Connected Farms User Guide/Theory of Operations

**Group Members:**

Nathan Libner, ndlibner@ncsu.edu

Artem Minin, aaminin@ncsu.edu

Stephanie Sierra, ssierra@ncsu.edu

Manish Goud, mgoud@ncsu.edu

**Sponsoring Company:**

NCSU College of Agriculture and Life Sciences

IBM

**Mentors:**

Dr. Paula Ramos Giraldo, pramosg@ncsu.edu

Dr. Anand Singh, anand.singh@ibm.com

Dr. Natalie Hummel, nahummel@ncsu.edu

James Jushchuk, jushchuk@ibm.com

**Intended Audience:** Researchers, Plant Breeders and Farmers

**Table of Contents:**

[**Camera Setup**](#_6difmkswfsmn) **3**

[TensorFlow Install Commands](#_qh8cor9vnoug) 3

[Keras Install Commands](#_uaygepc9mj3l) 4

[IBM IoT Install Commands](#_4ht7388lc2wc) 4

[Testing IoT.py w/ Test Images](#_509fb3t44co6) 4

[Increasing RAM:](#_40285y7y1rv6) 7

[IoT Log Files](#_9djomiqrqica) 7

[**IBM Cloud Setup/Use**](#_6259a7bqj1vs) **9**

[IBM Watson IoT Platform](#_5g5w0vjc6486) 9

[Connecting the device to the IBM Watson IoT platform](#_m7uqfwv1gthf) 10

[Node-Red](#_kgqn4na74poz) 11

[DB2 Database](#_pk7da7ciobpc) 14

[Cognos Embedded Dashboard](#_nlxp37ptqfr7) 16

[**Web Platform Setup/Use**](#_g24n32sq0wtc) **16**

[Create Account/Login](#_u2dc95xezsnl) 16

[Plots Tab](#_5z5qc5cwove) 17

[Graphs Tab](#_hv44pik6hvv4) 19

[Historical Data Tab](#_3jrpj52j8yx3) 20

[Device Commands](#_9dh0gk3czotj) 21

# 

# **Camera Setup**

The manual setup guide for the Raspberry i is described in detail below. However, the recommended method for provisioning and imaging a Raspberry Pi Camera is flashing the SD card with a copy of an already working image. That process is described in “Building and Setup Guide”. This process is mentioned for potential future work and adaptation.

## TensorFlow Install Commands

To run the machine learning algorithm on a Raspberry Pi and rate the water stress level of captured images, certain libraries need to be installed for the model to correctly run. Specifically, this means that Tensorflow, Keras, and other accessory libraries are necessary to download.

Instructions followed and adapted from:<https://www.tensorflow.org/install/pip>

Dependencies: Python 3.7, pip3

To check the versions, type the following commands:

python3 --version

pip3 --version

Tensorflow requires **Python > 3.4 and pip >= 19.0.** If those versions are not the most current or are not present type the following commands:

sudo apt update

sudo apt install python3-dev python3-pip

sudo -H pip3 install --upgrade pip

Now to install Tensorflow on the raspberry pi:

pip3 install --no-cache-dir --upgrade tensorflow

If this command gives a “MemoryError” jump to the “Increasing RAM” section down below.

## Keras Install Commands

Instructions gathered and adapted from:

<https://medium.com/@paroskwan/layman-installation-guide-for-keras-and-tensorflow-on-rpi-3-38b84f3e59dc>

sudo apt-get install libblas-dev

sudo apt-get install liblapack-dev

sudo apt-get install libatlas-base-dev

sudo apt-get install gfortran

sudo apt-get install python3-dev

sudo apt-get install python3-setuptools

sudo apt-get install python3-scipy

sudo apt-get update

sudo apt-get install python3-skimage

sudo apt-get install python-imaging-tk

pip3 install scikit-image

pip3 install keras

## IBM IoT Install Commands

For the Raspberry Pi to connect to IBM’s IoT platform, and receive commands from the website, the following library is needed.

sudo pip3 install wiotp-sdk

## Testing IoT.py w/ Test Images

Import the following files into the home directory of the Raspberry Pi:

· **IoT.py**

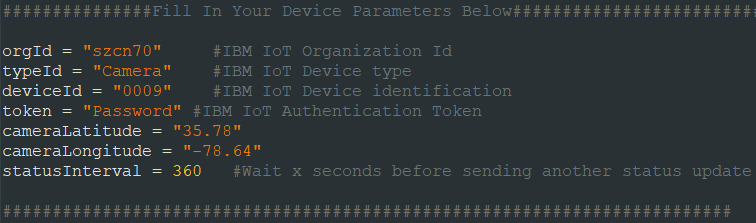
· **converted\_model.tflite**

Then, create a directory called **images** in the home directory, this is where captured images will be stored.

mkdir images

This program, IoT.py, can be easily altered to send the captured photo to the machine learning algorithm, by uncommenting line 266. However, at the moment it analyzes and rates test images. If your SD card image does not have any Soybean testing images in the “Pictures” directory, they will need to be added.

Open the IoT.py file and modify the parameters below to match the parameters created by you on the IBM IoT platform for your particular camera. Typically orgId, typeId, and the authentication token will be the same for all devices. This is also where the camera location can be specified as a latitude and longitude; this location will be displayed as a map on the website. Status interval is the time between data transmissions to the database.



Then type the following command. This program connects your camera to the IoT platform which allows data and commands to flow between the website and the Raspberry Pi. This program also captures images and sensor data, receives and executes user-sent commands, and performs edge computing by analyzing and rating captured images utilizing a TensorFlow lite model:

python3 IoT.py

Our program will load the Keras model and analyze and assign a water stress level rating to an image in the “Pictures” directory. If the program runs completely and the following is outputted, you have correctly installed the dependencies. This program will repeat every [statusInterval] minutes and can receive commands from the website anytime within hours scheduled on the WittyPi. The array of values printed on the output corresponds to the machine learning model's confidence as a probability[array values] for each of the water stress levels[array indices]. So the value: [0.11378021 0.651998 0.20384002 0.02714281 0.00323909] means that the trained model is 11.37% confident the soybeans in the image: 001405.jpg are a water stress level of 0(not dehydrated). 65.2% confident that the soybeans in the image are a water stress level of 1(slightly dehydrated) and so on for the rest of the values of the array. The program then selects the most confident rating which gets recorded and printed, in this case it was most confident that the image was a water stress level of 1. Note it may take up to 3 minutes for the program to run and evaluate the test image.

2020-04-27 07:02:32.320393: E tensorflow/core/platform/hadoop/hadoop\_file\_system.cc:132] HadoopFileSystem load error: libhdfs.so: cannot open shared object file: No such file or directory

2020-04-27 07:03:37,703 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:szcn70:Camera:0001(Press Ctrl+C to disconnect)

taking Image

resizing image

allocating tensors

invoke interpreter

[0.11378021 0.651998 0.20384002 0.02714281 0.00323909]

001405.jpg

Water Stress Level 1

Percent Confident 65.1997983455658

Sensor Reading

Sending Data

## Increasing RAM:

If IoT.py does not run correctly, and a “bad\_alloc” error is given, the following commands will be necessary. These commands increase the RAM of the Pi Zero by allocating part of the SD card as RAM swapped memory. If the RAM is getting fully utilized, consider using these commands as well

sudo dphys-swapfile swapoff

sudo nano /etc/dphys-swapfile

CONF\_SWAPSIZE=1024

sudo dphys-swapfile setup

sudo dphys-swapfile swapon

## IoT Log Files

IoT.py stores data on the Raspberry Pi as well as sending it to the cloud databases(DB2, Cloudant). These log files are located in the /home/pi directory and are named scriptOutput.txt, waterStressLog.txt, and data.txt. scriptOutput saves the console output of IoT.py, and can be used to detect if there are any problems while the script is being run automatically by the wittyPi. waterStressLog is where the image name, the associated water stress level prediction, and the percent confidence. data.txt is where the sensor data is logged.

# 

# **IBM Cloud Setup/Use**

The IBM Cloud is a platform that offers many tools and services. Before you can start using the IBM Cloud, you’ll need the following:

* [IBM Cloud Account](https://cloud.ibm.com/registration)

Once the account is created you can begin to create the resources pertaining to this project. The following services and applications are needed:

* Internet of Things Platform
* DB2 Database
* NodeRed
* Cognos Embedded Dashboard

To create an instance of these services, click on create resource on the IBM Cloud dashboard and configure each resource individually. Once they’re created, they will appear under resource summary on the IBM dashboard.

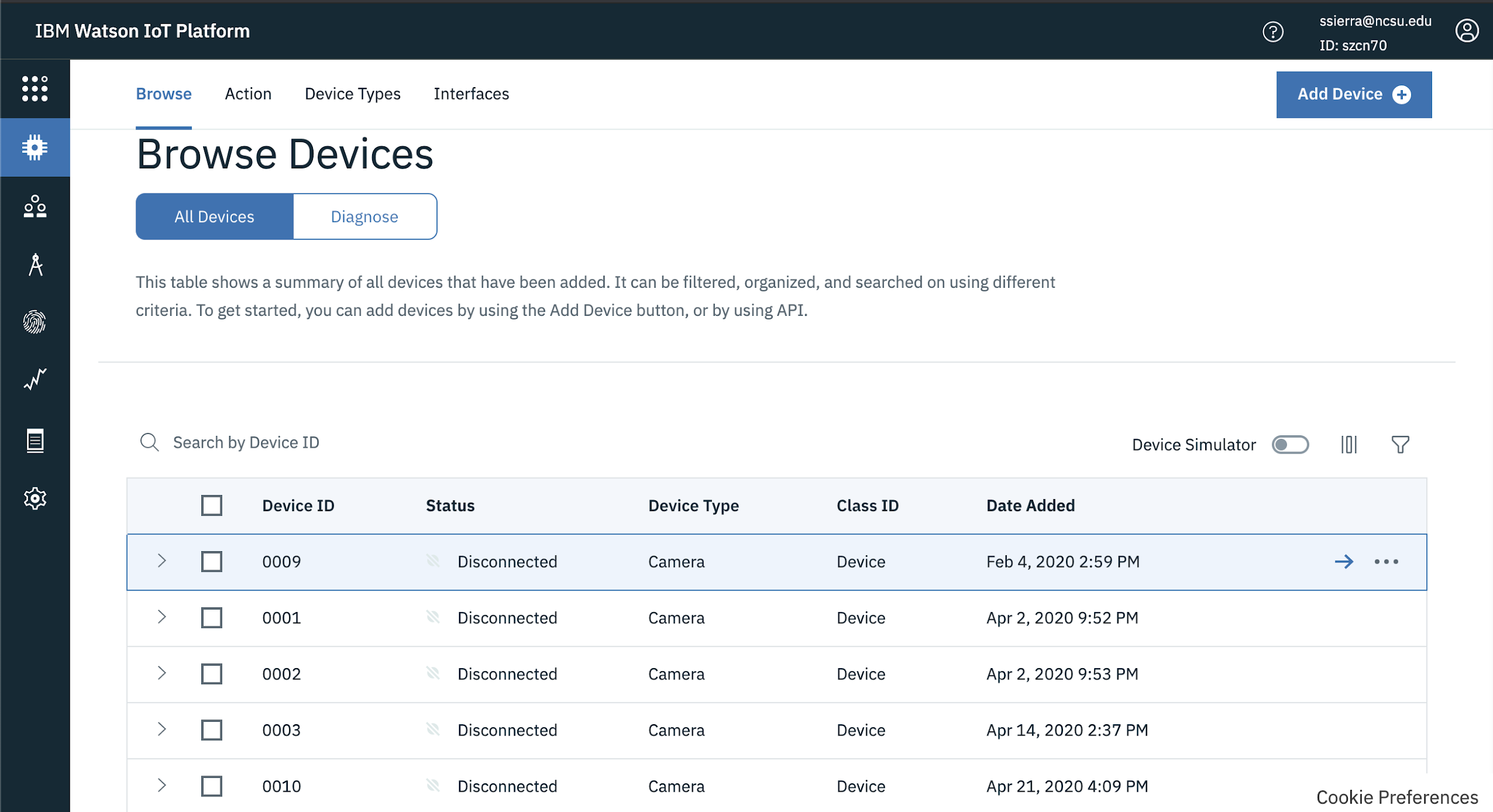
### IBM Watson IoT Platform

The Watson IoT platform is a service that allows users to connect and manage IoT devices, and analyze the data they produce. Before you can start using the IoT Platform, you must have the following items:

* [IBM Cloud Account](https://cloud.ibm.com/registration)
* An instance of Watson IoT
* A device (Raspberry Pi)

#### Connecting the device to the IBM Watson IoT platform

Instructions adapted from [IBM Getting Started](https://cloud.ibm.com/docs/services/IoT?topic=iot-platform-getting-started)



Once the platform is launched you will be able to register a device by creating the credentials needed in order to link the platform and device together.

In the overview dashboard, select devices and then click add device. Select camera as the device type and enter a unique device ID.

Under the security tab, the authentication code can be either auto-generated, or you can provide your own authentication token for the device. Please note that any lost authentication tokens cannot be recovered.

Click Finish.

In the device information page copy and save the following details:

* Organization ID
* Device Type
* Device ID
* Authentication Method
* Authentication Token

This information will be used in the script to configure the cameras.

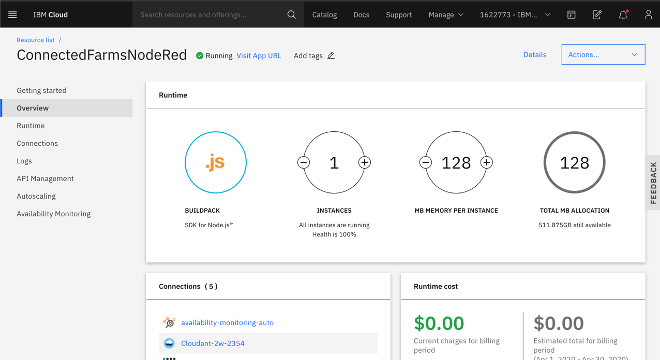
### Node-Red

Node-Red is a graphical programming tool in the IBM Cloud suite that controls the flow of data and information anywhere from hardware devices to other software programming tools. We use Node-Red to connect the cameras to the web platform.

Before you can start using Node-Red, you must have the following items:

* [IBM Cloud Account](https://cloud.ibm.com/registration)
* An instance of Node-Red
* Create a connection between Node-Red and other applications that you’d like to control the flow of data:
  + IoT Watson
  + Db2 Database

On the IBM Cloud launch the Node-Red application by clicking on the application under the resource tab and clicking Visit App URL

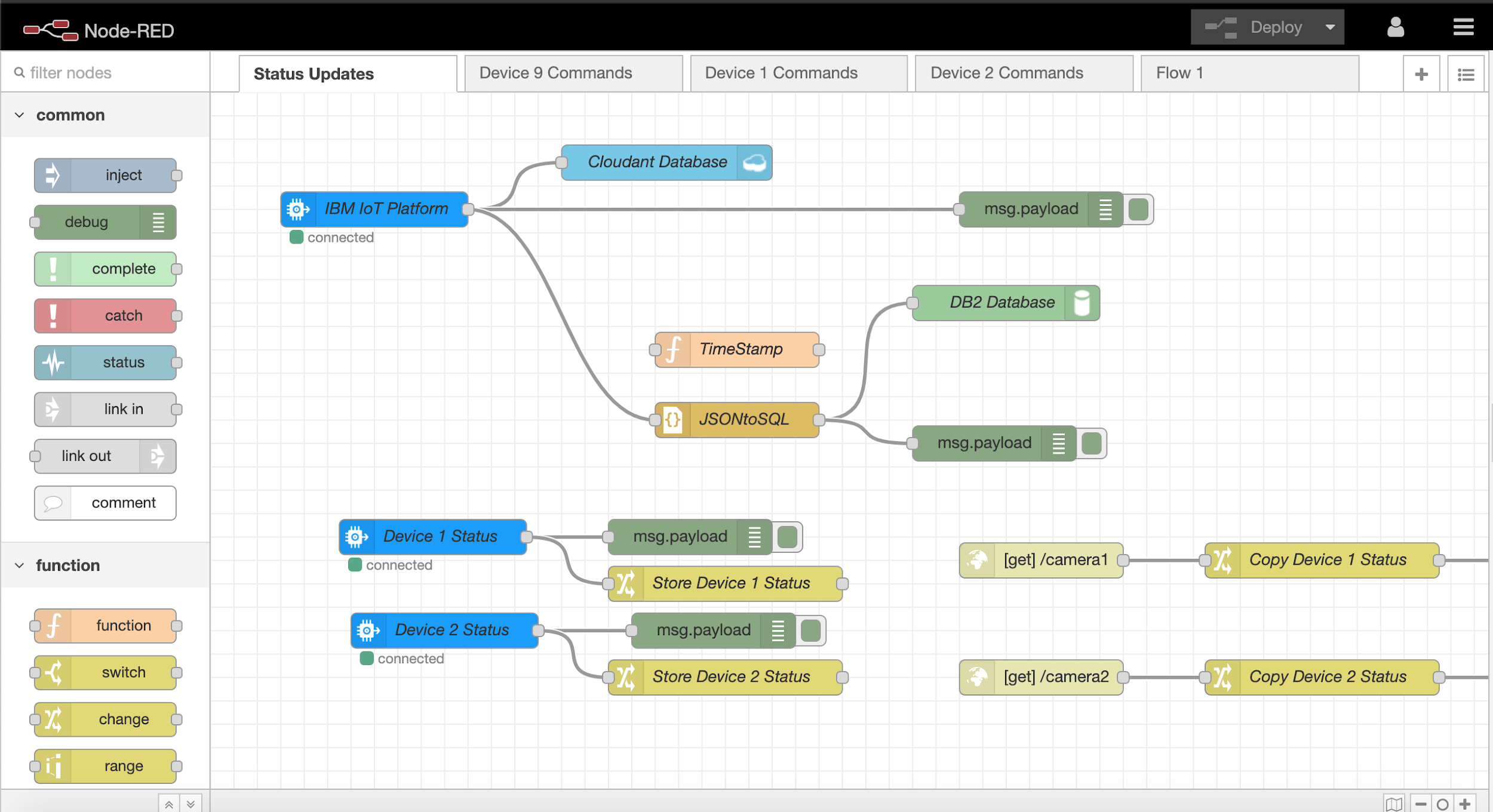


Once the application is launched enter the login credentials access the flows for this project.

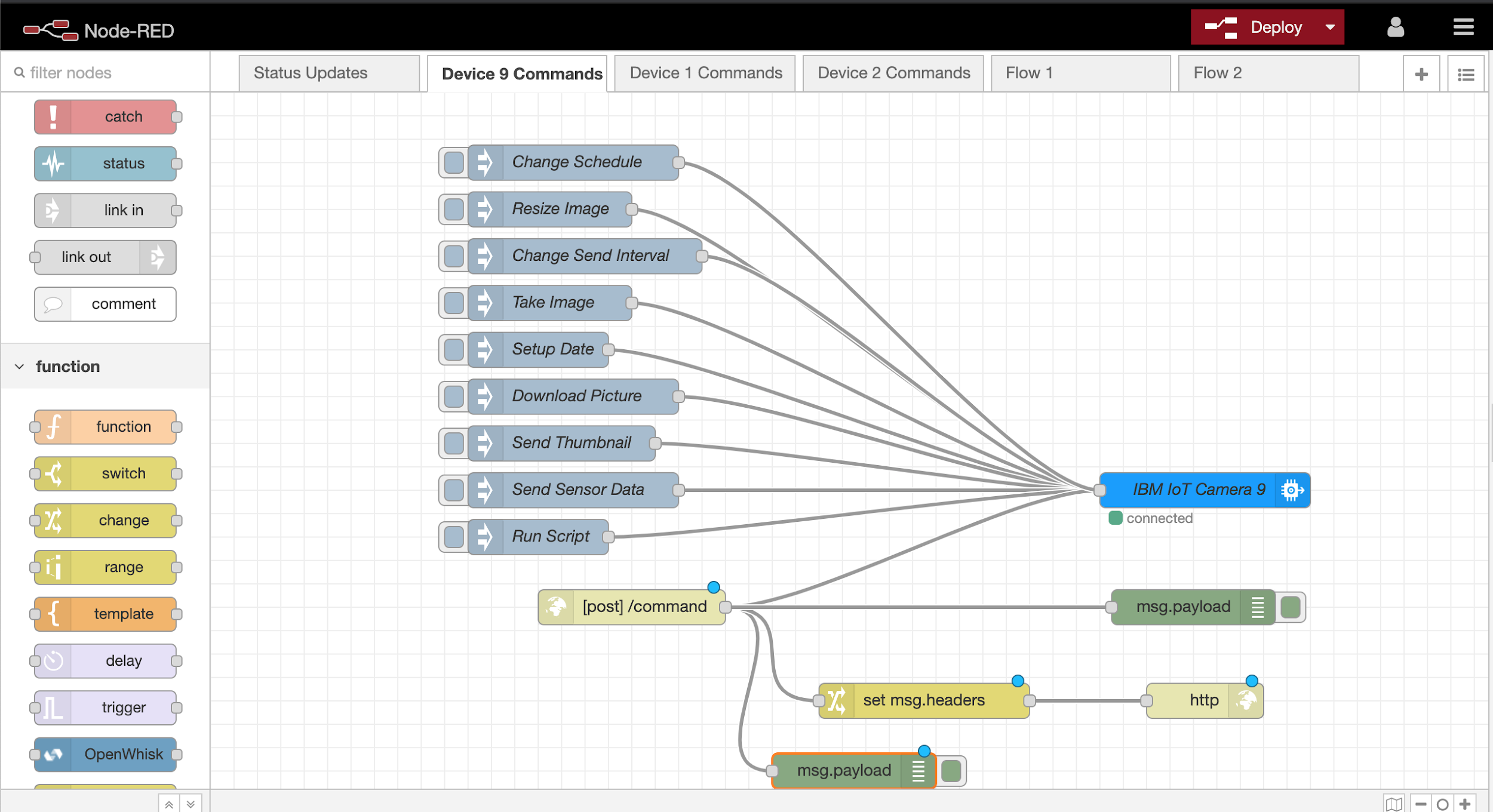
On the left are all the nodes needed to create a flow.

The Status Updates flow is responsible for transmitting data from the IBM IoT platform to the DB2 Database. Since the DB2 Database can only store data in SQL the data must first go through the json parser, which converts between a JSON string and JavaScript object representation.

The Device 1 and Device 2 Status nodes establish a connection with the website to notify the user whether or not the cameras are connected or disconnected.



The Device Commands flow is responsible for sending commands to the camera from the website. The inject nodes on the left are used for debugging purposes. You can manually send a command to the camera by clicking one of those nodes and you will be able to view the output from the camera on the debug widow on the right side of the screen.



### DB2 Database

The DB2 database is used to store data from the device sensors. It is also used to retrieve information for the visual graphical representation on the website.

Once the database is created, you may start to create tables with the attributes and attribute types for the data that needs to be stored.

To create a table, you must navigate to Run SQL and then you must choose the way you’d like to create a table. You may start from blank or you can start from a file. The [DB2 Create Table guide](https://www.db2tutorial.com/db2-basics/db2-create-table/) was used to learn more about the structure of tables.

Below is the syntax that is used to create the tables needed to store the information from the cameras and sensors :

|  |  |  |  |
| --- | --- | --- | --- |
| CREATE TABLE CAMERA (  DEVICE\_ID VARCHAR(4) ,  DEVICE\_STATUS VARCHAR(32) ,  LATITUDE DECIMAL(10,0) ,  LONGITUDE DECIMAL(10,0) ,  WATER\_STRESS\_LEVEL VARCHAR(32) ,  AIR\_TEMPERTATURE VARCHAR(32) ,  WITTYPI\_TEMPERATURE VARCHAR(32) ,  CPU\_TEMPERATURE VARCHAR(32) ,  LUXOMETER VARCHAR(32) ,  DATE\_AND\_TIME CHAR(5)  ) | | | |

To ensure that the table was correctly created you can navigate to the explore page and select tables. If you see the name of your table then it was correctly created and you may use it in Node-Red.



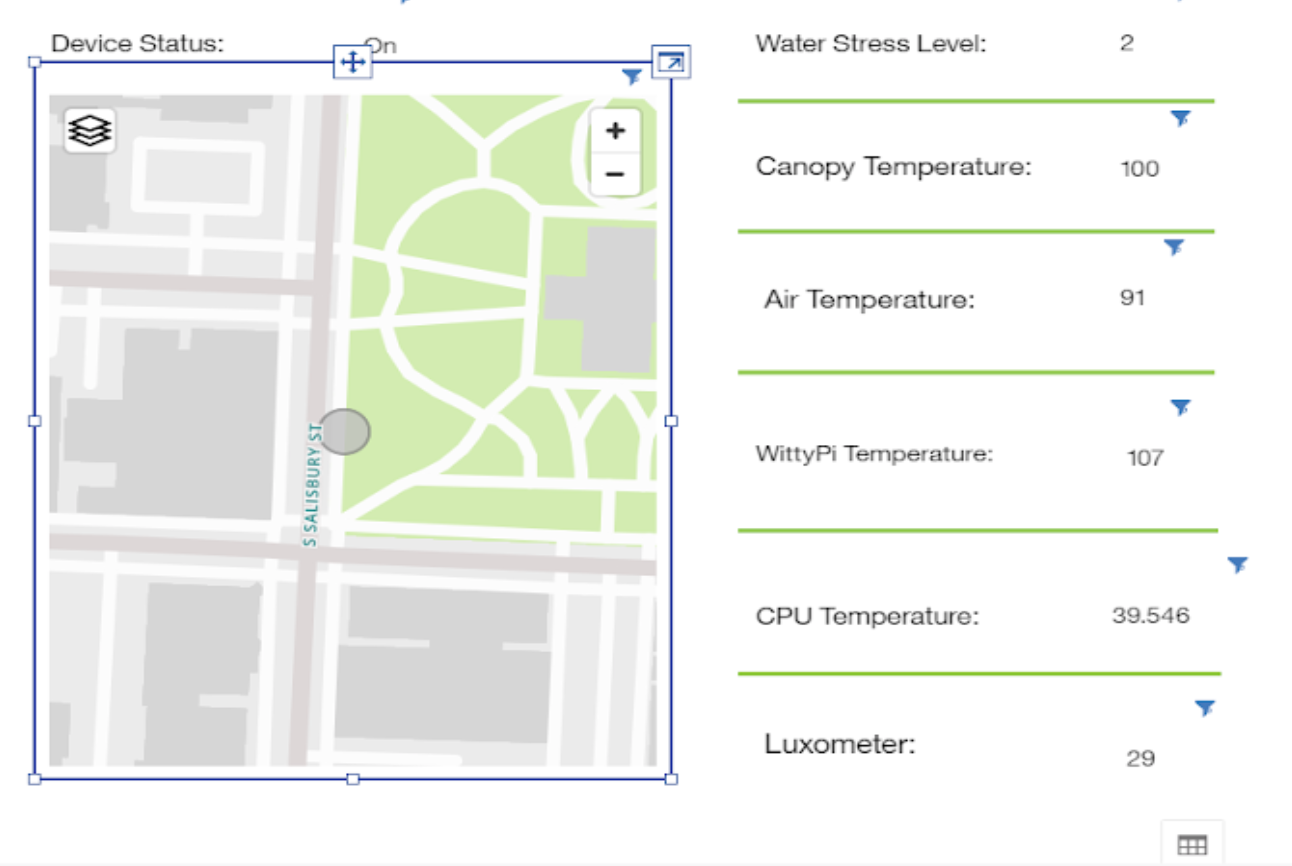
# 

### Cognos Embedded Dashboard

Cognos is used to create graphical representation of data. Before you can start using Cognos, you must have the following items:

* [IBM Cloud Account](https://cloud.ibm.com/registration)
* An instance of Node-Red
* A cognos session
* DB2 Database with data

Cognos must be connected with Node-Red in order to retrieve data from the database. Once it’s connected navigate to Create New, click crosstab and then ok. On this page you’ll be able to request data from the DB2 database and select the chart that you’d like to create to represent the data by selecting the attributes. For example the locations of the cameras can be displayed by using the longitude and latitude attributes.



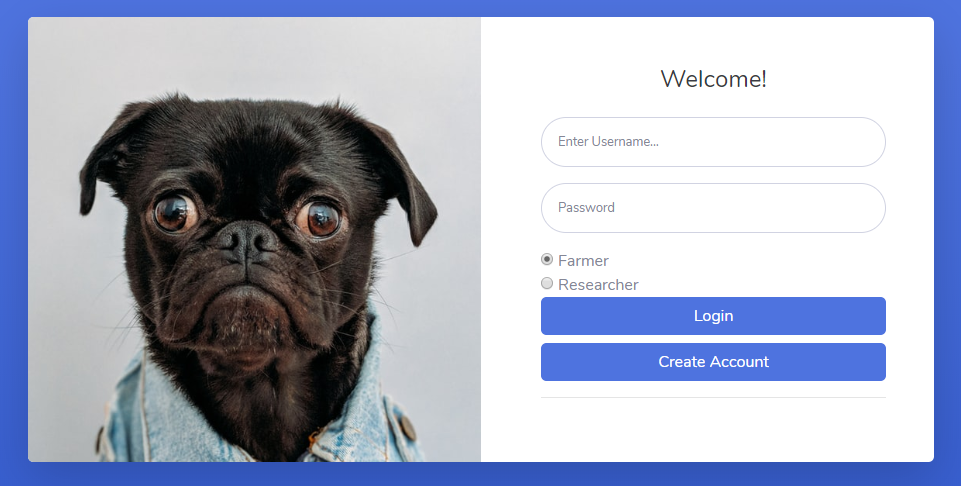
# **Web Platform Setup/Use**

The web platform was coded in HTML and uses Javascript and Python scripts for some functionalities such as authenticating a Cognos session.

## Create Account/Login

To access the web platform, the first thing you will need to do is head to the Connected Farms Web link, listed below, and create an account.

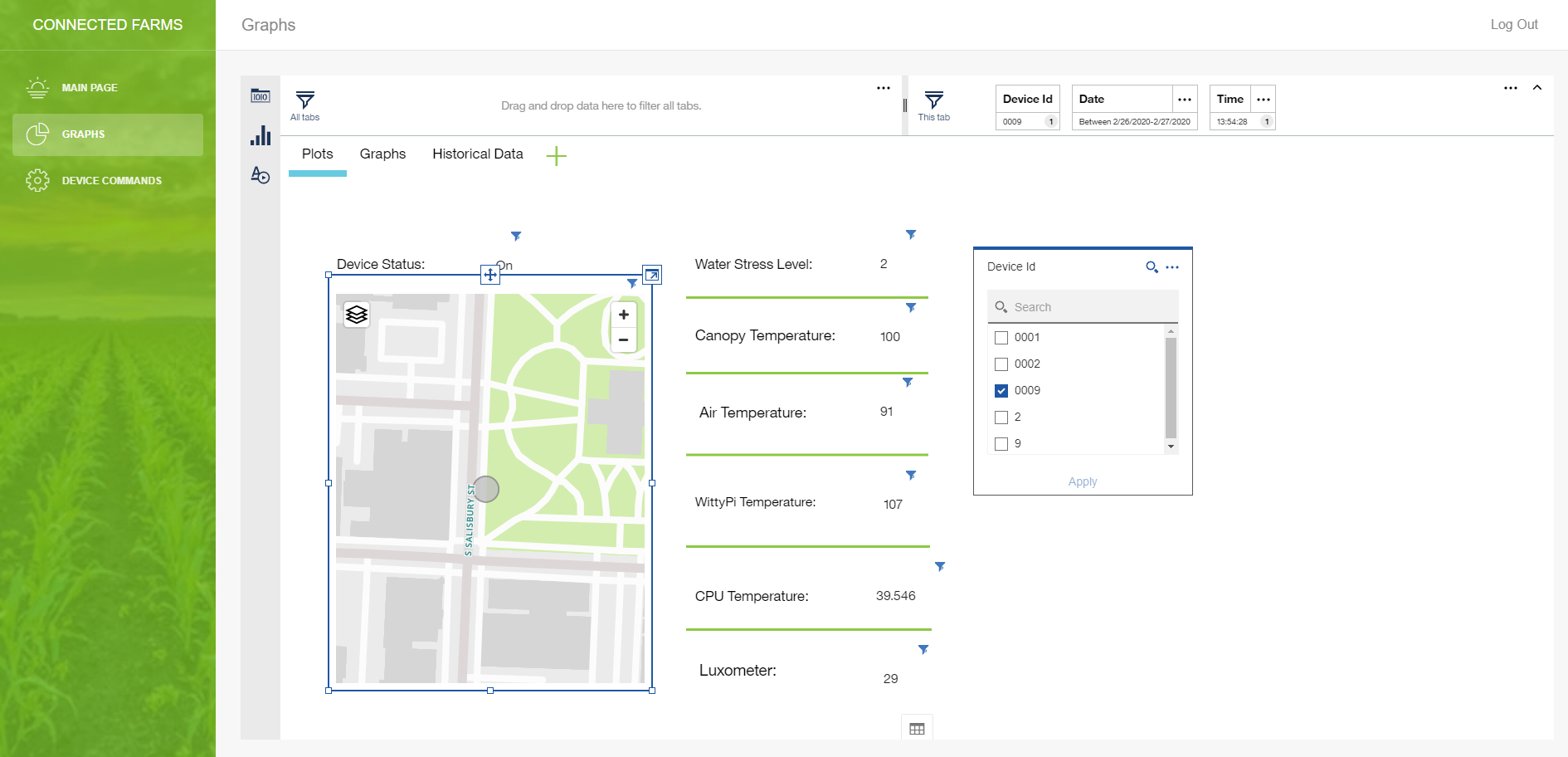
[https://connectedfarmsweb.mybluemix.net/](https://connectedfarmsweb.mybluemix.net/index.html)

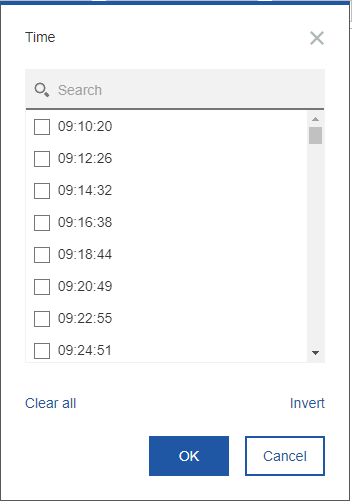
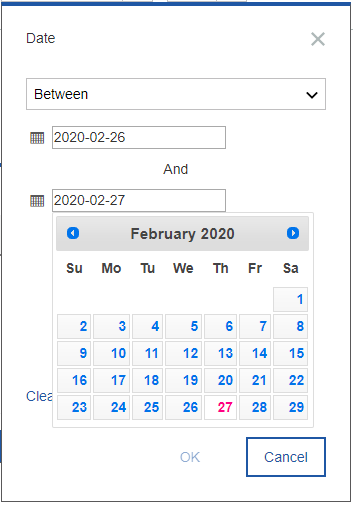


In order to create a new account you must type in your desired username and password and then select either farmer or plant breeder. Then hit the “Create Account” button and your credentials will be registered in the system.

## Plots Tab

When you have created your account, the graphs page of the web platform will load in and default to the “Plots” tab. This is where you will view all of the data coming in from the camera.

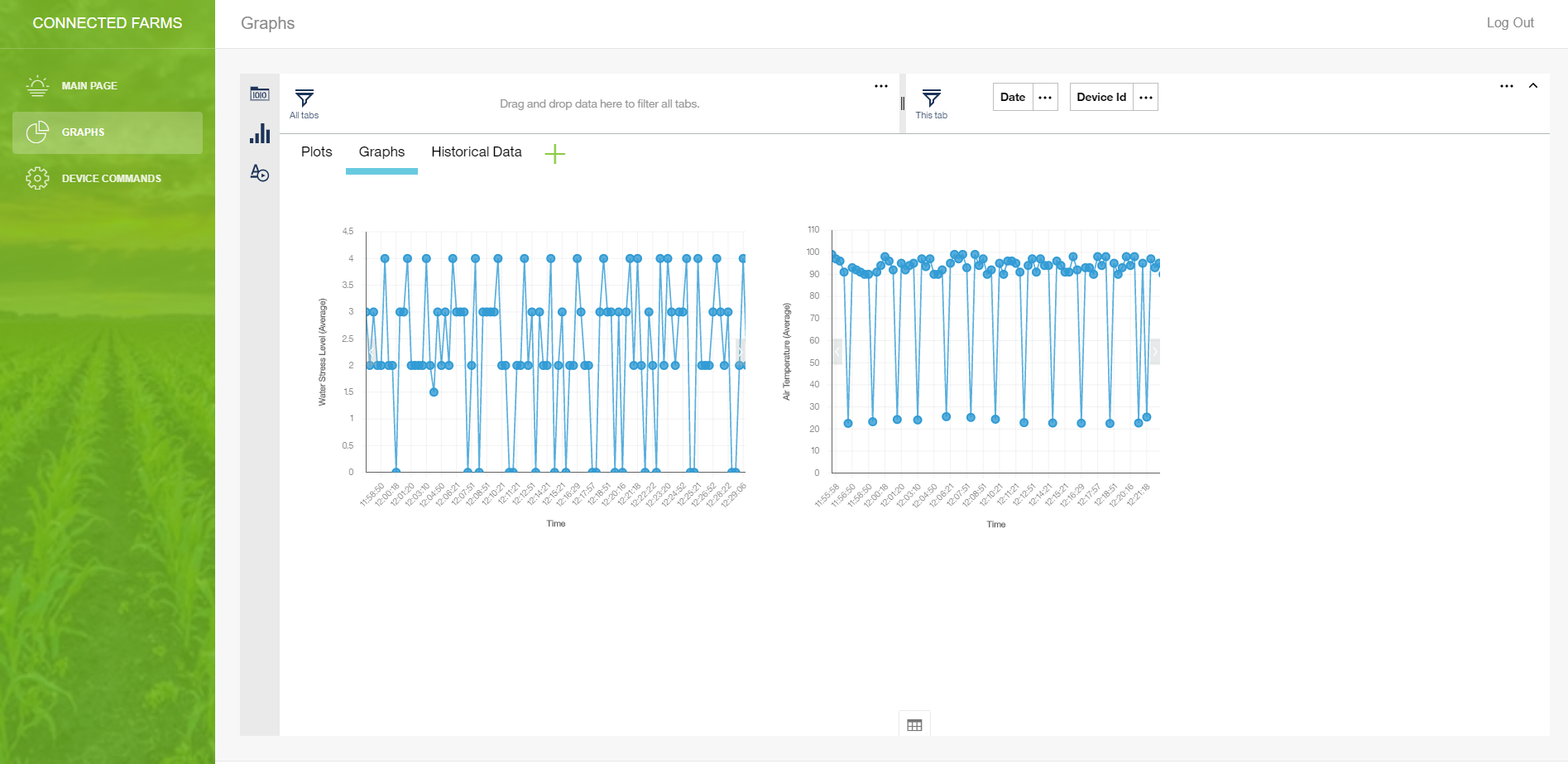
On the right side of the screen, you will see a box labelled ‘Device Id” where you can select the camera that will display its data. Then, at the top right corner of the screen, you will see two boxes labelled “Date” and “Time.” 



In the “Date” tab, to select a specific day, you set the top dropdown to say between, as shown above, and set the first day to be the day you want to view, and the second day to be the next day. Then you will go to the “Time” tab and select the time of day that you would like to view the data (note: the website saves every time value ever stored here as an option, so some will not be valid) .

## Graphs Tab

To view daily graphs, click on the “Graphs” tab.

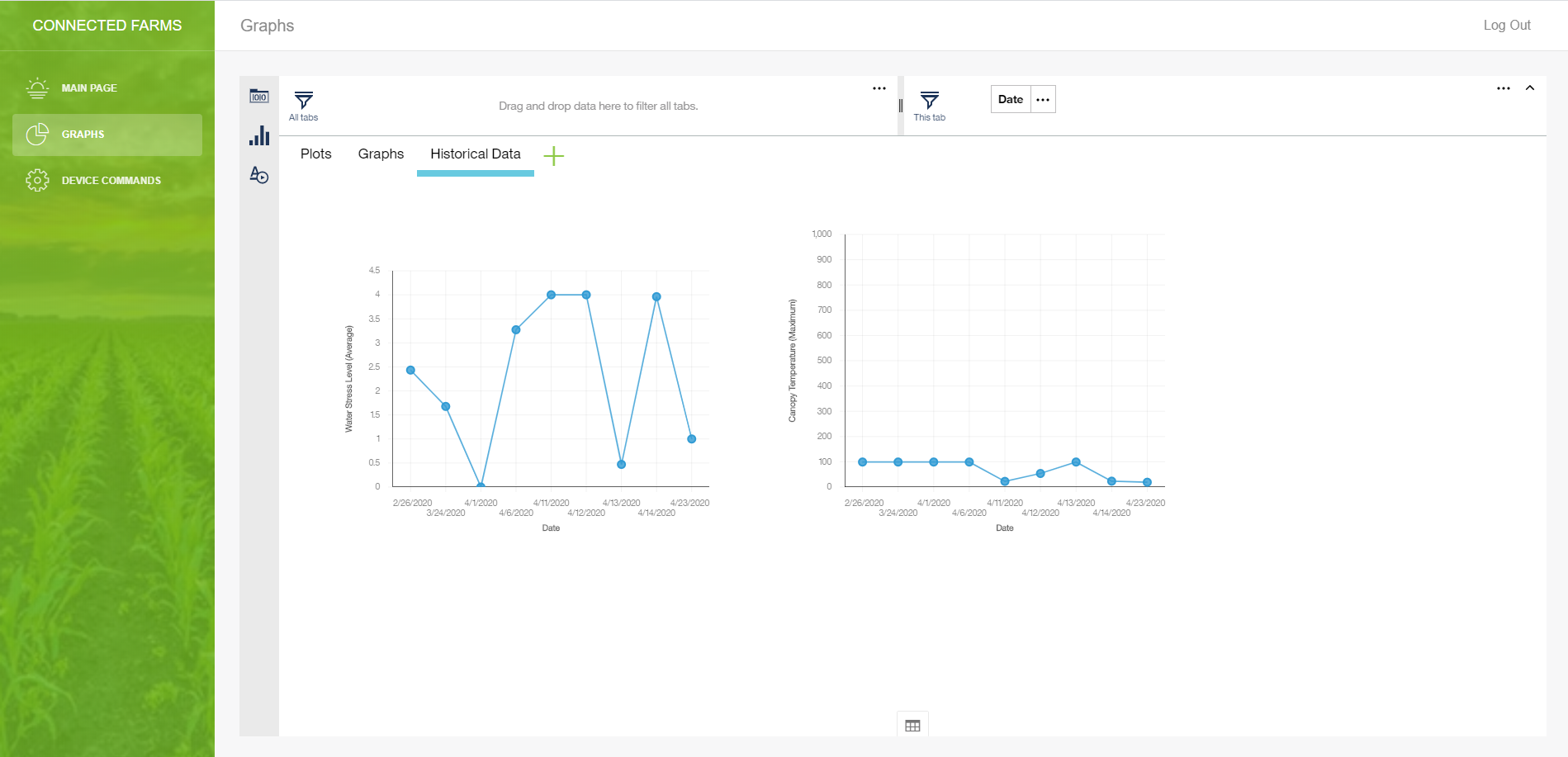


On this tab you will have to click on the tab at the top right corner and select the current date the same way that you did on the previous page and then select the “Device Id” tab and choose a camera. The graphs will display the data over the course of the day for the chosen camera.

## 

## Historical Data Tab

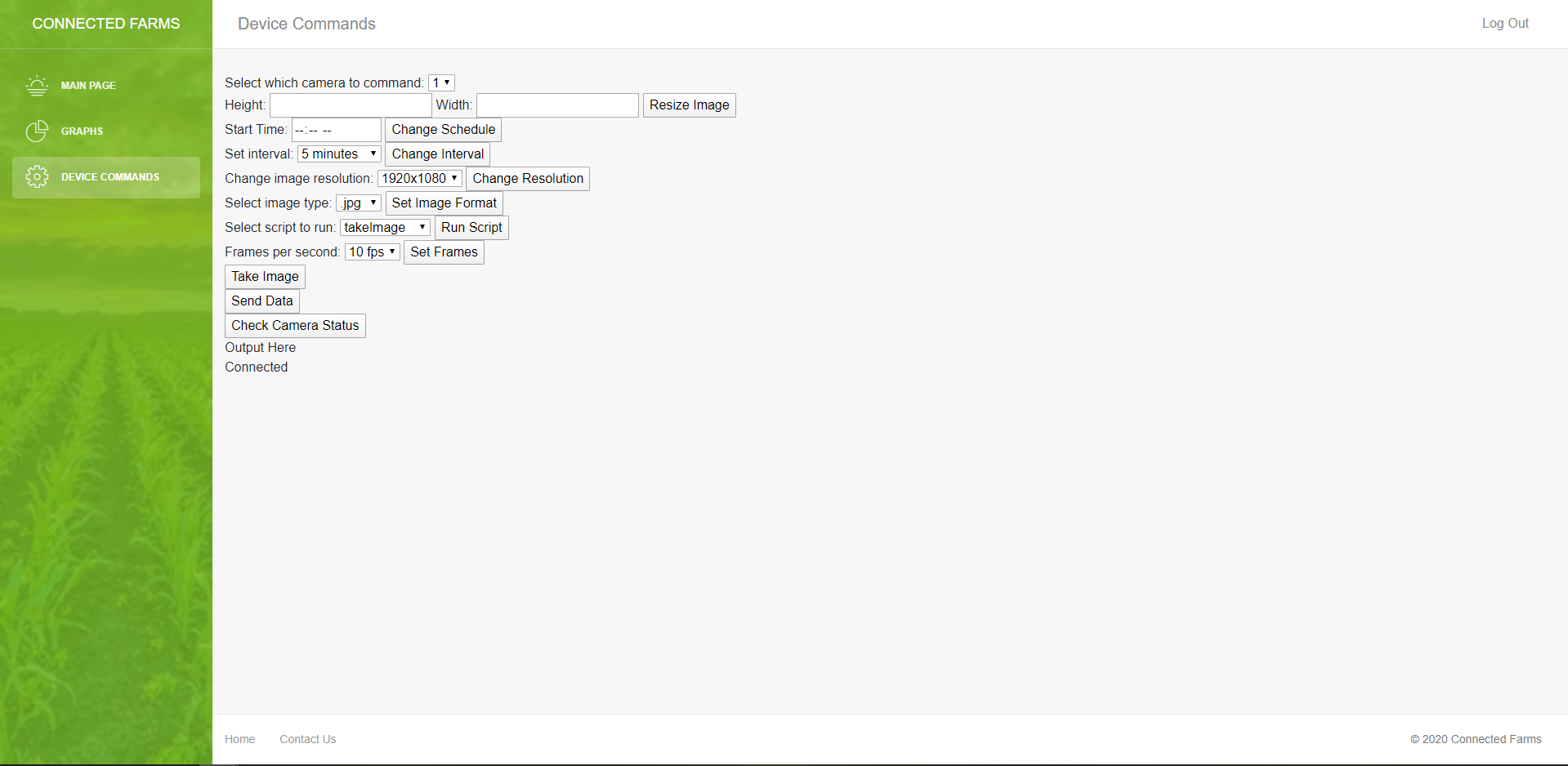
The final tab on the graphs page is the “Historical Data” tab.



On this tab, you can click the “Date” tab in the top right corner and select the range of dates that the graphs will show. The graphs will display the average value of the parameter over the course of each day.

## Device Commands

To control the cameras, you can click on the “Device Commands” button on the left side of the screen which will take you to the screen shown below.



You can choose from the list of commands and enter in the desired parameters where necessary. When you load this page, the status of the camera selected will be shown on the very last line of the screen. If you would like to refresh and check the camera’s connectivity, you can click on “Check Camera Status.”

When you are finished using the web platform, you can press the “Log Out” button located at the top right corner of the screen.