**IoT Installation Guide Example**

This guide documents how to install and deploy a development platform for IoT, we will use an example of facial analysis to implement and test. All artifacts for facial analysis are contained in the zip file that can be downloaded from [www.amphoria.com](http://www.amphoria.com)

Click here A black and yellow rectangle with white text

Description automatically generated then download the facialanaysis.zip file by clicking here A logo of a blue cube

Description automatically generated. We will reference this file later in this document.

Environments that can be used for this IoT Starter Platform:

* Raspberry Pi 3 or 4
  + Memory: 2GB minimum
  + Disk: 32GB SD Card
* A Windows desktop or laptop, a Linux Desktop or laptop, or an Apple Mac desktop or laptop.
  + Memory: 4GB minimum
  + Disk: 32GB
* TCP/IP network with Internet access

The following is a list of the installation steps.

## Node.js

[**Node.js**](https://en.wikipedia.org/wiki/Node.js) is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 JavaScript engine and executes code outside of a web browser.

In your browser, go to [www.nodejs.org](http://www.nodejs.org) and download the latest installation package for your platform, make sure you select the correct version for your platform. Note that for the Raspberry Pi we will choose ARM processor.

Graphical user interface, table

Description automatically generated

Follow the instructions for installation on your platform from the Node.js site. Once your installation is complete, check that it is properly installed as follows:

Open a command line window and do the following:

Type *node -v* and press return / enter.

Your results should look something like this:

Graphical user interface, text

Description automatically generated

The information returned will be the installed version, in this example it is version 14.17.6

## Node-Red

[**Node-RED**](https://en.wikipedia.org/wiki/Node-RED) is a flow-based development tool for visual programming originally developed by IBM to connect hardware devices, APIs, and online services as part of the Internet of Things (IoT). Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions. Note that it is possible to develop functions in other languages besides the default JavaScript (e.g. Python and others).

In your browser, go to <https://nodered.org/> click "Get Started".

There are options to install it locally on a device or in the cloud.

For this exercise we will install locally.

Diagram

Description automatically generated

Read and follow the instructions carefully. Pay close attention to the difference in commands between Windows, Mac, Linux, and Raspberry Pi platforms.

Now we will check Node-Red is properly installed as follows:

Open a command line window and do the following:

Type node-red and press return / enter.

This will start the node-red environment.

Your results should look something like this:

Text

Description automatically generated

Open a browser (Firefox, Safari, Chrome, Microsoft Edge)

Go to [*http://localhost:1880*](http://localhost:1880) this will open the Node-Red development palette.

Your results should look something like this:

Graphical user interface, text, application

Description automatically generated

## MQTT Broker

## *Note: this step can be skipped to run our sample, it will be required later if you want your own MQTT Broker running locally.*

## [MQTT](https://en.wikipedia.org/wiki/MQTT) is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. MQTT today is used in a wide variety of industries, such as automotive, manufacturing, telecommunications, oil and gas, etc.

[**MQTT**](https://en.wikipedia.org/wiki/MQTT) is a lightweight publish-subscribe network protocol that transports messages between devices. The protocol runs over TCP/IP and can also be configured to run over WebSockets.   
  
WebSocket is a stateful communication protocol that provides a persistent **full-duplex channel between a web server and a web client (browser)** by reusing the same connection that was opened at the beginning of the communication (during the opening handshake).

WebSocket works over a TCP connection and resides at layer 7 in the [OSI model](https://en.wikipedia.org/wiki/OSI_model).  
  
Any network protocol that provides orderly, lossless, and bidirectional connections can support MQTT.

The download can be found here <https://mosquitto.org/download/>   
  
Download the appropriate version for your platform and follow the installation instructions.   
  
Mosquitto MQTT Broker customization is required to support TCP/IP connectivity as well as WebSocket connectivity.

Locate the mosquitto.conf file in your installation. Edit this file with any available text or code editor.  
  
Search for the “Listeners” section. In this section, add:

listener 1883

listener 9001

protocol websockets

allow\_anonymous true

Search for the “Logging” section. In this section, add the following:

connection\_messages true

log\_timestamp true

We will need the above commands only to show the Mosquitto logs in the console. You may not want to touch them, but we will enable the logs for testing later.

To connect to the Mosquitto Broker using TCP/IP:

Mosquitto-server-ip@:1883 example 192.168.254.39:1883

To connect to the Mosquitto Broker using WebSockets (from a html program for example):

Mosquitto-server-ip@:9001 example 192.168.254.39:9001

## MQTTX

MQTTX your all-in-one MQTT client toolbox,

Go to <https://mqttx.app/> download and install for your platform.

This will provide you with a utility that can be used to test MQTT messaging and validate your MQTT Mosquitto Broker installation.

## Grafana

[**Grafana**](https://en.wikipedia.org/wiki/Grafana) is a cross-platform web application for open-source analysis and interactive visualization. Provides tables, graphs, and alerts for the web when connecting to supported data sources.

Go to <https://grafana.com> download and install for your platform.

This tool is not part of the facial recognition project, but you will use it later to develop your own Dashboards for IoT projects.

## Face++

Go to [*https://www.faceplusplus.com*](https://www.faceplusplus.com)

We will register to use the facial recognition service provided by Face++/

*Please note that this service is* ***free*** *for low volume use.*

Click "Sign Up"  
Graphical user interface, application

Description automatically generated

Complete the form to set up a personal use account.

Enter the verification code you receive by text message or email, enter a password, confirm that you have read the terms and conditions, and accept the Services.

A screenshot of a application

Description automatically generated

Click "Get API Key"

Graphical user interface, application

Description automatically generated

A new key will be generated for you, as well as an API secret. Click "View" and copy the API key and secret key into a document (*this is important*). Do not lose this document. We will need to use them later to configure the execution node that will invoke the face++ Facial Recognition service.

## Facial Analysis Example.

Locate the facialanalysis.zip file that you downloaded at the beginning of this document and move it to your desired location and unzip it.

A folder called “Facial Analysis” will be created and two sub-folders will be created, one called “Artifacts” and the other “Pictures for Analysis”.

Open the “Artifacts” folder and edit file *config.js* using your editor. This file contains the defaults that will be used by our Facial Analysis example.

It should look something like this:Text

Description automatically generated

*Note that line 1 is pointing the MQTT Broker to a publicly available MQTT Server on the Internet provided by hiveMQ.*

*Once you have completed the testing of the facial analysis example, you may proceed to install your own Mosquitto Broker and then you would modify line 1 to your host IP@ e.g. 192.168.254.39 then you can test again using your local Broker.*

Change line 3 topic as follows:

topic = ‘*phonenumber*/TEST/PHOTO’; replace *phonenumber* with your personal phone number which will make the topic unique to you.

Example: topic=’18135551212/TEST/PHOTO’;

*Save and close the file.*

Using your text editor, edit the *facepp.html*  file in the Artifacts folder.

Scroll down and find line 123

Graphical user interface, text, application, chat or text message

Description automatically generated

Change the following in line 123:

var mytopic = “*phonenumber*/TEST/PHOTO/RESPONSE/”;

by replacing *phonenumber* with your personal phone number which will make the topic unique to you (same number you used above)

Example: var mytopic = “*18135551212*/TEST/PHOTO/RESPONSE/”;

Do it again for line 287:

Text

Description automatically generated

*Save and close the file.*

## Configuring Node-Red

Using the browser go to <http://localhost:1880> to open the Node-Red development palette on your machine. Here's what you should see:

Graphical user interface, text, table

Description automatically generated

We need to add base64 node to the Node-Red palette which will be used by the flow that we will import.

Let's add *node-red-node-base64* support to the Node-Red palette as follows:

Click on the Hamburger icon in the upper right corner  
  
A screenshot of a phone

Description automatically generated with medium confidence

The following dialog box will open, click "Manage Palette"  
A screenshot of a computer

Description automatically generated with medium confidence

Click on the "Install" tab  
Graphical user interface, application, email

Description automatically generated  
  
Graphical user interface, application, email

Description automatically generated

Type base64 over "Search modules", the screen will show options to install. Select *node-red-node-base64* and click on the *"Install"* button This action will complete the installation of a new node calledbase64 to the Node-Red development palette.

Now we are going to install the FacialAnalysis flow in the Node-Red development palette.

Using your editor open the file named faceppFlow.txt in the Facial Analysis directory. Text

Description automatically generated  
Select the entire contents of the file with *Ctrl + A* (on Mac *Cmd + A*)  
and copy it to the clipboard with *Ctrl + C* (on Mac *Cmd + C*)

Now we will import the flow, click on the hamburger icon again in the upper right corner and the dialog box on the right will open.

A screenshot of a phone

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

Click *Import*

You will see the following dialog box, click on the pink area, and paste the contents of the clipboard with *Ctrl + V* (on Mac *Cmd + V*)

Graphical user interface, application

Description automatically generated  
Here's what you'll see, click the red *"Import"* buttonGraphical user interface, text

Description automatically generated

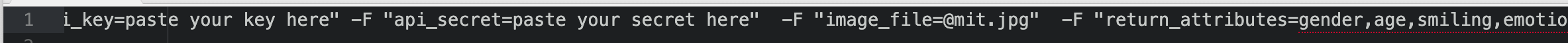
Here's what you'll see, the flow will be placed in the window   
*Note: that all of the flow nodes will have a blue circle next to them, this means they need to be deployed.*

A picture containing diagram

Description automatically generated

Click the *"Deploy"* button and the blue circles will disappear, this records the flow in the Node-Red palette, and makes it ready to execute.

Open your editor and edit the file named *CurlCommand.txt* in the Artifacts folder.

It will look something like this:   
Open the document where you saved the apikey and secretkey issued to you by face++.

Replace *apikeygoeshere* with the face++ API key.

Replace *apisecretgoeshere* with the face++ Secret key.

Save the *CurlCommand.txt* file but do not close it yet, instead do a Ctrl + A (on Mac *Cmd + A*) in the editor window to select all the contents and Ctrl + C (on Mac *Cmd + C*)  
to copy it to the clipboard.

in node-red double-click the *"curl"* node to open the configuration dialog A screenshot of a computer

Description automatically generated

Put the cursor in the *"Command"* field and press *Ctrl + V* (on Mac *Cmd +* (on Mac *Cmd + V*) to paste the content from the clipboard into the Command field (this is what we updated earlier using your editor)  
Graphical user interface, text, application

Description automatically generated

Click the *"Done"* button.

Graphical user interface, website

Description automatically generated

Click the "Deploy" button

Double click to open the MQTT In node.

A screenshot of a computer

Description automatically generated

Update the topic and Name fields replacing *phonenumber* with your phone number.

A screenshot of a computer

Description automatically generated

Double click to open the MQTT Out node.

A screenshot of a computer

Description automatically generated

Update the topic and Name fields replacing *phonenumber* with your phone number.

A screenshot of a computer

Description automatically generated

Click Done.

A screenshot of a computer

Description automatically generated

## Now we can check if it works...

Open the Artifacts folder on your file system (inside Facial Analysis folder).

Double-click facepp.html this will open the html application in your default browser.

It should look like this

Graphical user interface, application, Word

Description automatically generated  
  
Make sure the Status at the bottom says "WebSocket Connected" this means that the WebSockets connection between the html program and the MQTT Broker is open and ready.

Click the "Take a Picture" button.

Graphical user interface, application, Word

Description automatically generated

Go to the "Pictures for Analysis" folder, and select an image for analysis, double-click it. Your results will look something like this:

Graphical user interface, application

Description automatically generated

If you want to test it with photos of yourself or your family, take a photo and place it in the *“Pictures for Analysis”* folder and then run the web program again and select the image you want to process. If the image is too large, the program will tell you, make it smaller and try again.

The html program face++.html is written using the bootstrap framework, which means that it will automatically resize for a smartphone or tablet.

If your phone/tablet are on the same network try the following:

1. Open your browser on your phone/tablet enter the IP Address and press enter  
   A screenshot of a phone

   Description automatically generated
2. Click on Choose File, you will get an option to use the Photo Library, Take a Picture with the camera, or Choose a File.

A screenshot of a phone

Description automatically generated

If you have any problems, please contact one of us:

[patrick@amphoria.com](mailto:patrick@amphoria.com)

[peter@amphoria.com](mailto:peter@amphoria.com)