

# CCI Server

Find out how to connect to the CCI Server [cci.arts.ac.uk](http://cci.arts.ac.uk).

- [How to access the CCI air quality data](#)
- [How to connect to the CCI server](#)
- [Using CCI's MQTT endpoint](#)
- [How to Connect to MySQL Database on the CCI Server](#)

# How to access the CCI air quality data

We have several AirGradient indoor and outdoor air quality sensors at the Greencoat building, which log data every 30 seconds via MQTT and are available to students and staff who want to access this data.

## MQTT Server

The MQTT server sits at `mqtt.cci.arts.ac.uk` on port `1883`.

The read-only student username and password can be provided by asking on Slack in the `#technical` channel.

## Data format

The data format is `airgradient/readings/(sensor)`, the sensors is the MAC address and serial number of that location, the locations are:

### Greencoat

- GB\_G03 = 0cb815082660
- GB\_G04 = 4022d8f9b4d8

### Peckham Road

- PR\_B501-01 = dc5475bb845c
- PR\_B501-02 = b48a0a613900
- PR\_B501-03 = dc5475bcc430

### High Holborn

- HH\_302 = dc5475bce770
- HH\_308 = dc5475bacb84

### Millbank (Chelsea College of Arts)

- MB\_AG06 - Lecture Theatre
  1. a842e3285cf8
  2. 64b70835462c (tbc)
  3. c049ef0c92fc
- MB\_ALG02 - Interior Design Studio
  1. c049ef0c9514

2. 90380c6dc6c8
  3. a842e3283e84
  4. a842e3285cf0
- MB\_BLG19B = c049ef0c8648
  - MB\_BLG19C = 0cb815081120
  - MB\_BLG19D = 64b70834d3b0
  - MB\_BLG20 = 64b70834fa14
  - MB\_BLG20A = 64b7083502c8

Example data:

```
{
  "firmware":"9.3.0",
  "wifi":-39,
  "ssid":"UAL-IoT",
  "light":4095,
  "hwVersion":8,
  "rco2":409,
  "atmp":31.2,
  "rhum":31.55,
  "tvoc_index":164,
  "nox_index":1,
  "pm003_count":544,
  "pm01":2,
  "pm02":2,
  "pm10":2,
  "boot":1537,
  "wdog":1
}
```

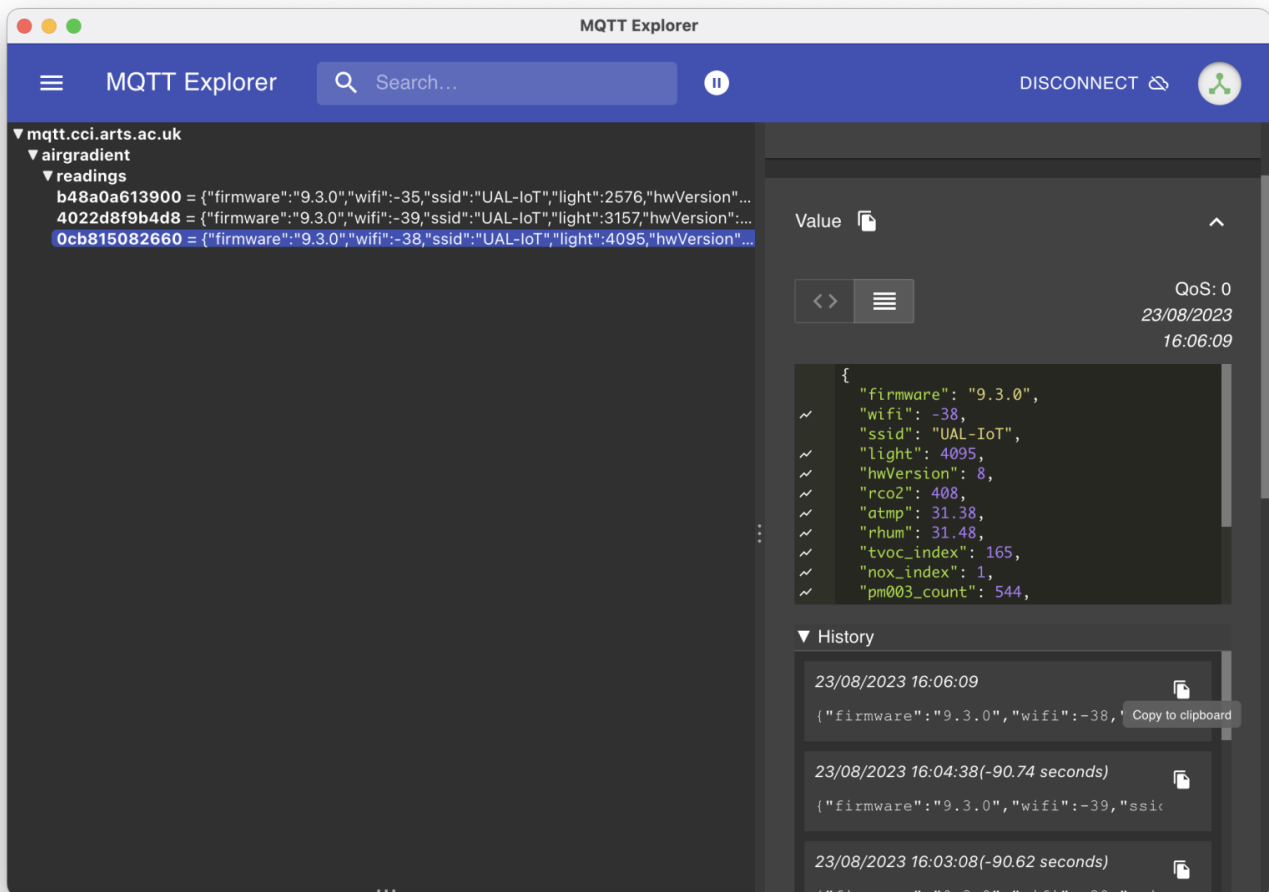
## What are these values?

- pm003\_count: PM0.3 particle count
- pm01: PM1 µg/m3
- pm02: PM2.5 µg/m3
- pm10: PM10 µg/m3
- nox\_index: [NOx Index \(Ind41\)](#)
- tvoc\_index: [TVOC Index \(Ind40\)](#)
- rhum: Relative Humidity %
- atmp: Temperature °C
- rco2: CO2 PPM

# Subscribing to the data

In order to view the data in something like a GUI you need to subscribe with a wildcard such as `airgradient/#` the default subscription for apps like MQTT Explorer shown below is just `#` however the permissions of the student account only allow you to view the data inside `airgradient`, `sensors`.

If you wanted just 1 sensor, you could subscribe to `airgradient/sensors/(sensor)` where sensor is one of the sensors listed in the sections above.



# How to connect to the CCI server

We run the CCI server as a resource to host websites and projects for students and staff at CCI, the server currently hosts:

- Apache 2 (Web Server)
- PHP
- Node.js
- Python
- MySQL
- MQTT

For security reasons access to the server is via SSH (Secure SHell) using key authentication. To connect to the server you'll need to:

1. Request access via Slack #technical for a username and password
2. Create a public/private key pair on your computer
3. Upload the public key to the server
4. Login using SSH or SFTP

Creating an SSH key requires some basic SSH commands on your computer, it's also really important that you keep the private key secret and that you revoke it immediately if you ever loose it or suspect illegal access to your computer.

## Check for an existing key

1. Open the Terminal / Command Prompt
2. View the contents of the `.ssh` folder in your home directory

macOS/\*nix: `ls .ssh` and press enter Windows: `dir .ssh` and press enter

3. Check for 2 files `id_rsa` and `id_rsa.pub`

If these files exist then you have existing keys, otherwise follow the instructions to create a key

## Create a key

1. Open the Terminal / Command Prompt
2. Type `ssh-keygen` and press enter. The computer will hold on `Generating public/private rsa key pair.` for a short time.
3. Press enter when you see `Enter file in which to save the key (/Users/username/.ssh/id_rsa):`
4. Press enter when you see `Enter passphrase (empty for no passphrase):` unless you want to be prompted for a password when you connect to the server, in which case you should type this now (when typing passwords in the Terminal it will not be visible).

5. When you see `Enter same passphrase again:` either retype the passphrase or press enter for no password.
6. You'll see a number of lines output:

```
Your identification has been saved in id_rsa.  
Your public key has been saved in id_rsa.pub.  
The key fingerprint is:  
SHA256:wS+TjYgAeQftejsnZOE4b7SQJ+iRAPAd6FwA1/ykCcY username@computername  
The key's randomart image is:  
+---[RSA 3072]----+  
|==++B.          |  
|o+E.*...        |  
|.o=.o=.o o      |  
|.o.=... *       |  
|...=... S o     |  
|.oB B  o        |  
|. .@ o          |  
|. B .           |  
|. . +          |  
+----[SHA256]-----+
```

After this is complete you should check the key using the previous instructions.

## Upload the key

1. Open the Terminal / Command Prompt
2. Copy the contents of `id_rsa.pub`

macOS: Type `pbcopy < ~/.ssh/id_rsa.pub` and press enter, this will copy the key to your clipboard.

Windows/Linux users go to your file browser and find the folder `.ssh` in your home directory and then open `id_rsa.pub` in your plain text editor and copy the text.

### Tip

Your username is the one for the CCI server, not your UAL login.

3. Open a web browser and login using your username and password a <https://cci.arts.ac.uk:20000>
4. Select SSH Configuration
5. On your first login you need to select Key Type as `RSA (for SSH v2)` and then press `Setup SSH Keys`
6. Select "Authorised Keys"

7. Click "Add a new SSH 2 authorised key"
8. Type a key name as one word i.e. "MyKey"
9. Set the type to "RSA"
10. Paste the key into the large text box, and remove the suffix ( `ssh-rsa` ) and prefix ( `username@computer` ) as highlighted...

#### “ ssh-rsa

```
AAAAB3NzaC1yc2EAAAADAQABAAQGCvYsTJEu5YabS5sXnquAh7Cxn+UK9Rmc
f2iM+qsObyOp4VWcSCGHUHgsgUPnSeOijtwgRDGQI8M7ZcXKuUFSOV2FCHTFzTi
G7F/lrFWt/PpgPUajCF3WXmH19baCBO6ymTV3wUWXg7xHsSc83k+cb9af5QT+ml
315vqSF45oO935DadaOJnKTZkMHLBydmIVSkxDFWHLkNURw7jmDNwZ2yNCru6y
2iamgdQFwGxoNIVewhG+k2jUKURN7JeGP/4dOD8u+ajeY57wp3+GxQKcp/JvtDhF
Ux92TO+1SIYk1ZcKilxActc3L6j2U3TIk3LOCRyBmwPNeeHuhLgD4ePsR4PsHz5zdO
smTLvjfM4n6B5K+Y52Re+gOE3dG5gjkvY/bhQdKGes1hBP2p3NW8/vjdZ08/auCs
wEFWoCnDKT9h8OQUTEjtywDzius6LYIHZFSLTnedEf54CfePzVcDNhdSuKc2SD6df
gE+Jk/zzqpBNmLJwCb0jxrWU0xyQDGbQFk= username@computer
```

11. Press Create

## Verify SSH key

### Tip

Remember to replace username with your CCI server username, not your UAL username!

1. Open the Terminal / Command Prompt
2. Type `ssh username@cci.arts.ac.uk -p 2020` and press enter
3. You will see the UAL CCI logo if you've logged in correctly:

```

-
| | | |  o
| | | |  o
-
/  _ _ _ _  o  _
\  ( ( ( ( ( ( \  ( (
-  -  -  -
/  _ _ | )  _ _ o _ (
\  ( ) | | | | | | | |
-
| _ _ _ _  o _ _ _
_ | | | > | | | | | (
```

Welcome to the CCI server

Ada Lovelace

cci.arts.ac.uk



1. Type `exit` and press enter to disconnect.

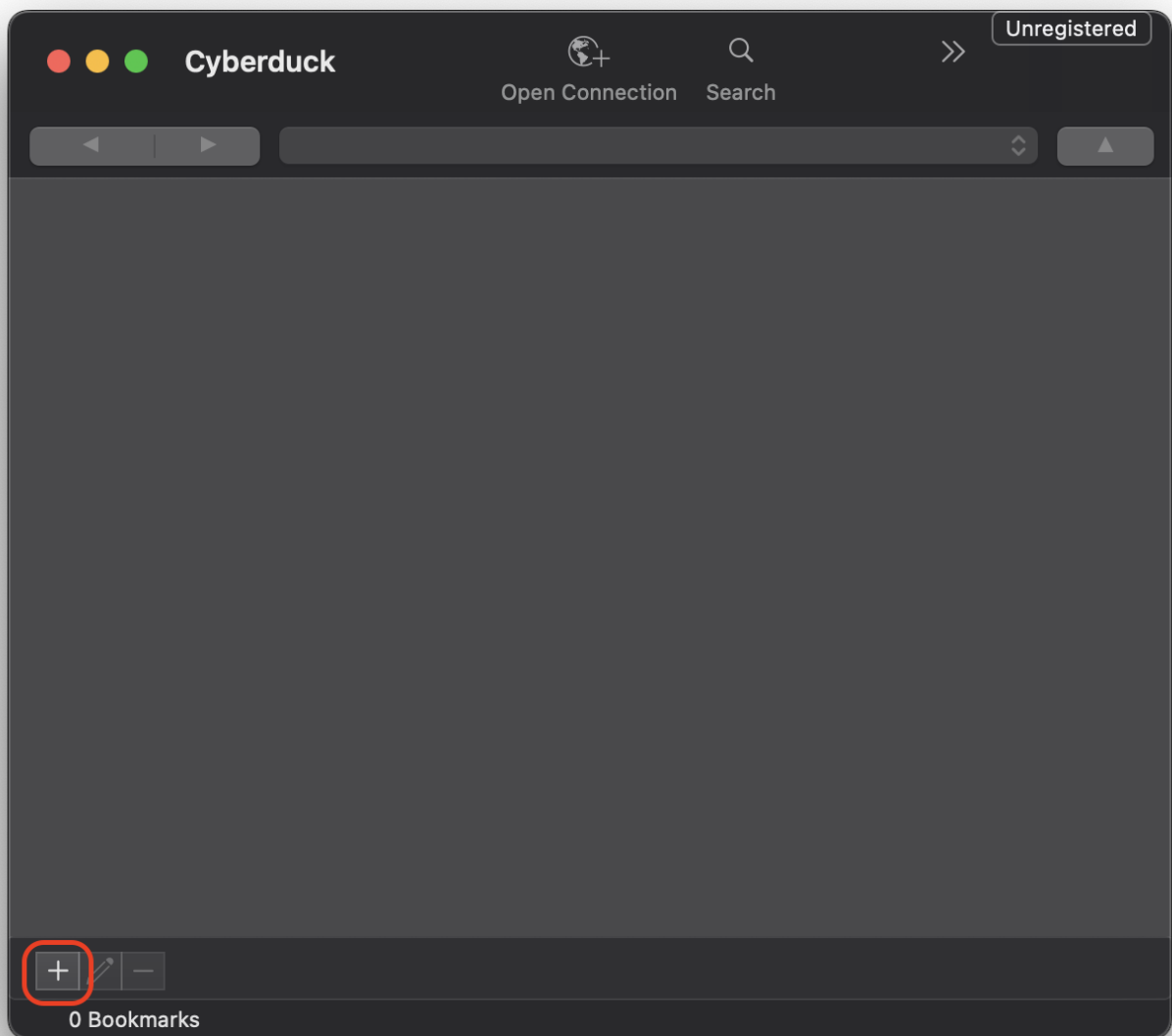
If you've seen the UAL CCI logo it means everything is working, otherwise you'll need to check your steps, or contact support in Slack #technical sharing a screenshot of the issue you're trying.

## Using SFTP with CyberDuck

💡 Tip: Don't type your password as it may not work.

1. Open CyberDuck
2. Click the `+` button in the bottom left corner of the main window

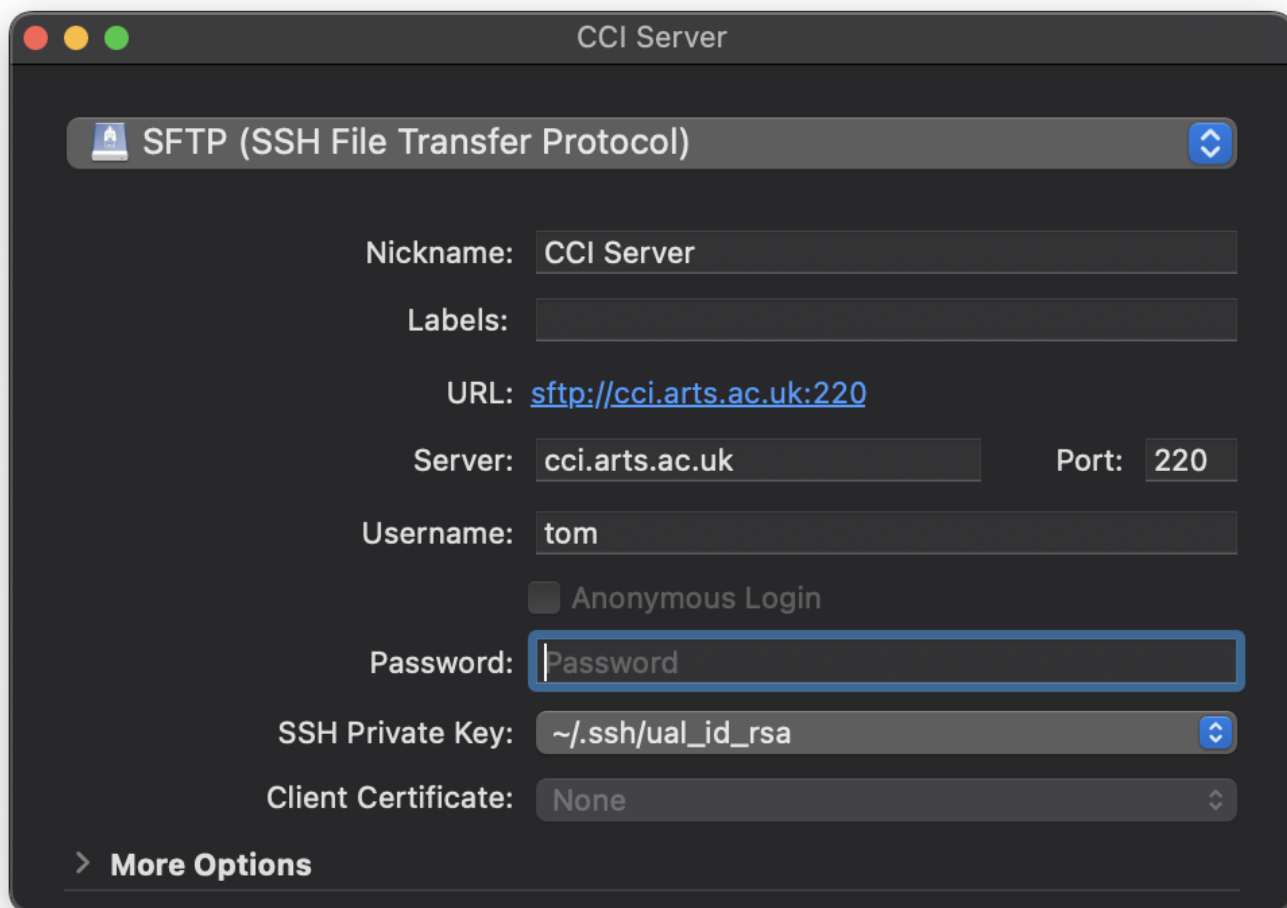




3. Choose "SFTP" from the dropdown list at the top
4. Pick a nickname like "CCI"
5. Enter the server details:

- **Server:**
- **Port:**
- **Username:** *your cci server username*
- **Password:** *leave blank*
- **SSH Private Key:** Pick from the list, there will probably only be one option ending...

6. Close the window, no need to save!



7. Double click on the blue drive icon that appeared with your nickname "CCI" in the main window.

You should now be connected, if you experience issues please screenshot and share in Slack #technical.

# Using CCI's MQTT endpoint

MQTT is a networking protocol that uses a 'publish-subscribe' model (where some machines 'publish' data to an endpoint, and others subscribe to that data stream) to share data. It's commonly used to share networked sensor data.

We run a MQTT server at the CCI, where we publish things like the [air quality data](#), and also allow students to publish dedicated streams for physical computing projects.

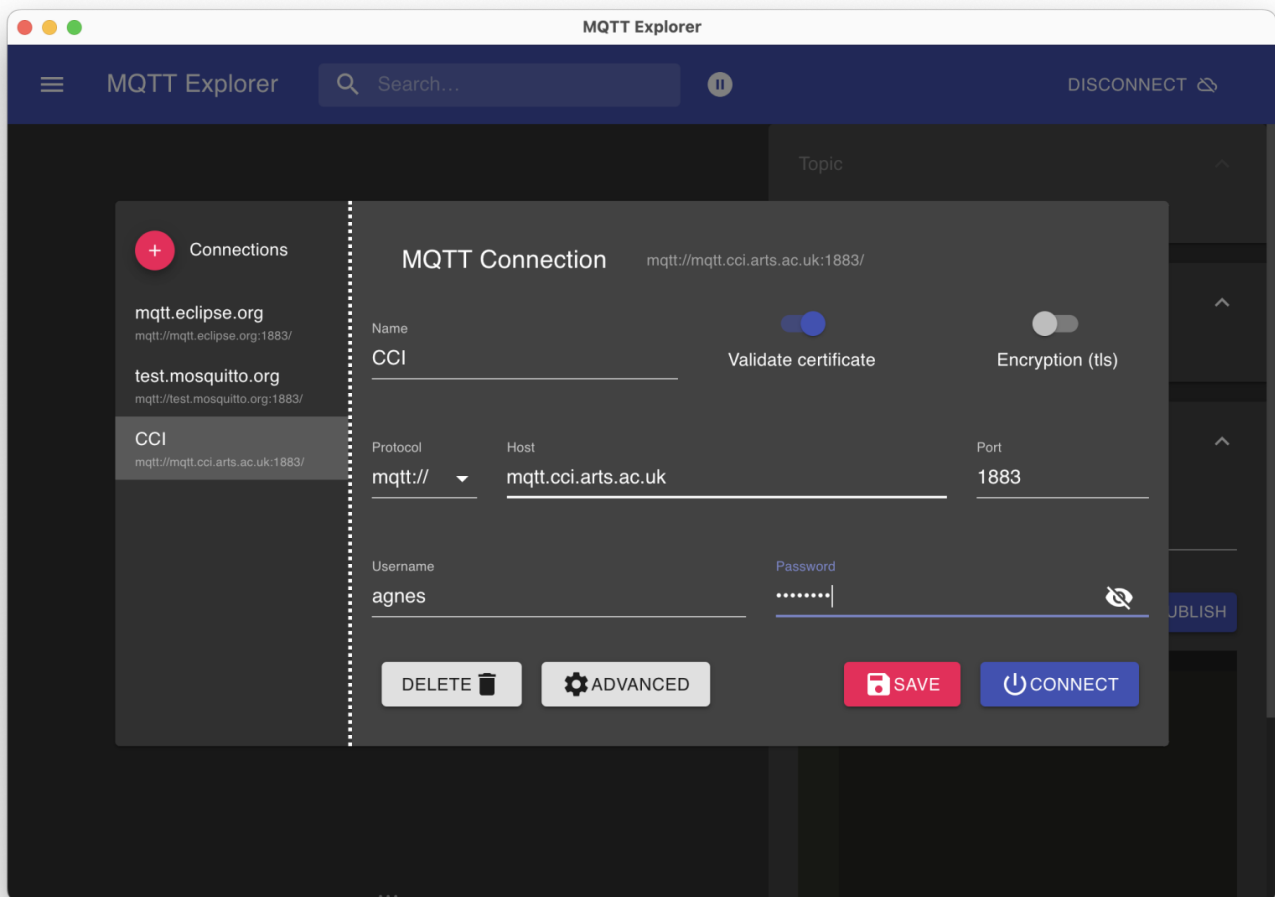
In order to follow any of the below instructions, you will first need a username and password. Ask for this on the #technical channel in Slack! The MQTT server sits at `mqtt.cci.arts.ac.uk` on port `1883`.

## Warning

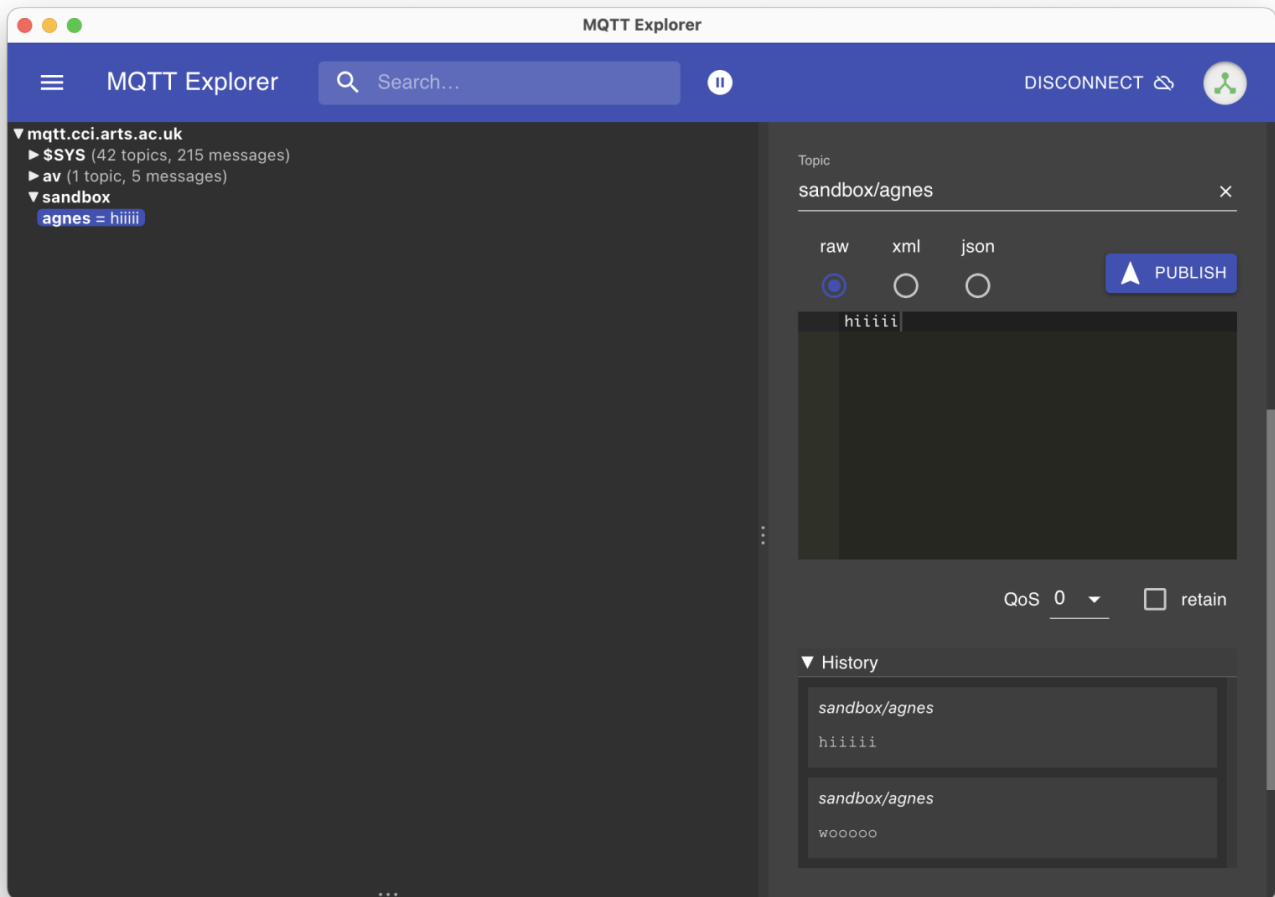
Data published to the CCI's MQTT server is readable by anyone we have given access to! If you need the data you are publishing to be private, consider setting up your own server using a **Raspberry Pi** using the instructions linked below.

## Accessing MQTT Data Through a GUI

To use MQTT on your computer, download an application like [MQTT Explorer](#). The setup for accessing the CCI's topics is as follows.



Topics will appear in the interface as messages are published to them -- if you don't see everything at once, it's because it hasn't yet had a message in the time you've been subscribed. You can also use the interface to publish to a topic (see below).



## MQTT from the Command Line

The best package for programatically interfacing to MQTT is [mosquitto](#). On a mac, this can be installed using Homebrew. Once it is installed, use the following commands in separate terminal windows, substituting the corresponding values for the variables in triangle brackets:

To subscribe:

```
mosquitto_sub -h mqtt.cci.arts.ac.uk -p 1883 \  
-u <your-username> -P <your-password> \  
-t "sandbox/<your-topic>"
```

To publish:

```
mosquitto_pub -h mqtt.cci.arts.ac.uk -p 1883 \  
-u <your-username> -P <your-password> \  
-t "sandbox/<your-topic>" -m "hello everyone"
```

You could also use this method to connect to an MQTT endpoint via a Raspberry Pi -- you can use [these instructions](#) to get set up. (if you're interested in setting up *your own* MQTT endpoint, rather than using ours, you can also [use a Pi to do this](#)).

## MQTT from ESP32 (for use with Arduino)

If you want to publish data using a microcontroller (the ESP32/ESP8266 are obvious candidates, as they're networked already!), the [PubSubClient](#) library provides support for microcontroller boards including the ESP32 and Arduino Ethernet.

There's a nice tutorial for doing this available [here](#) -- note that instead of needing to set up the Raspberry Pi mosquitto broker as they suggest, you can use the CCI's MQTT server as described above.

The last thing you will need to do, if you want to run this project on campus, is to register your devices with [UAL-IoT](#), as they won't work over UAL's general wifi. (if it's just a one-off, phone hotspot wifi will also work for this!)

# How to Connect to MySQL Database on the CCI Server

Please consult this [article](#), if you don't know how to connect to the CCI Server.

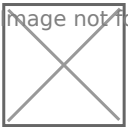
## Download and install MySQL Workbench

MySQL Workbench is a convenient GUI that helps you work with MySQL databases.

1. This [link](#) will take you to the MySQL Workbench's download page.
2. Click on the download button.
3. In case you don't have an Oracle account, you will need to create one (you can use your university email).
4. Download and install MySQL Workbench.

## Connecting MySQL Workbench to CCI Server

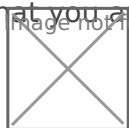
1. Open **Connect to Database** window, by clicking Database > Connect to Database.



2. Name the connection as CCI Server or something similar
3. For the Connection Method select **Standard TCP/IP over SSH.**
4. **SSH Hostname:** cci.arts.ac.uk:2020
5. **SSH Username:** use the one you use to connect to the SSH server.
6. **SSH Password:** use the one you use to connect to the SSH server.
7. **SSH Key File** can be found here /Users/username/.ssh/id\_rsa (.ssh is a hidden folder)
8. **MySQL Hostname:** 127.0.0.1
9. **MySQL Server Port:** 3306
10. **Username and Password:** use the same as SSH Hostname and SSH Password.

## Check if you are connected

The simplest way to check that you are connected is to check the server status. If the server is



running you are good to go!