message to serial
  Serial.print("Incoming Message (");
  Serial.print(topic);
  Serial.print("): ");
  for (int i = 0; i < length; i++)
  {
    Serial.print((char)payload[i]);
  }
  Serial.println();

  // Jika awal message adalah '1' berarti user ingin lampu terus menyala
  userLampOn = ((char)payload[0] == '1');
  updateLamp();
}
```

QUESTIONS?

eFishery

ADVANCE

eFishery

What is M2M

Machine-to-Machine (M2M) is direct communication between devices using any communication protocol

IoT = M2M + Internet

eFishery

Protocols

eFishery

Firmware Coding: Rest on ESP

- Source code: https://bitbucket.org/snippets/efishery/re87pk#file-XAdvance_I.cpp

Firmware Coding: Endpoint

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ESPAsyncWebServer.h>

//atur Konfigurasi Wifi
#define WIFI_SSID "EDIT SSID WIFI"
#define WIFI_PASS "EDIT PASSWORD WIFI"
#define SOFTAP_SSID "DSW-TEST1"
#define SOFTAP_PASSWORD "DSW-TEST1"

//atur konfigurasi sensor (RCWL-0516) dan lampu
#define PIN_RCWL D2
#define PIN_LAMP D1
#define RCWL_ACTIVE HIGH
#define LAMP_ON HIGH
#define LAMP_OFF LOW

//atur konfigurasi MQTT server / broker
const char* mqttServer = "test.mosquitto.org";
int mqttPort = 1883;

//Berhubungan dengan koneksi MQTT
WiFiClient espClient;
PubSubClient client(espClient);
String clientId;

//HTTP Server
AsyncWebServer server(80);
```

eFishery

Firmware Coding: Endpoint

```
void setup()
{
  Serial.begin(115200);
  delay(10);
  setupSensor();
  setupAP();

  //tambahkan endpoint dan mulai server
  ....
  server.begin();

  setupStation();
  client.setServer(mqttServer, 1883);
  client.setCallback(callback);

  // generate client id untuk koneksi mqtt
  randomSeed(micros());
  clientId = "DSWClient-";
  clientId += String(random(0xffff), HEX);
}
```

eFishery

Firmware Coding: Endpoint

```
server.on("/info", HTTP_GET, [](AsyncWebServerRequest *request)
{
  //susun response terlebih dahulu
  String responseText = "Lamp Status: ";
  if(userLampOn || sensorState == RCWL_ACTIVE)
  {
    responseText += "ON";
  } else {
    responseText += "OFF";
  }

  responseText += "\n";
  responseText += "WIFI SSID Setting: ";
  responseText += WiFi.SSID();

  responseText += "\n";
  responseText += "WIFI Connection Status: ";

  //stringfy semua status koneksi WiFi
  //daftar lengkap state ada pada file
  //https://github.com/esp8266/Arduino/blob/master/libraries/ESP8266WiFi/src/include/wl_definitions.h
  switch (WiFi.status()) {
    case WL_IDLE_STATUS:
      responseText += "Idle";
    break;
    case WL_NO_SSID_AVAIL:
      responseText += "Wifi AP not found";
    break;
    case WL_SCAN_COMPLETED:
      responseText += "Wifi scan finished";
    break;
    case WL_CONNECTED:
      responseText += "Connected, IP address ";
      responseText += WiFi.localIP().toString();
    break;
    case WL_CONNECT_FAILED:
      responseText += "Failed to connect";
    break;
    case WL_CONNECTION_LOST:
      responseText += "lost connection";
    break;
    case WL_DISCONNECTED:
      responseText += "Has been disconnected";
    break;
    default:
      responseText += "Unknown";
  }

  //kirim response
  request->send(200, "text/plain", responseText);
});
```

eFishery

Firmware Coding: Endpoint

```
server.on("/lamp", HTTP_GET, [](AsyncWebServerRequest *request)
{
  String responseText = "Please send query parameter 'command' with value 'on' or 'off'";
  if(request->hasParam("command"))
  {
    AsyncWebParameter* command = request->getParam("command");
    if(command->value() == "on" || command->value() == "off")
    {
      responseText = "user set lamp to ";
      responseText += command->value();
      // Jika command 'on' berarti user ingin lampu terus menyala
      userLampOn = (command->value() == "on");
      responseText = "user set lamp to ";
      responseText += userLampOn;
      updateLamp();
    }
  }

  //kirim response
  request->send(200, "text/plain", responseText);
});
```

eFishery

Firmware Coding: Endpoint

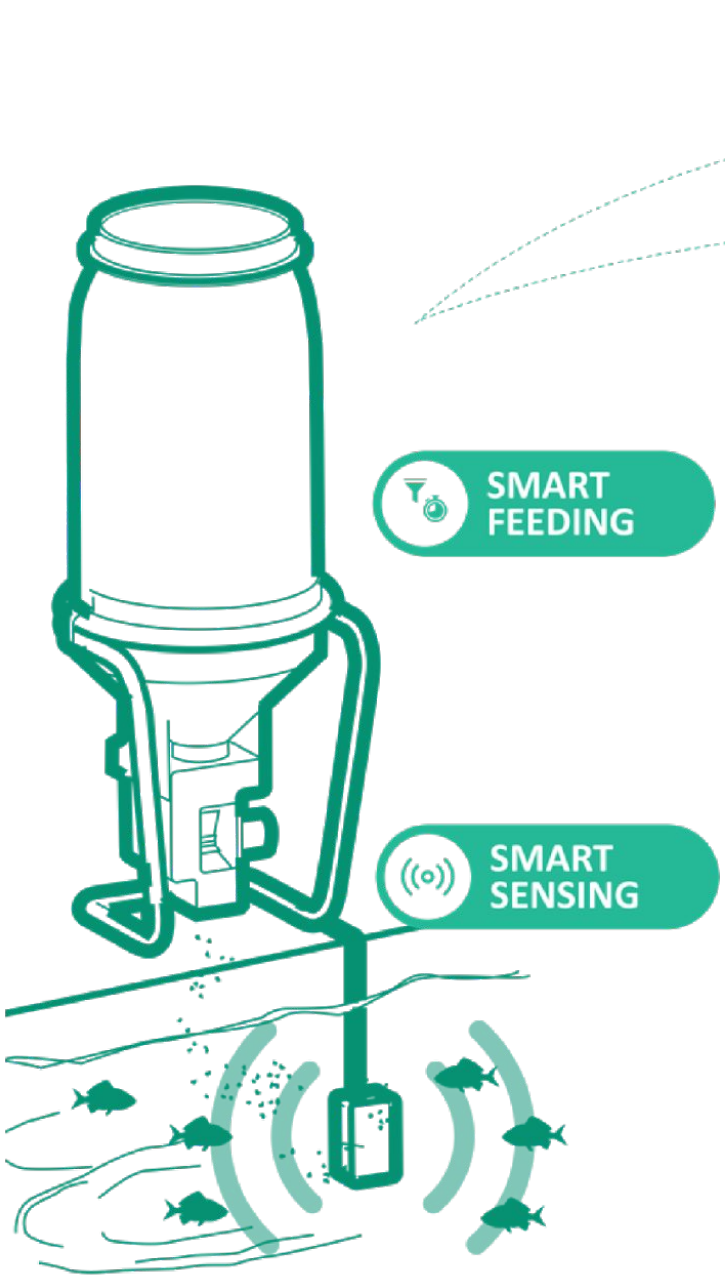
```
void setupAP()
{
  // WiFi.softAP(SOFTAP_SSID, SOFTAP_PASSWORD);
  WiFi.softAP(SOFTAP_SSID);

  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
}

// void setupWifi()
void setupStation()
{
```

eFishery

QUESTIONS?

eFishery

SMART FEEDING SOLUTION

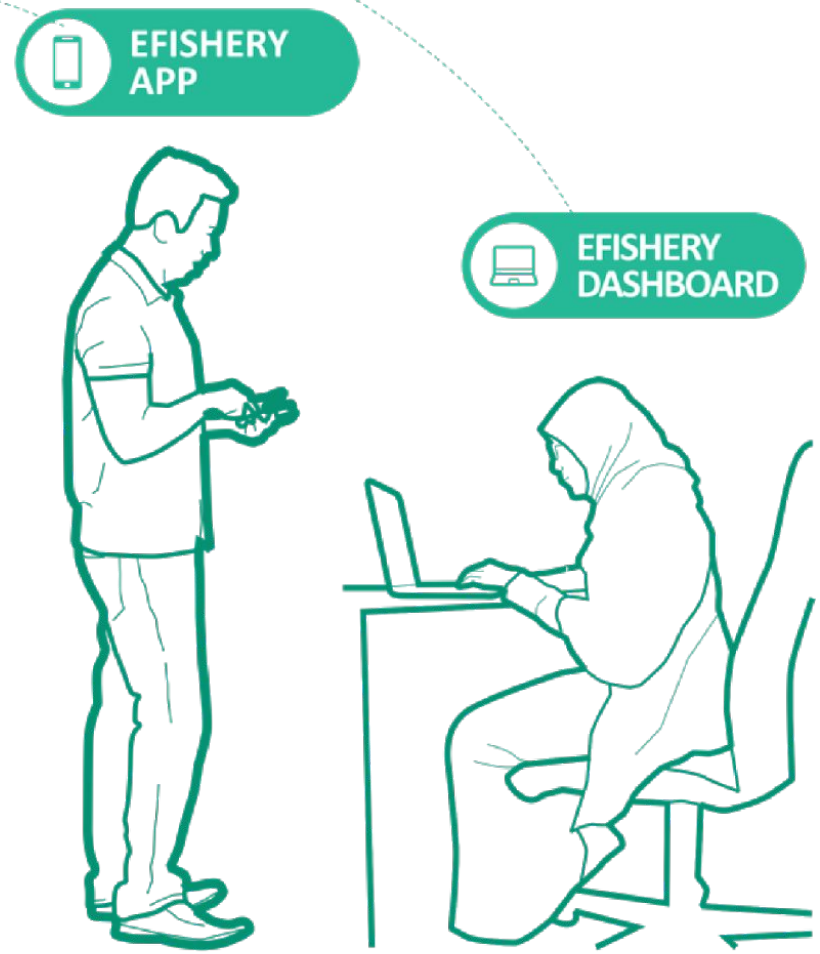


SMART feeding system for aquaculture,

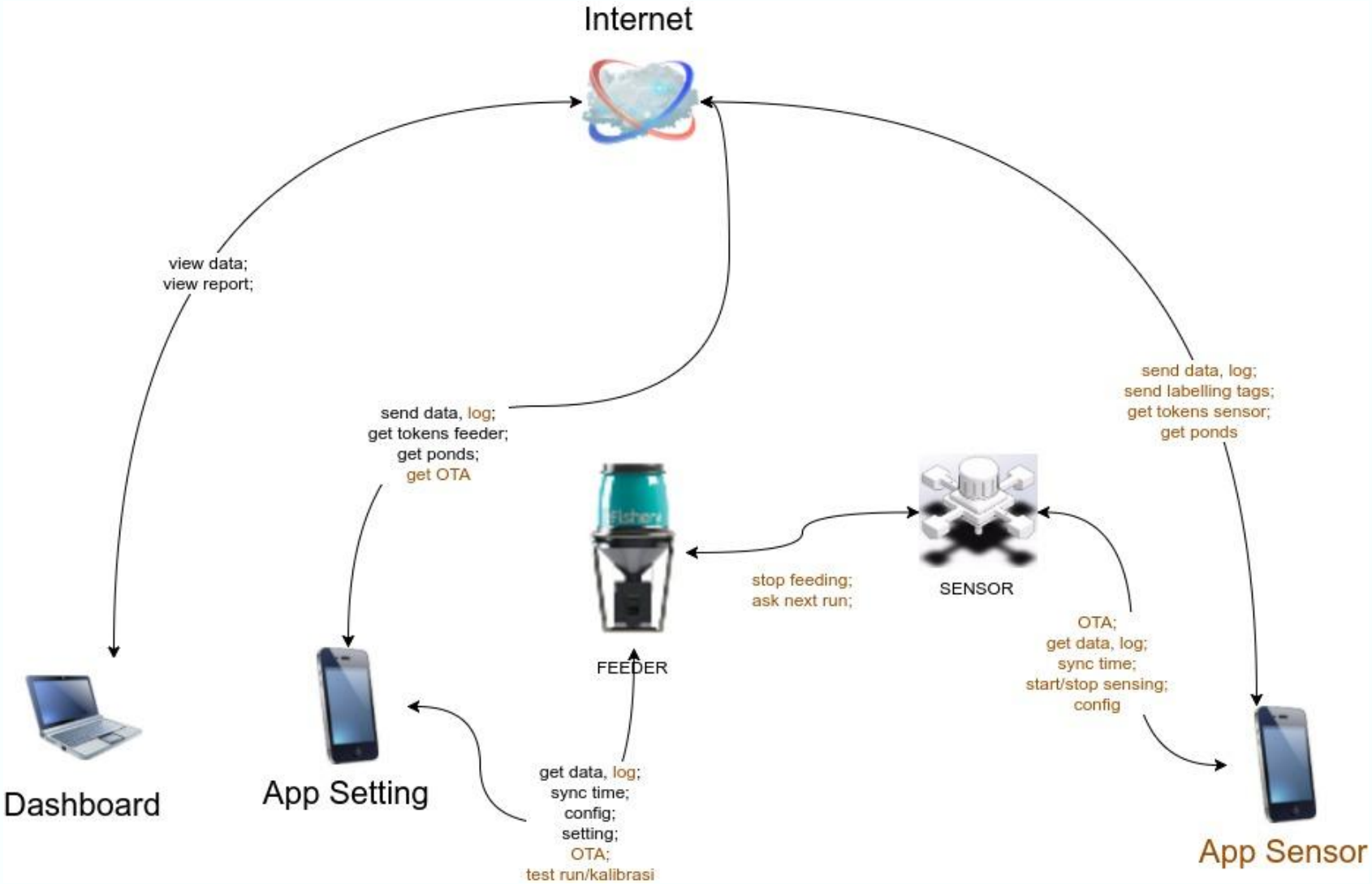
that can **SENSE** the fish's appetite,

and **CONNECTED** to smartphone and cloud.

Anytime. Anywhere.



IoT is about Context



eFishery

More Advance IoT Project

just join us

career@efishery.com

Subject: [firmware|software|android|data] engineer

eFishery