

## SYS-STEM ArdLAB Setup

### 1 Raspberry PI Setup

To setup the Raspberry PI you should follow these steps:

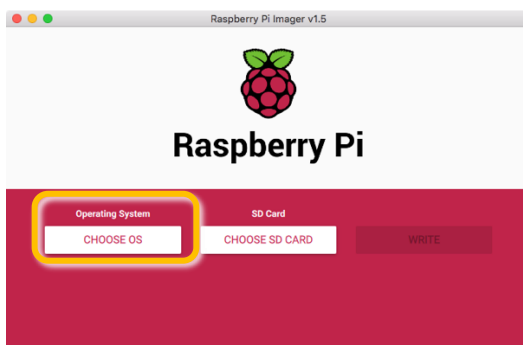
- Download the Raspberry PI OS from
- Burn the Raspberry PI OS image to an microSD card
- Setup the Raspberry PI OS connection and security settings

#### 1.1 Install Raspberry Pi OS

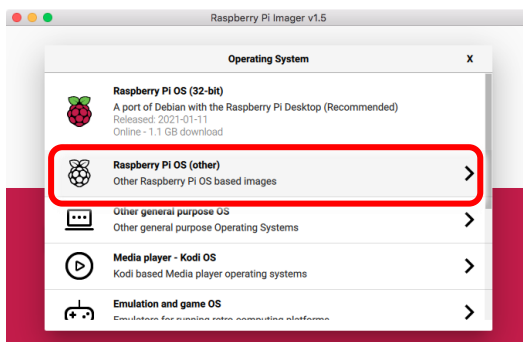
Download the Raspberry PI Imager for your computer Operating System, from the Raspberry website ([link](#)). Install the Raspberry PI Imager and perform the following steps.

##### 1.1.1 Copy the Raspberry Pi OS to the microSD Card

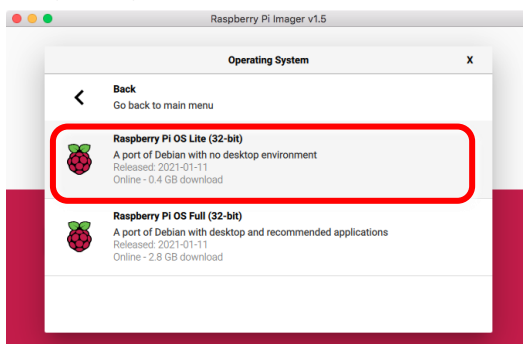
Select “CHOOSE OS”



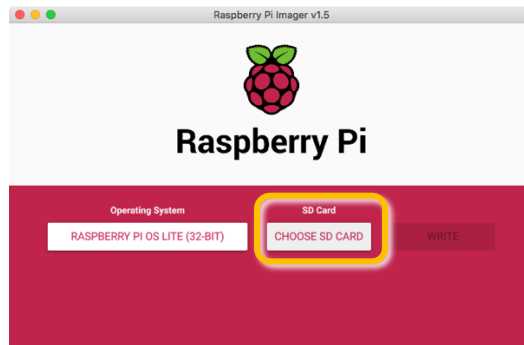
Select “Raspberry Pi OS (Other)”



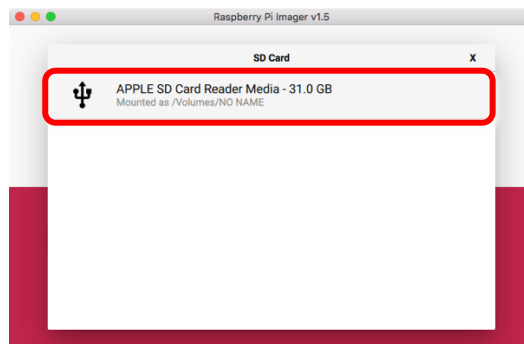
Select “Raspberry Pi OS Lite (32-bit)”



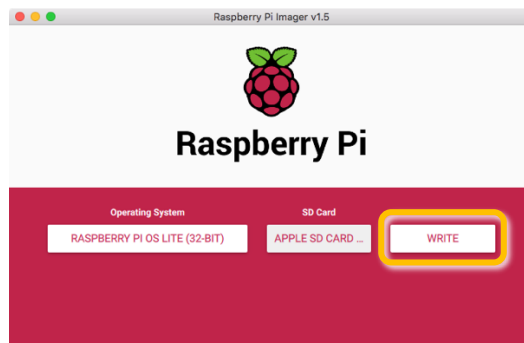
Select “Choose SD Card”



Select “The SD card to use...”

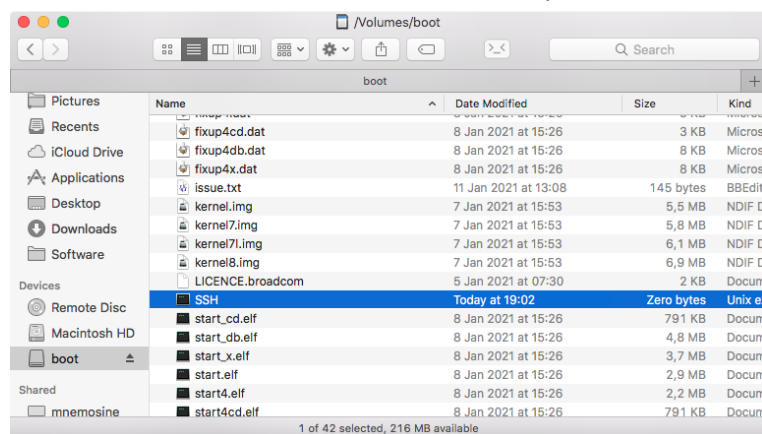


Finally, copy the image to the microSD card pressing “Write”



### 1.1.2 Enable SSH before first boot

To enable remote secure connection (SSH) to the Raspberry Pi before the first boot, an empty file named SSH should be created in the microSD card boot partition.



## 1.2 First boot and basic configuration

Mount the microSD card on your computer and create the file (on windows be very careful, because most editors always add an extension, and the windows default behavior, is to hide known extensions)

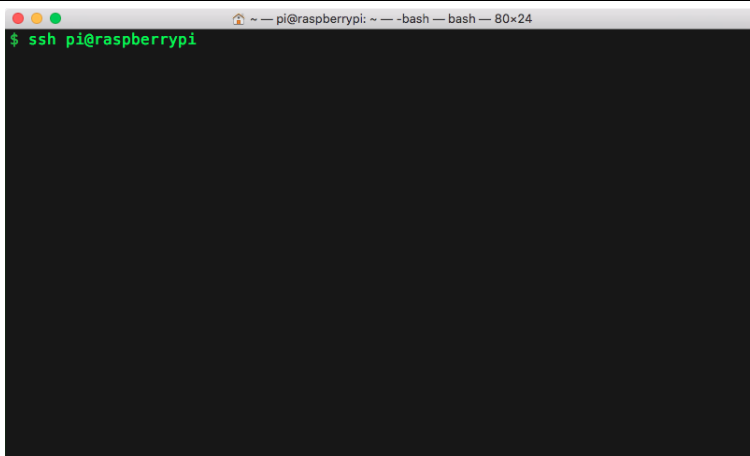
### 1.2.1 Install the microSD Card on the Raspberry Pi, and do the first Run

To prevent the need to connect a monitor and keyboard to the Raspberry Pi, a network cable should be connected to the Raspberry Pi. After the Raspberry Pi boot, you should be able to connect using SSH.

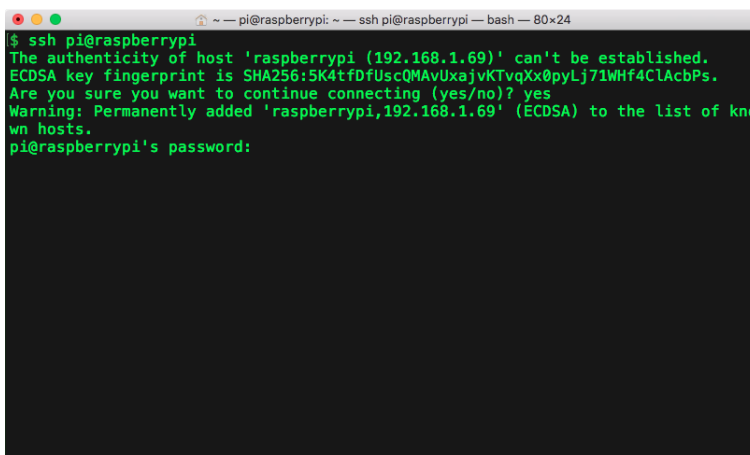
### 1.2.2 Connect to the Raspberry Pi

- To connect to the Raspberry Pi:

```
ssh pi@raspberrypi
```



- When asked for the authenticity of the server, answer “yes” and login with the default password “raspberrypi”.



- If you cannot connect, you will need to add a monitor to the Raspberry PI, identify the IP presented in the console and use it instead of `raspberrypi`, ex:

```
ssh pi@192.168.1.69
```

- After logging in, you should change the PI password for a more secured one. Use the command:

```
passwd
```

```
pi@raspberrypi: ~ -- ssh pi@raspberrypi -- bash -- 80x24
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'raspberrypi,192.168.1.69' (ECDSA) to the list of known hosts.
pi@raspberrypi's password:
Linux raspberrypi 5.4.83-v7l+ #1379 SMP Mon Dec 14 13:11:54 GMT 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Feb  7 19:57:22 2021 from 192.168.1.65
-bash: warning: setlocale: LC_ALL: cannot change locale (pt_PT.UTF-8)

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ passwd
```

- Run the network information program:

```
ifconfig
```

Annotate the eth0 interface IP address

```
pi@ardlab-dei:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.69 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::2826:1161:e4e7:e8f0 prefixlen 64 scopeid 0x20<link>
    inet6 2001:8a0:ff38:f700:e2cf:7707:32ef:9e38 prefixlen 64 scopeid 0x0<
global>
    ether dc:a6:32:7d:99:38 txqueuelen 1000 (Ethernet)
    RX packets 98094 bytes 145123424 (138.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 47839 bytes 4259441 (4.0 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

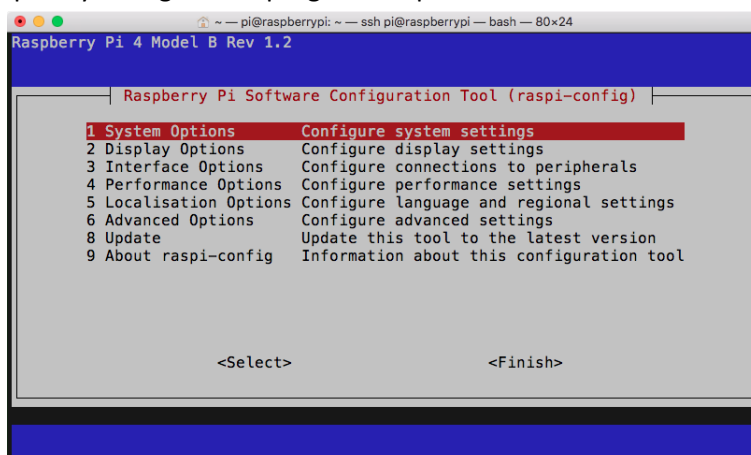
pi@ardlab-dei:~$
```

- Next you can setup some of the default configurations on the Raspberry Pi. You should use the command:

```
sudo raspi-config
```

The `sudo` command allows running the `raspi-config` with root (administrator) permissions.

- The raspberry configuration program is opened



- You should configure the following options:

1. System Options

**S4 Hostname** – You should change the raspberry hostname, ex.: ArdLAB-1. If the Raspberry IP is public, this name should be registered in the DNS server. If the Raspberry IP is private, the IP should be static and a port forwarding rule should be setup in the NAT server for the Raspberry Pi port 7575.

**S1 Wireless LAN, not recommended** – You can configure the Raspberry Pi WiFi connection (you should connect with an ethernet cable)

3. Interface Options

**P1 Camera** – If you are using the Raspberry PI as the ArdLAB camera, you should enable it here

5. Localization Options

**L2 Timezone** – Configure the Timezone for your server, ex. Europe/Lisbon, Europe/Madrid, Europe/Athens, Europe/Zagreb, etc.

After exiting the `raspi-config`, the Raspberry Pi may ask to reboot.

### 1.3 Update the Raspberry Pi OS

Next you should update the Raspberry Pi OS using the following commands in sequence (the `apt-get` command can also be used for updating/upgrading):

```
sudo apt update
sudo apt upgrade
sudo reboot
```

These commands: *update* the packages information, downloading from the configured internet sources; *upgrade* the outdated the packages currently installed; and finally *reboot* the Raspberry Pi

```
pi@ardlab-dei:~$ sudo apt update
Hit:1 http://archive.raspberrypi.org/debian buster InRelease
Hit:2 http://raspbian.raspberrypi.org/raspbian buster InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
28 packages can be upgraded. Run 'apt list --upgradable' to see them.
pi@ardlab-dei:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  bluez-firmware ca-certificates device-tree-compiler file firmware-atheros
  firmware-brcm80211 firmware-libertas firmware-misc-nonfree firmware-realtek
  iproute2 libgnutls30 libldap-2.4-2 libldap-common libmagic-mgc libmagic1
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0
  python-rpi.gpio raspberrypi-bootloader raspberrypi-kernel
  raspberrypi-sys-mods rpi-eeprom rpi.gpio-common sudo tzdata unzip
28 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 137 MB of archives.
After this operation, 3575 kB of additional disk space will be used.
pi@ardlab-dei:~$
```

Install vim editor, for improved vi editor

```
sudo apt install vim
```

To edit files in the terminal, you can use *vi* or *nano*. For users not familiarized with *vi*, the *nano* editor is the best option. After editing a file, press *ctrl+X* to exit, and choose the appropriate answer to save the modifications.

```
nano file.txt
```

## 2 ArdLAB Setup

To setup the ArdLAB you should follow these steps

- Install the Docker daemon in the Raspberry PI
- Clone the ArdLAB repository
- Setup and install the ArdLAB Docker Container(s)
- Create the ArdLAB configuration file
- Connect the Arduino UNO to the Raspberry PI
- Test the Setup

### 2.1 Install docker in the Raspberry Pi

**Docker** is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. Because all of the containers share the services of a single operating system kernel, they use fewer resources than virtual machines.

These steps must be followed to install dockers in the Raspberry PI (based on instructions available in this [link](#))

- Install the prerequisites

```
sudo apt install apt-transport-https ca-certificates software-properties-common -y
```

- Restart the Raspberry Pi

```
sudo reboot
```

- Download the Docker installer and run it

```
curl -fsSL get.docker.com -o get-docker.sh && sh get-docker.sh
```

- Give the 'pi' user the ability to run Docker.

```
sudo usermod -aG docker pi
```

- Import Docker CPG key

```
sudo curl -fsSL https://download.docker.com/linux/raspbian/gpg |  
sudo apt-key add -
```

- Add the docker repo to the apt sources list

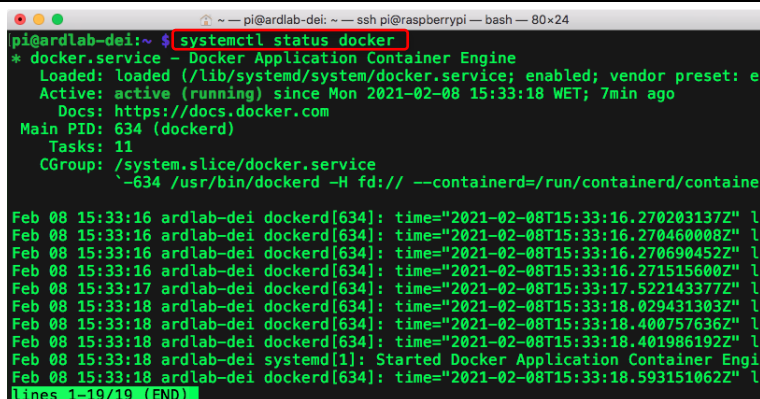
```
sudo sh -c 'echo "deb https://download.docker.com/linux/raspbian/  
buster stable" >> /etc/apt/sources.list'
```

- Update the Raspberry Pi OS and restart

```
sudo apt update  
sudo apt upgrade  
sudo reboot
```

- Check if the docker daemon is running (press q to exit listing)

```
systemctl status docker
```



```
pi@ardlab-dei:~$ systemctl status docker
* docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: e
   Active: active (running) since Mon 2021-02-08 15:33:18 WET; 7min ago
     Docs: https://docs.docker.com
    Main PID: 634 (dockerd)
      Tasks: 11
   CGroup: /system.slice/docker.service
           └─634 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containe

Feb 08 15:33:16 ardlab-dei dockerd[634]: time="2021-02-08T15:33:16.270203137Z" l
Feb 08 15:33:16 ardlab-dei dockerd[634]: time="2021-02-08T15:33:16.270460087Z" l
Feb 08 15:33:16 ardlab-dei dockerd[634]: time="2021-02-08T15:33:16.270690452Z" l
Feb 08 15:33:16 ardlab-dei dockerd[634]: time="2021-02-08T15:33:16.271515600Z" l
Feb 08 15:33:17 ardlab-dei dockerd[634]: time="2021-02-08T15:33:17.522143377Z" l
Feb 08 15:33:18 ardlab-dei dockerd[634]: time="2021-02-08T15:33:18.029431303Z" l
Feb 08 15:33:18 ardlab-dei dockerd[634]: time="2021-02-08T15:33:18.400757636Z" l
Feb 08 15:33:18 ardlab-dei dockerd[634]: time="2021-02-08T15:33:18.401986192Z" l
Feb 08 15:33:18 ardlab-dei systemd[1]: Started Docker Application Container Engi
Feb 08 15:33:18 ardlab-dei dockerd[634]: time="2021-02-08T15:33:18.593151062Z" l
lines 1-19/19 (END)
```

After rebooting, the docker service should be running. To manage the docker service you can use the following `systemctl` options:

*start* – Start the service

*stop* – Stop the service

*restart* – Restart the service, equivalent to *stop* + *start*

*reload* – Reload the service configuration files

*enable* – Enable the service (auto-runs on boot)

*disable* – Disable the service

- Check the installed docker info

```
docker info
```

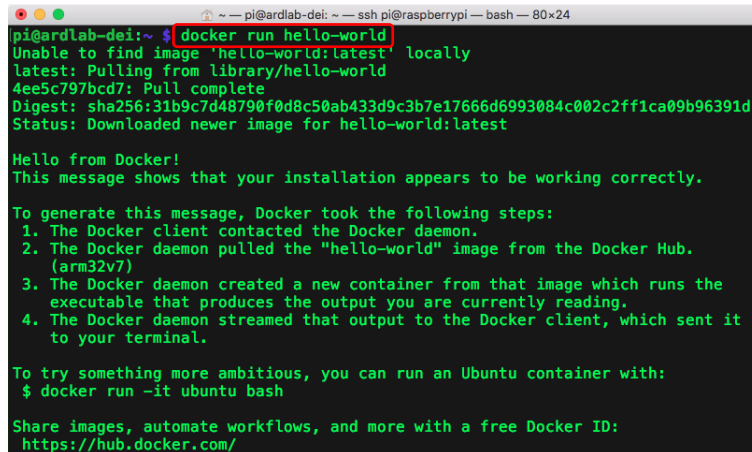
- Install the docker compose

```
sudo apt install docker-compose
```

## 2.2 Create and run the first container

- To test the Docker installation, create and run a hello-world container

```
docker run hello-world
```



```
pi@ardlab-dei:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
4ee5c797bcd7: Pull complete
Digest: sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c2ff1ca09b96391d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (arm32v7)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
```

If you see the previous message, the container was created and runned. To manage the containers and images, the following command can be used:

- Show container images existing locally

```
docker image ls
```

- Show docker help page

```
docker --help
```

- Show docker command help page

```
docker image --help
```

- Remove local image (use the short hash or name, use `docker image ls` to display names/hashs)

Use the TAB key to autocomplete names/commands.

- Images used in containers cannot be removed without removing the container.

```
docker image remove hello-world:latest
```

- An alternative command to remove images is:

```
docker rmi hello-world:latest
```

- Show running containers

```
docker ps
```

- Show all containers

```
docker ps -a
```

- You can now remove the hello-world container, using the name/hash

```
docker rm happy_rhodes
```



- Alternative command to remove container using the hash key

```
docker rm 2cdaa3512ba
```

The result of some of the previous commands:

```
pi@ardlab-dei:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    851163c78e4a   13 months ago 4.85kB
pi@ardlab-dei:~$ docker image remove hello-world:latest
Error response from daemon: conflict: unable to remove repository reference "hello-world:latest" (must force) - container 2cdaa3512ba is using its referenced image 851163c78e4a
pi@ardlab-dei:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
2cdaa3512ba    hello-world "/hello"                34 seconds ago Exited (0) 33 seconds ago
happy_rhodes
pi@ardlab-dei:~$ docker rm happy_rhodes
happy_rhodes
pi@ardlab-dei:~$ docker rmi hello-world:latest
Untagged: hello-world:latest
Untagged: hello-world@sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c2f1ca09b96391d
Deleted: sha256:851163c78e4ad68e6fe5391f0894aafd164d40c4d4d0a56b4291f0dc2c75cc2c
Deleted: sha256:2536d8d4e4b1baa6515d44eb77a1402d6be0a533e7d191c51cb8428ba5ece3f4
pi@ardlab-dei:~$
```

## 2.3 Setup the ArdLAB

- Clone the repository inside the pi user home, using the following command:

```
git clone https://bitbucket.org/pssmatos/ardlab-setup.git
```

A new folder named `ardlab-setup` is created with the container configuration file

### 2.3.1 Setup and install the ArdLAB Docker Container

- Change the current directory to `ardlab-setup` (you can use the TAB key to autocomplete the directory name)

```
cd ardlab-setup
```

- List the directory contents:

```
ls -la
```

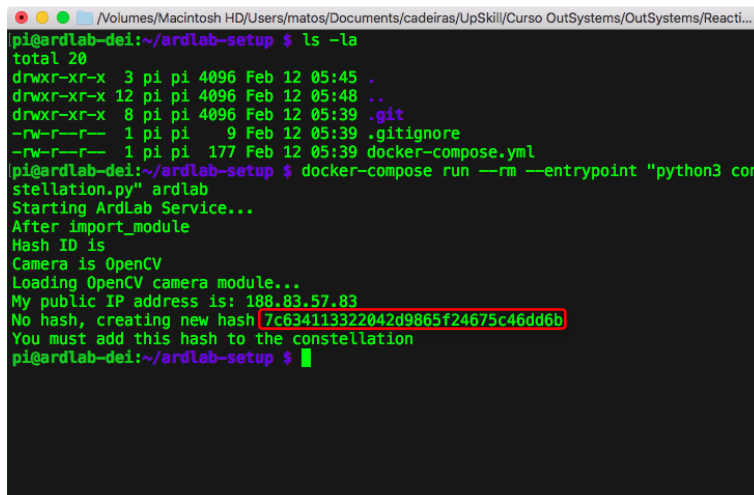
There is only one important file in this repository, the docker-compose definition file

```
/Volumes/Macintosh HD/Users/matos/Documents/cadeiras/UpSkill/Curso OutSystems/OutSystems/Reacti...
pi@ardlab-dei:~$ git clone https://pssmatos@bitbucket.org/pssmatos/ardlab-setup
.git
Cloning into 'ardlab-setup'...
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (4/4), done.
pi@ardlab-dei:~$ cd ardlab-setup/
pi@ardlab-dei:~/ardlab-setup$ ls -la
total 20
drwxr-xr-x  3 pi pi 4096 Feb 12 05:39 .
drwxr-xr-x 12 pi pi 4096 Feb 12 05:39 ..
drwxr-xr-x  8 pi pi 4096 Feb 12 05:39 .git
-rw-r--r--  1 pi pi   9 Feb 12 05:39 .gitignore
-rw-r--r--  1 pi pi 177 Feb 12 05:39 docker-compose.yml
pi@ardlab-dei:~/ardlab-setup$
```

- Start the configuration by running the container to create the configuration file with the following command:

```
docker-compose run --rm --entrypoint "python3 constellation.py"
ardlab
```

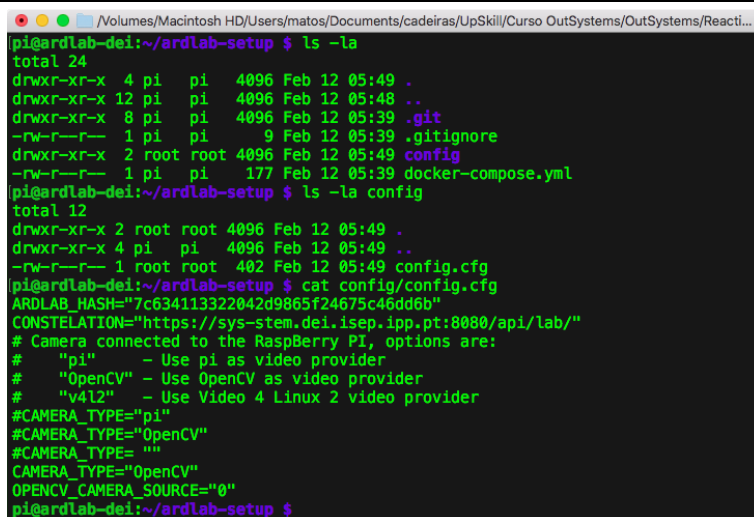
The script should run and present some info: the public IP address, the new ArdLAB hash code, and one message to register this hash in the constellation site (either by adding a new lab or editing an existing one)



```
pi@ardlab-dei:~/ardlab-setup $ ls -la
total 20
drwxr-xr-x  3 pi pi 4096 Feb 12 05:45 .
drwxr-xr-x 12 pi pi 4096 Feb 12 05:48 ..
drwxr-xr-x  8 pi pi 4096 Feb 12 05:39 .git
-rw-r--r--  1 pi pi   9 Feb 12 05:39 .gitignore
-rw-r--r--  1 pi pi 177 Feb 12 05:39 docker-compose.yml
pi@ardlab-dei:~/ardlab-setup $ docker-compose run --rm --entrypoint "python3 constellation.py" ardlab
Starting ArdLab Service...
After import_module
Hash ID is
Camera is OpenCV
Loading OpenCV camera module...
My public IP address is: 188.83.57.83
No hash, creating new hash 7c634113322042d9865f24675c46dd6b
You must add this hash to the constellation
pi@ardlab-dei:~/ardlab-setup $
```

- The previous step also created the configuration file, included in the directory config that can be presented using:

```
cat config/config.cfg
```



```
pi@ardlab-dei:~/ardlab-setup $ ls -la
total 24
drwxr-xr-x  4 pi pi 4096 Feb 12 05:49 .
drwxr-xr-x 12 pi pi 4096 Feb 12 05:48 ..
drwxr-xr-x  8 pi pi 4096 Feb 12 05:39 .git
-rw-r--r--  1 pi pi   9 Feb 12 05:39 .gitignore
drwxr-xr-x  2 root root 4096 Feb 12 05:49 config
-rw-r--r--  1 pi pi 177 Feb 12 05:39 docker-compose.yml
pi@ardlab-dei:~/ardlab-setup $ ls -la config
total 12
drwxr-xr-x  2 root root 4096 Feb 12 05:49 .
drwxr-xr-x  4 pi pi 4096 Feb 12 05:49 ..
-rw-r--r--  1 root root 402 Feb 12 05:49 config.cfg
pi@ardlab-dei:~/ardlab-setup $ cat config/config.cfg
ARDLAB_HASH="7c634113322042d9865f24675c46dd6b"
CONSTELATION="https://sys-stem.dei.isep.ipp.pt:8080/api/lab/"
# Camera connected to the RaspBerry PI, options are:
# "pi" - Use pi as video provider
# "OpenCV" - Use OpenCV as video provider
# "v4l2" - Use Video 4 Linux 2 video provider
#CAMERA_TYPE="pi"
#CAMERA_TYPE="OpenCV"
#CAMERA_TYPE=""
CAMERA_TYPE="OpenCV"
OPENCV_CAMERA_SOURCE=""
pi@ardlab-dei:~/ardlab-setup $
```

This configuration file will be edited in a later step, to configure your ArdLAB name, port and camera.

### 2.3.2 Connect the Arduino UNO to the Raspberry PI

Now you can connect the Arduino UNO to the raspberry PI using one of the USB ports

### 3 Configure dDNS and obtain SSL certificates

To access the ArdLAB with encryption (HTTPS), we need to use a valid SSL certificate, which depending on the laboratory can be obtained using two distinct options.

If the ArdLAB has a fixed public IP address and an valid DNS name, continue on the section 3.3.

In case your ArdLAB has one of the following restrictions, continue to the respective section:

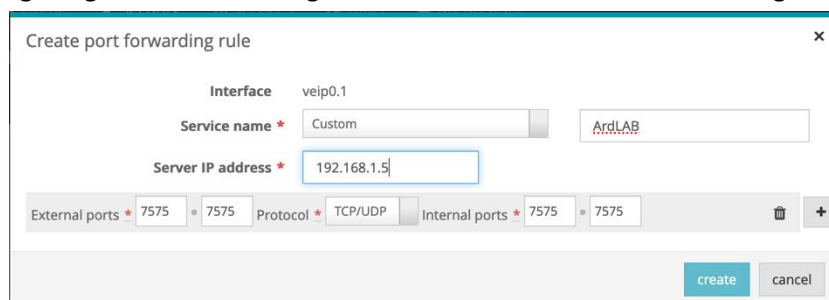
- Fixed IP address, but you cannot assign it a DNS name (section 3.2 and 3.3)
- Dynamic public IP address (section 3.2 and 3.3)
- Private IP address (sections 3.1, 3.2 and 3.3)

#### 3.1 Adding port forward rule to your ArdLAB

If your ArdLAB is behind a NAT router, it has a private IP address, and the lab will not be accessible from the internet. To overcome this problem, you must add a *FORWARD PORT RULE* on the NAT server, to redirect incoming traffic to the ArdLAB communication port (7575).

Ask your network administrator to create such a rule, or if you have access to the NAT server (or router) configure it yourself. The default ArdLAB port is 7575 (you can use another if the 7575 port is unavailable) and the server IP address will be the ArdLAB private IP.

The following image shows the configuration of such a rule at a router configuration page.



#### 3.2 Adding your Raspberry PI to a DDNS service

If your ArdLAB doesn't have a fixed IP with a valid DNS name, you will need to create an account in a DDNS service provider and create a new DNS registry for the ArdLAB.

In this example, we will use the Dynu dDNS service (<https://www.dynu.com/>).

It is important that the service provider can create "TXT records" using the **ACME Shell script: acme.sh** dnsapi (<https://github.com/acmesh-official/acme.sh/wiki/dnsapi>)

**NOTE:** If you have a private address or a public IP without DNS name on your ArdLAB, continue to the next section, otherwise go to section 3.3.

### 3.2.1 Create an account on the Dynu dDNS service

- Access the <https://www.dynu.com/> site:
  - 1 Click the “Create Account” link on top right
  - 2 Fill your data
  - 3 Submit the “Create Account” request
  - 4 Open your email, search for the “Dynu Verification Email” and click the URL

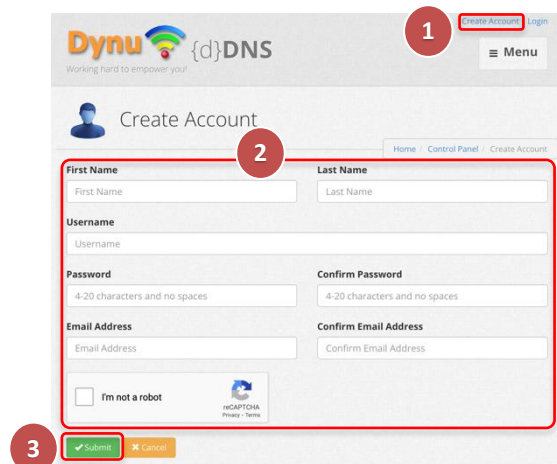


Figure 1 - Create an account on the Dynu dDNS service

### 3.2.2 Create a DNS registry for your ArdLAB

- Login to your account in the <https://www.dynu.com/> site:
  - 1 Access The “Control Panel” (gear icon)
  - 2 On the “Control Panel”, select “DDNS Services”
  - 3 On the “Dynamic DNS Service” Page, click on “+Add”
  - 4 Fill your ArdLAB name (host) and choose a “Top Level” domain
  - 5 Click “+ Add”

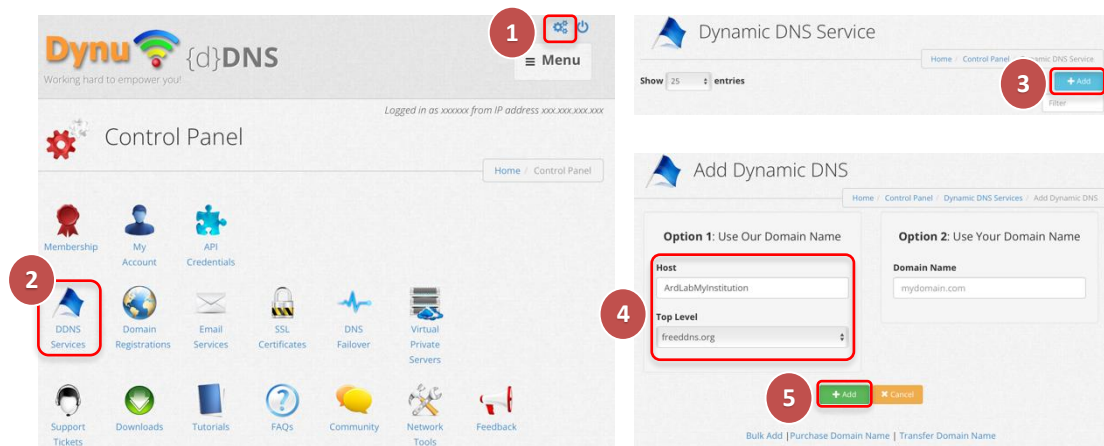


Figure 2 - Create a DNS registry for your ArdLAB

### 3.2.3 Manage your ArdLAB dDNS registry

- The “Manage DNS Service” appears, and you should:
  - 1 Change your IPv4 Address to a “invalid” one, ex: 1.1.1.1
  - 2 Click “Save” to update your IPv4 address
  - 3 Click the “IP Update Password” on the bottom of the page
- On the “Manage Credentials” page:
  - 4 Enter your Dynu Password
  - 5 Enter a new IP update Password (this password will be used on the ArdLAB)
  - 6 Click “Save” to update the password

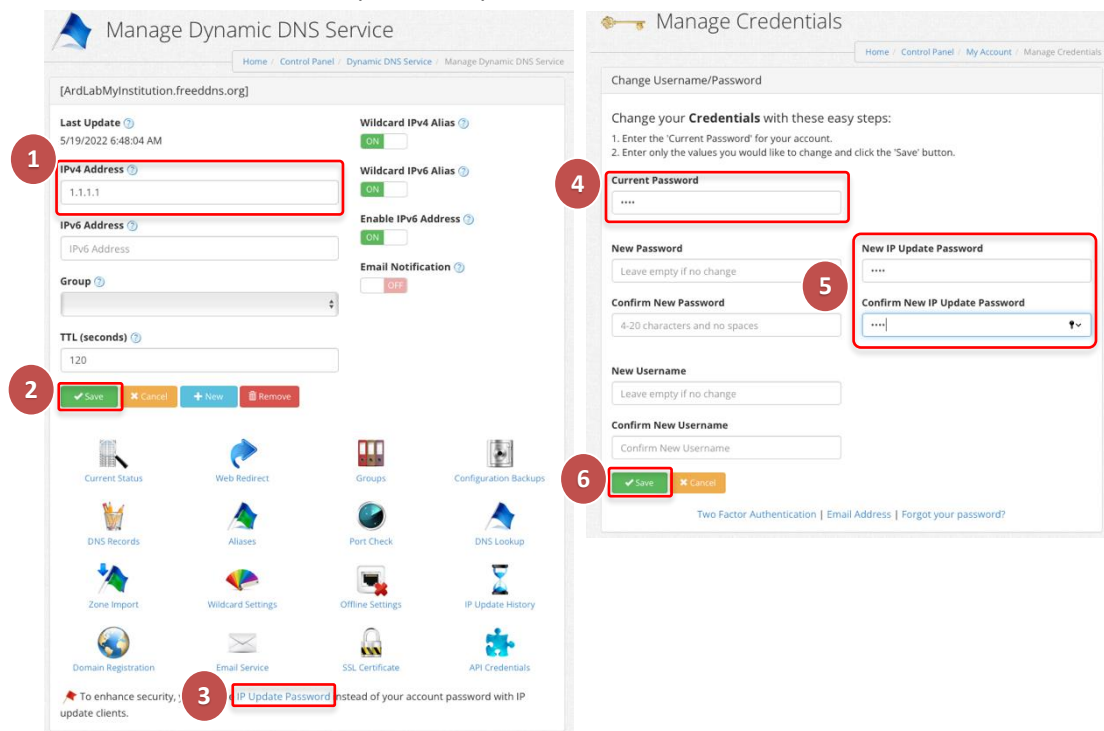


Figure 3 -Manage your ArdLAB dDNS registry

### 3.2.4 Configure the ArdLAB to automatically update the Dynu IP address

Now we need to configure the raspberry to publish the ArdLAB public address to the Dynu site (instructions on <https://www.dynu.com/DynamicDNS/IPUpdateClient/RaspberryPi-Dynamic-DNS>).

- On the raspberry home directory create a new directory named `dynudns` and change to it

```
cd ~
mkdir dynudns
cd dynudns
```

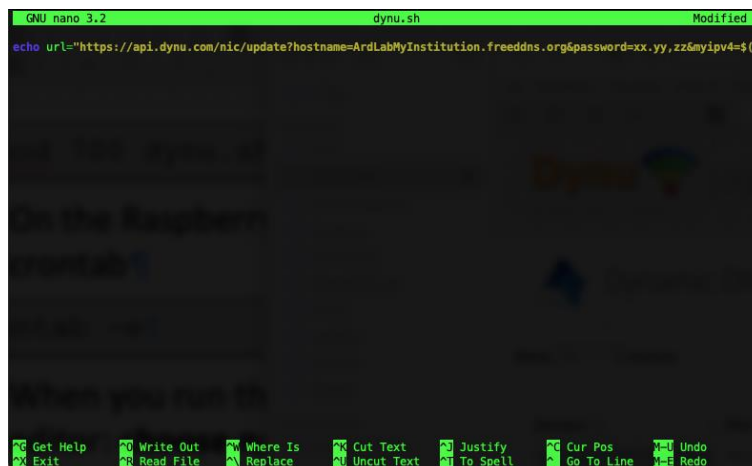
- Create a new script named `dynu.sh`

```
nano dynu.sh
```

- Add this line to the dynu.sh script, changing HOST for the “Domain” you created on the Dynu site, and IP\_PASS with the “IP update password” you defined on the Dynu site.

```
echo
url="https://api.dynu.com/nic/update?hostname=HOST&password=IP_PAS
S&myipv4=$(curl -s -4 ifconfig.co) &myipv6=$(curl -s -6
ifconfig.co) " | curl -k -o ~/dynudns/dynu.log -K -
```

Check that the command is written in only one line, and the hostname and password are ok



Press CTRL+X to save, and Y to confirm saving the document, then ENTER to confirm the filename

- Change the script permissions to 700

```
chmod 700 dynu.sh
```

- On the Raspberry Pi terminal, run the following command, to add the script to your crontab

```
crontab -e
```

When you run this command for the first time, you must select your favorite text editor:  
**choose nano**

- Paste the following line at the bottom of the text editor

```
* /5 * * * * ~/dynudns/dynu.sh >/dev/null 2>&1
```

Press CTRL+X to save, and Y to confirm saving the document, then ENTER to confirm the filename

```
GNU nano 3.2 /tmp/crontab.7nMj5i/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/5 * * * * ~/dynudns/dynu.sh >/dev/null 2>&1

Get Help  Write Out  Where Is  Cut Text  Justify  Cur Pos  Undo
Exit      Read File  Replace  Uncut Text  To Spell  Go To Line  Redo
```

Now you have a cronjob that runs every 5 minutes and will update your public IP on the Dynu dDNS server. You can check on your Dynu subdomain, to see if the IP address changes in the next 5 minutes.

If the IP doesn't update, you can view the `~/dynudns/dynu.log` file and check the error. If you get `badauth`, it means you have supplied the wrong domain or IP update password.

### 3.3 SSL certificate issuing and renewal

To access the server using HTTPS, a valid SSL certificate is needed. To simplify the certificate obtention and renewal, a certificate automation tool is needed. We will use the `acme.sh` (ACME Shell script), which can obtain and renew certificates from various Certification Authorities (CAs) and use several Challenge Types to confirm the domain ownership.

#### 3.3.1 Install the `acme.sh` script to automate the certificate renewal

- Install the `acme.sh` script

```
cd ~
curl https://get.acme.sh | sh -s email=my@example.com
```

- After the installation, you must close the current terminal and reopen it to make the alias take effect. Login again and run the following command

```
acme.sh --version
```

If you see the `acme.sh` version, the script is correctly installed, and you can proceed to generate the certificate.

**Note:** If you have an old "ArdLAB" version you will need to update now the container.

```
cd ~/ardlab-setup
git pull
docker-compose down
docker-compose pull
docker-compose up --detach
```



### 3.3.2 Generate the certificate for ArdLAB on private IP (using port forward)

If you have a private IP and created a Dynu dDNS record, you now need to generate the certificate using a DNS challenge (if you have a public IP address, continue to section 3.3.3.).

On your home directory, run the following commands, replacing the `ClientId` and `Secret` with your api keys, obtained on the Dynu “Manage Dynamic DNS Service” → “api credentials” (see Figure 3). You also need to replace the `DOMAIN` after the `-d` with your own Dynu dDNS domain.

**Note:** You only need to export the credentials on the first run, because `acme.sh` stores the keys on its configuration files.

- This configuration steps are presented on the `acme.sh` “How to use DNS API” page (<https://github.com/acmesh-official/acme.sh/wiki/dnsapi#24-use-dynu-api>)

```
export Dynu_ClientId=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx
export Dynu_Secret=yyyyyyyyyyyyyyyyyyyyyyyyyy
acme.sh --issue --dns dns_dynu -d DOMAIN \
    --cert-file /home/pi/ardlab-setup/certs/cert.pem \
    --key-file /home/pi/ardlab-setup/certs/key.pem \
    --fullchain-file /home/pi/ardlab-setup/certs/fullchain.pem \
    --reloadcmd "docker restart ardlab-setup_ardlab_1"
```

- After the command ends, you can check your issued certificates with the following command

```
acme.sh -list
```

You can now continue to section 0.

### 3.3.3 Generate the certificate on ArdLAB with public IP

- To generate the certificate on the ArdLAB with public IP, you need to install the `socat`

```
sudo apt install socat
```

- Then you need to allow `socat` to accept connections on restricted ports

```
sudo setcap 'cap_net_bind_service=+ep' /usr/bin/socat
```

- Finally, you can run the `acme.sh` to request for the new certificate. Don't forget to replace the `DOMAIN` after the `-d` with your own ArdLAB domain.

```
acme.sh --issue --standalone -d DOMAIN \
    --cert-file /home/pi/ardlab-setup/certs/cert.pem \
    --key-file /home/pi/ardlab-setup/certs/key.pem \
    --fullchain-file /home/pi/ardlab-setup/certs/fullchain.pem \
    --reloadcmd "docker restart ardlab-setup_ardlab_1"
```

- After the command ends, you can check your issued certificates with the following command

```
acme.sh -list
```



## 4 Add the ArdLAB to the Constellation

To add the ArdLAB to the Sys-Stem ArdLAB Constellation, you should follow these steps:

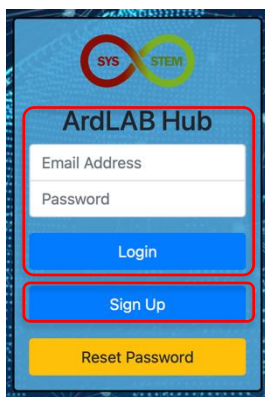
- Create an ArdLAB Constellation user account (only if you don't have one)
- Add your ArdLAB to the Constellation (must be approved by the Constellation administrator)
- Run the ArdLAB configuration
- Run the ArdLAB container
- Add sample sketches and test your ArdLAB in the Constellation Site

### 4.1 Configure the ArdLAB on the Constellation Hub

To add your ArdLAB to the Constellation Hub, you need to have an account and then add your ArdLAB parameters to the hub.

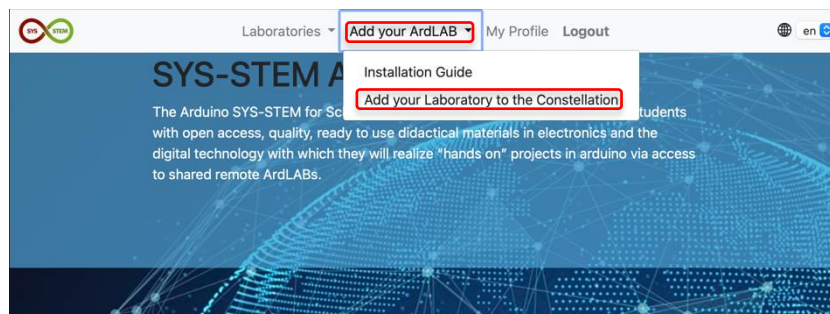
#### 4.1.1 Create an ArdLAB Constellation user account

- Access the SYS-STEM ArdLAB Constellation site (<https://sys-stem.dei.isep.ipp.pt/>)
- If you already have an account Login, otherwise, use the Sign-up button to create a new user account.

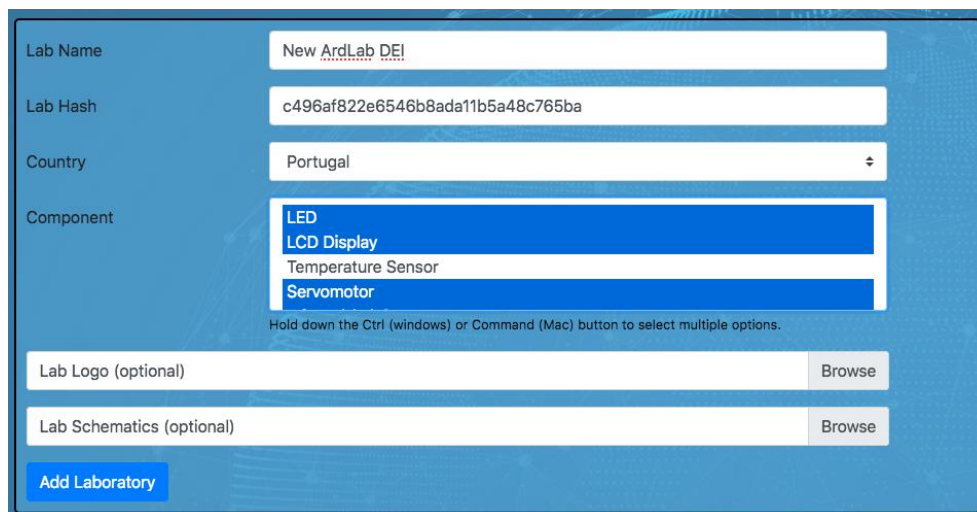


#### 4.1.2 Add your ArdLAB to the Constellation

- Select the “Add your ArdLAB” → “Add your Laboratory to the Constellation” menu option.



- Fill the New ArdLAB form, use the hash created previously, and press the “Add Laboratory” button



The screenshot shows a web form for adding a new ArdLAB. The fields are: Lab Name (New ArdLab DEI), Lab Hash (c496af822e6546b8ada11b5a48c765ba), Country (Portugal), Component (LED, LCD Display, Temperature Sensor, Servomotor), Lab Logo (optional) with a Browse button, and Lab Schematics (optional) with a Browse button. A blue 'Add Laboratory' button is at the bottom left. A note below the Component field says: 'Hold down the Ctrl (windows) or Command (Mac) button to select multiple options.'

The Laboratory needs to be approved by one of the SYS-STEM Constellation administrators, so before you can proceed with the following configuration, you need to wait for the authorization (you will receive an email after the ArdLAB authorization).

## 4.2 Run the ArdLAB configuration

- Now we can edit the ArdLAB configuration file, to configure the ArdLAB DNS name, port and camera

```
cd ~/ardlab-setup
sudo nano config/congif.cfg
```

- The following is an config.cfg sample with some of the most common configurations

```
ARDLAB_HASH="a826b23115b84f0d8c0d040b89739b"
CONSTELLATION="https://sys-stem.dei.isep.ipp.pt:8080/api/lab/"
HOSTNAME="ardlab.myInstitution.com"
FORWARD_PORT=7575
# Camera connected to the RaspBerry PI, options are:
#   "pi"      - Use pi as video provider
#   "OpenCV" - Use OpenCV as video provider
#CAMERA_TYPE="pi"
CAMERA_TYPE="OpenCV"
OPENCV_CAMERA_SOURCE="0"
```

The parameters you can change are:

**HOSTNAME** – is the name you registered on the DNS server for your ArdLAB

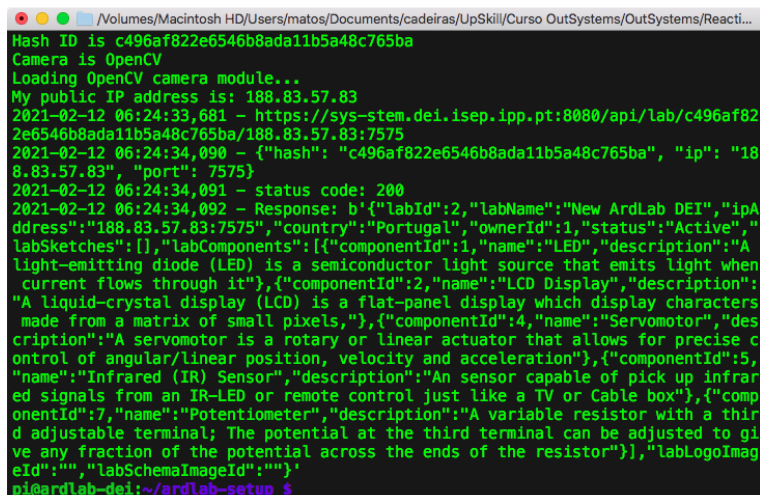
**FORWARD\_PORT** – is the port you forward on your NAT router. You only need to add this entry if the NAT router port is different from 7575

**CAMERA\_TYPE** – use PI for the raspberry PI camera and OpenCV for an USB camera

- When the Laboratory is approved, you need to run again the container to “connect” to the Constellation

```
docker-compose run --rm --entrypoint "python3 constellation.py"
ardlab
```

If the command succeeds, the following information is presented by the `constellation.py` script



```
Hash ID is c496af822e6546b8ada11b5a48c765ba
Camera is OpenCV
Loading OpenCV camera module...
My public IP address is: 188.83.57.83
2021-02-12 06:24:33,681 - https://sys-stem.dei.isep.ipp.pt:8080/api/lab/c496af82
2e6546b8ada11b5a48c765ba/188.83.57.83:7575
2021-02-12 06:24:34,090 - {"hash": "c496af822e6546b8ada11b5a48c765ba", "ip": "18
8.83.57.83", "port": 7575}
2021-02-12 06:24:34,091 - status code: 200
2021-02-12 06:24:34,092 - Response: b'{"labId":2,"labName":"New ArdLab DEI","ipA
ddress":"188.83.57.83:7575","country":"Portugal","ownerId":1,"status":"Active","
labSketches":[],"labComponents":[{"componentId":1,"name":"LED","description":"A
light-emitting diode (LED) is a semiconductor light source that emits light when
current flows through it"}, {"componentId":2,"name":"LCD Display","description":
"A liquid-crystal display (LCD) is a flat-panel display which display characters
made from a matrix of small pixels,"}, {"componentId":4,"name":"Servomotor","des
cription":"A servomotor is a rotary or linear actuator that allows for precise c
ontrol of angular/linear position, velocity and acceleration"}, {"componentId":5,
"name":"Infrared (IR) Sensor","description":"An sensor capable of pick up infrar
ed signals from an IR-LED or remote control just like a TV or Cable box"}, {"comp
onentId":7,"name":"Potentiometer","description":"A variable resistor with a thir
d adjustable terminal; The potential at the third terminal can be adjusted to gi
ve any fraction of the potential across the ends of the resistor"}],"labLogoImag
eId":"","labSchemaImageId":""}'
pi@ardlab-dei:~/ardlab-setup $
```

#### 4.2.1 Run the ArdLAB container

- Finally, the ArdLAB container can be runned to serve the camera and the programming API

```
docker-compose up --detach
```

The container will restart when the Raspberry PI starts or if the container webserver crashes.

- If you need to stop the container, use the command:

```
docker-compose down
```

- If in the future you need to update the container, you must do (after bringing down the container):

```
docker-compose pull
```

The update is silent and can take some minutes. After updating the container, you must run the container again, using the first command in this section.

You can check if the ArdLAB is running, accessing the following link:

<https://ArdLAB-DNS-Name:7575>