**MASTER OF TECHNOLOGY**

**(INTELLIGENT SYSTEMS)**

**Installation and User Guide**

**Mycroft Discovery and Application**

GROUP MEMBERS

LI XIAO XIA

YANG YANG

TAN WOEI MING

JIANG AI GUO

ANG JENN NING

MASTER OF TECHNOLOGY Institute of System Science, National University of Singapore

[**1.** **System Overview** 3](#_Toc70718179)

[**2.** **Web Setting and User Guide** 3](#_Toc70718180)

[**2.1** **Mycroft Home Setting** 3](#_Toc70718181)

[**2.1.1 Create Account** 3](#_Toc70718182)

[**2.1.2 Profile Maintenance** 4](#_Toc70718183)

[**2.1.3 Mycroft Data Privacy Set** 6](#_Toc70718184)

[**2.1.4 Maintain Skill** 6](#_Toc70718185)

[**2.1.5 Maintain Device** 7](#_Toc70718186)

[**2.2** **Mycroft Skill Create and Run** 8](#_Toc70718187)

[**2.2.1** **Private Skill Create and Run** 8](#_Toc70718188)

[**2.2.2** **Public Skill Create and Run** 9](#_Toc70718189)

[**3.** **System Installation** 13](#_Toc70718190)

[**3.1** **Mycroft Linux VM Installation** 13](#_Toc70718191)

[**3.1.1** **Linux VM Installation** 13](#_Toc70718192)

[**3.1.2** **Mycroft Linux Download** 14](#_Toc70718193)

[**3.1.3 Pair Device** 16](#_Toc70718194)

[**3.2** **Picroft Installation** 16](#_Toc70718195)

[**3.3** **Mycroft Personal Server Installation and Setup** 19](#_Toc70718196)

[**3.4 Voice assistant for routes optimized parcel delivery Installation** 23](#_Toc70718197)

[**4 Use Case Instruction** 24](#_Toc70718198)

[**4.1 Use Case #1 - Picroft Run Voice assistant for routes optimized parcel delivery API Call Skill** 24](#_Toc70718199)

[**4.2 Use Case #2 - Voice assistant for routes optimized parcel delivery API Host Personal Server** 24](#_Toc70718200)

[**4.3 Use Case #3 - Load Project Skill** 24](#_Toc70718201)

[**4.4 Use Case #4 - DailyQuiz Skill** 25](#_Toc70718202)

[**4.5 Use Case #5 – Test ISS Project** 25](#_Toc70718203)

[**4.5.1 ISS API** 25](#_Toc70718204)

[**4.5.2 mycroft-iss-project Skill** 32](#_Toc70718205)

# **System Overview**

This project is to discover the data privacy handling, architecture, main components of Mycroft, interactions of components to make functionality and experiments on software and hardware tools for it. Main components are Wake Word Detection, Speech to Text (STT), Intent Parser, Text to Speech (TTS), Middleware (Mycroft core, home and API) and Skills. In addition, to set-up Mycroft core in devices like Mycroft Linux and Picroft (Raspberry PI based) followed by skill operation interfacing with Mycroft server and Personal server (setup based on Mycroft server), it brings all components together to fulfil the expected functionalities and experiments and can be treated as infrastructure for future exploration and usage.

Furthermore, our project builds customized skills as use cases which are running against devices (Mycroft Linux and Picroft) interfacing both Mycroft server and Personal server. And we also develop routes optimized parcel delivery system which is a REST API-enabled system to be called by skill to perform designed functions against Mycroft infrastructure.

# **Web Setting and User Guide**

## **Mycroft Home Setting**

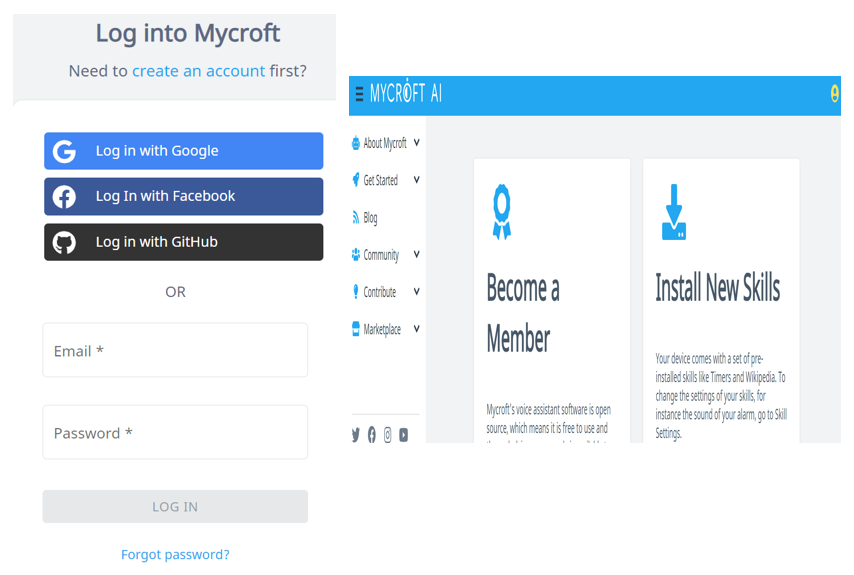
### **2.1.1 Create Account**

Click the **Create Account** button in Download segment of [Get Started Page](https://mycroft.ai/get-started/#download) to create your own Mycroft account. Or you may login with anyone of your Google Account, Facebook Account and GitHub Account.

To login with Mycroft account, you need to input your registered email and password following by clicking the LOG IN button.

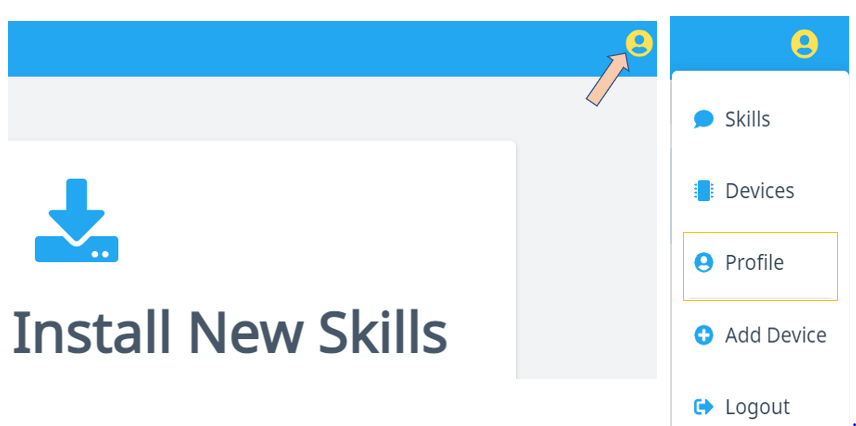
To login with Google/Facebook/GitHub Account, you can select one type of account to click the specific button and then choose an account to continue on Mycroft AI.

After login, you are directing to your Mycroft Home dashboard thereby able to maintain your profile, skills and devices.

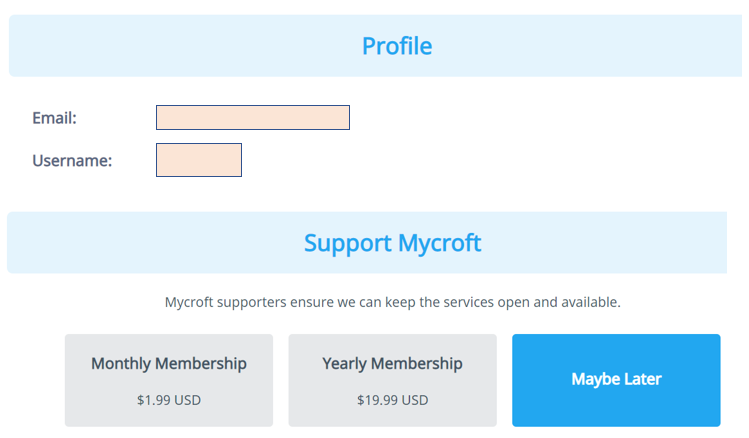


### **2.1.2 Profile Maintenance**

On the dashboard page, click the person icon in the upper right corner followed by clicking the Profile menu.

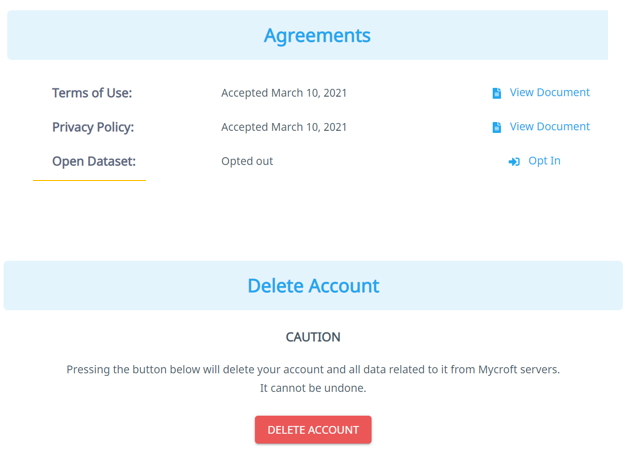


You can view your registered email and username in profile. And you may choose to join membership to support Mycroft.



By clicking the **Opt In** to change, your voice data will be treated as **Open Dataset** which is used to train Mycroft to be more accurate on Speech To Text (STT) function.

To completely remove your Mycroft account, you can click the DELETE ACCOUNT button. But this action can’t be undone.



### **2.1.3 Mycroft Data Privacy Set**

By default, Mycroft is not involving your voice data (Opted out). But you may change to Opt-In Open Dataset therefore Mycroft can use your voice data to do training which targets to improve the STT accuracy.

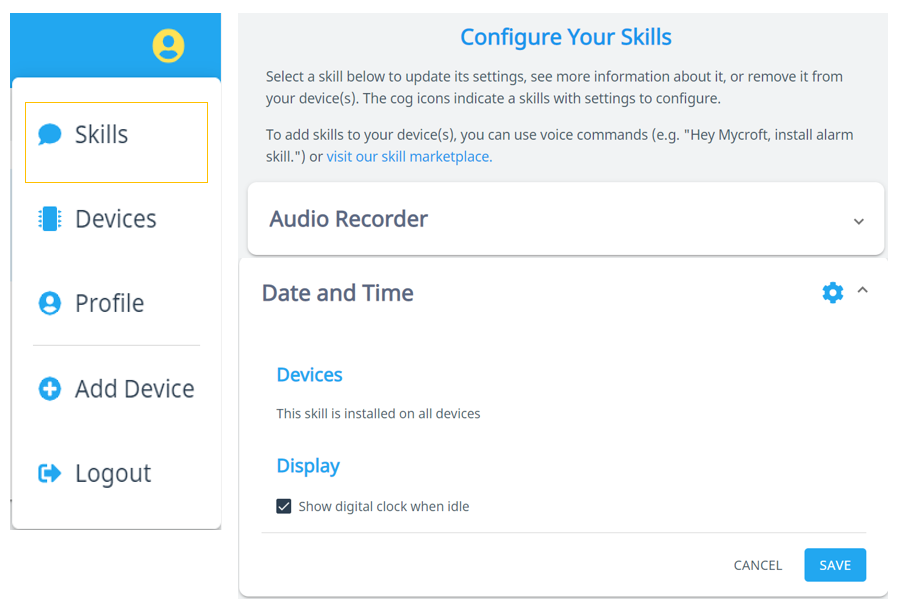
In the Agreements segment of Profile page, you can click the Opt In to contribute Open Dataset.



### **2.1.4 Maintain Skill**

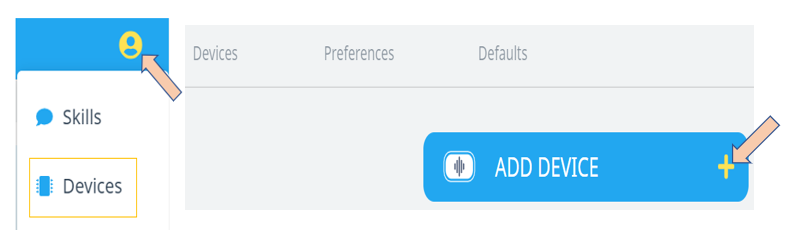
After clicking the Skills button, your skills configuration page shows up which includes all skills downloaded into your device. You can change the parameter of specific skill followed by pushing the SAVE button. New feature will be effective when skills are updated to the device.

To install skill to your device from marketplace, you can use voice commands (e.g., "Hey Mycroft, install alarm skill.")



### **2.1.5 Maintain Device**

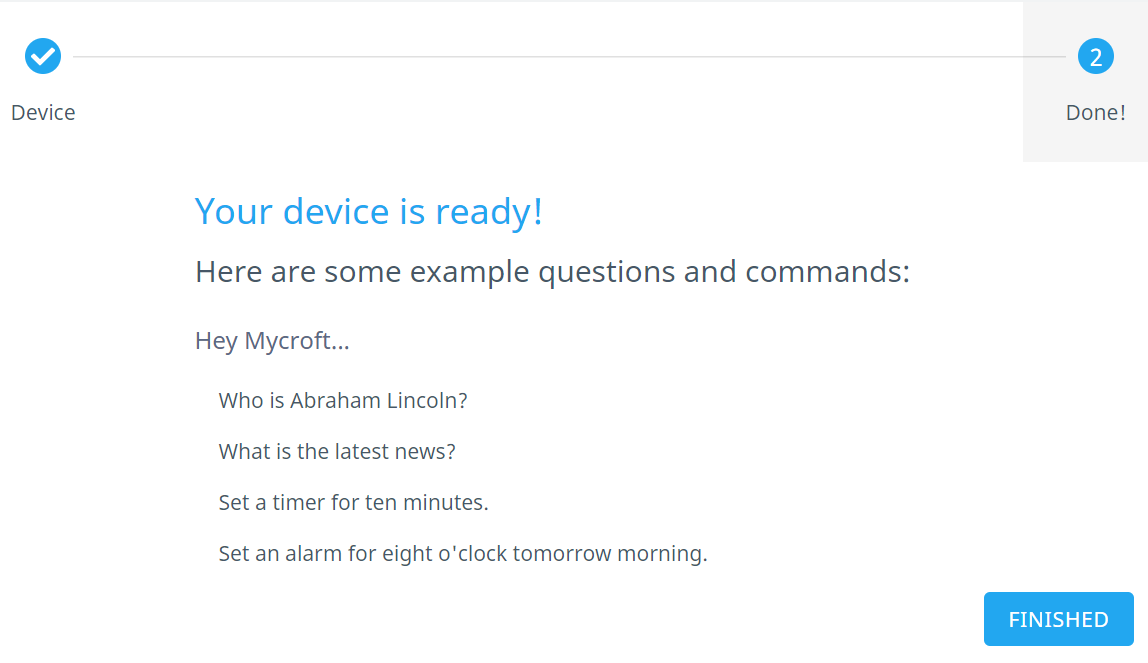
To add a device to your account, you need click the Devices followed by clicking “ADD DEVICE”. Or you can click the Add Device menu to go to Configure new device page.



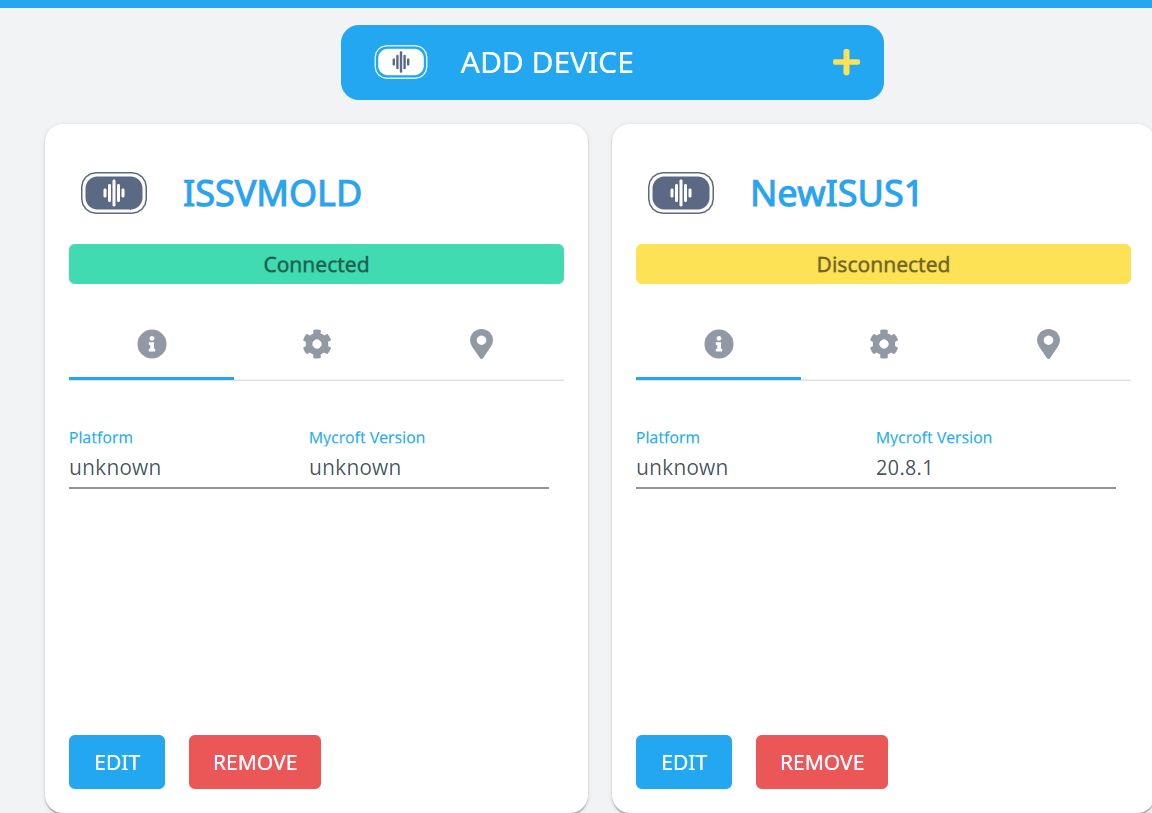
Input the fields in the new device configure page1. Pairing code is getting from history in the CLI window. When all fields are filling up, button NEXT is enabling.



After clicking the NEXT button, you are directed to device configure page2. By clicking the FINISHED button, a new device is added.



New added device is appearing in device list.



## **Mycroft Skill Create and Run**

### **Private Skill Create and Run**

Follow below steps to get private skill to your device:

* cd ~/mycroft-core/skills
* git clone skill-git-repo # skill repo in GitHub

Wait for the skill to be updated in device and configuration ready, you can run this skill in your device. Following the way in **4.3 Use Case #4** to run this private skill.

### **Public Skill Create and Run**

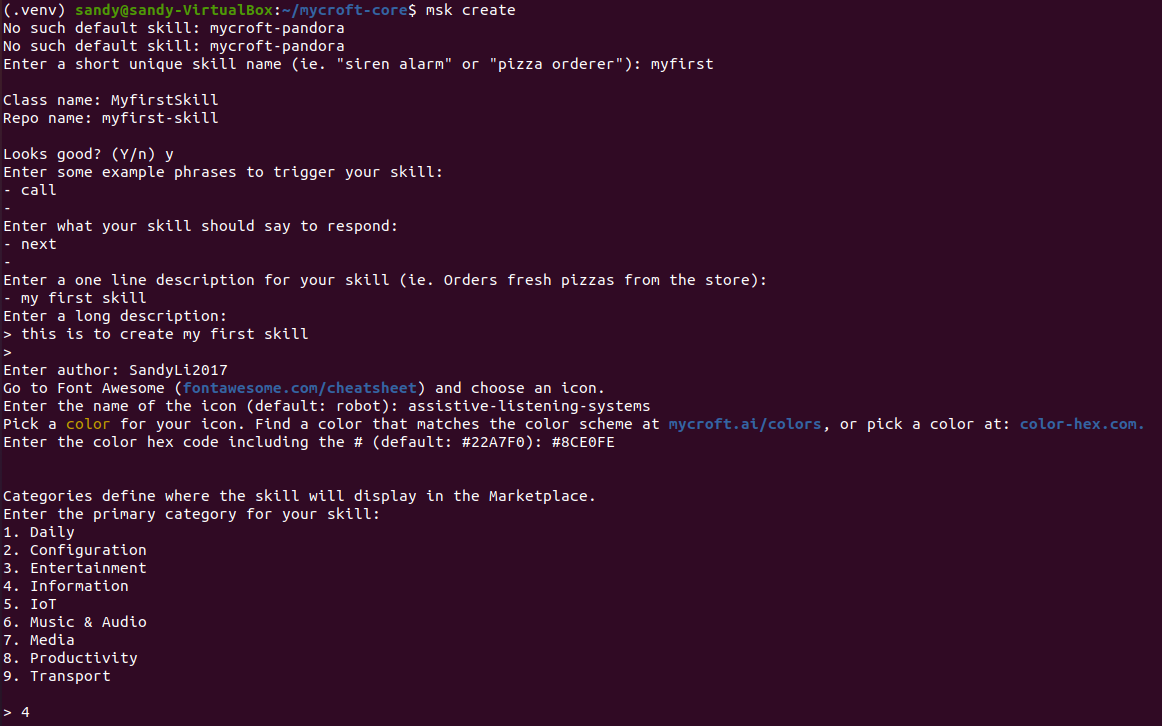
Run below commands:

* cd ~/mycroft-core # or the path to your mycroft-core installation
* source venv-activate.sh; pip install msk

Then you can run msk create to create skill.

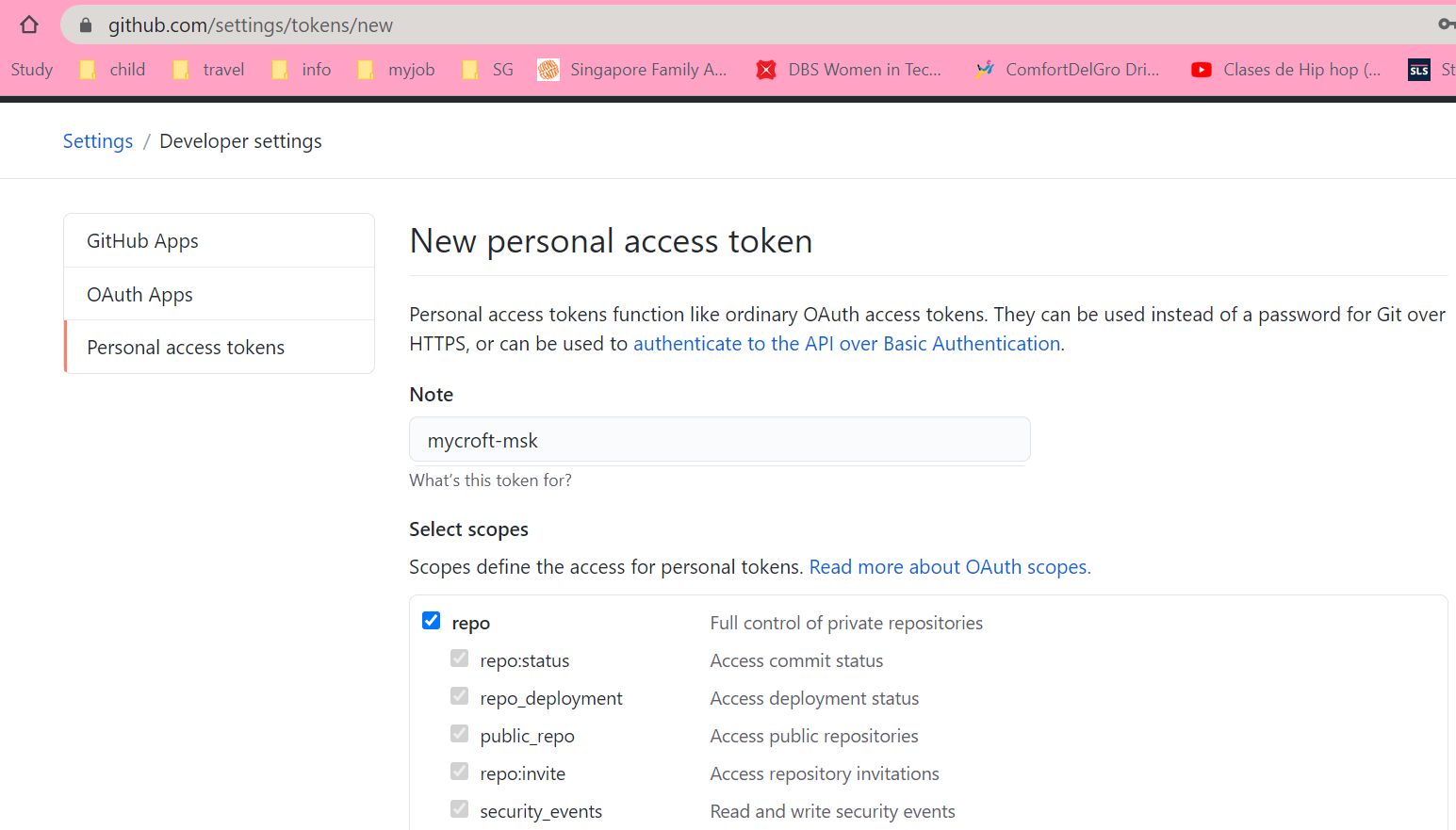
Fill up the fields including:

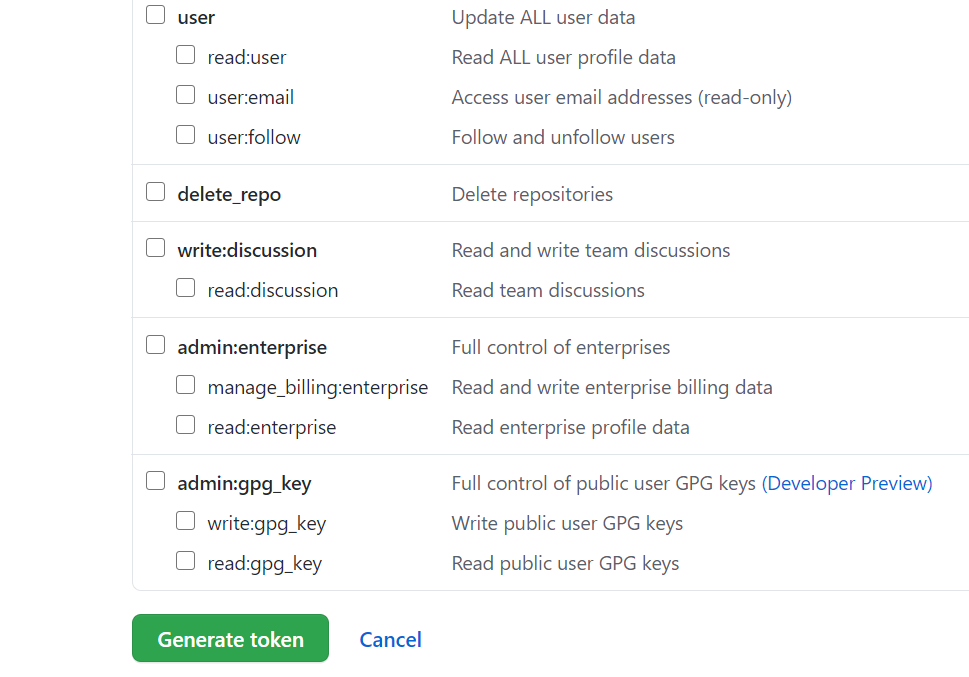
* Skill name which should not be duplicated with other skills in your device
* Utterance
* Response
* Your GitHub account which using to create repo
* Skill description
* Category
* Personal Access Token which you can generate against GitHub and input here
* Etc..

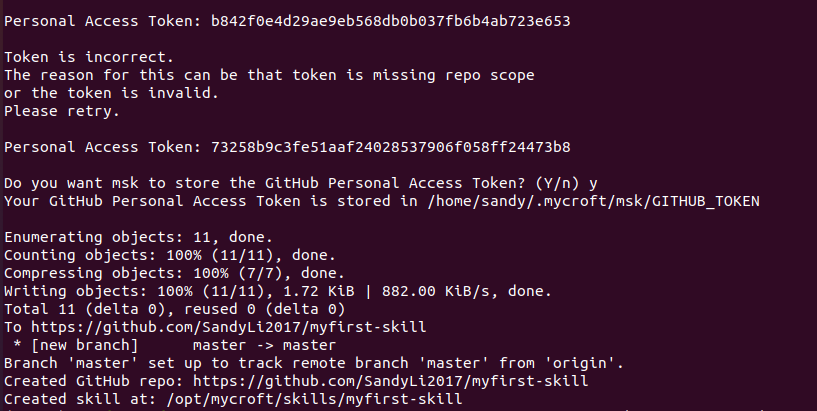




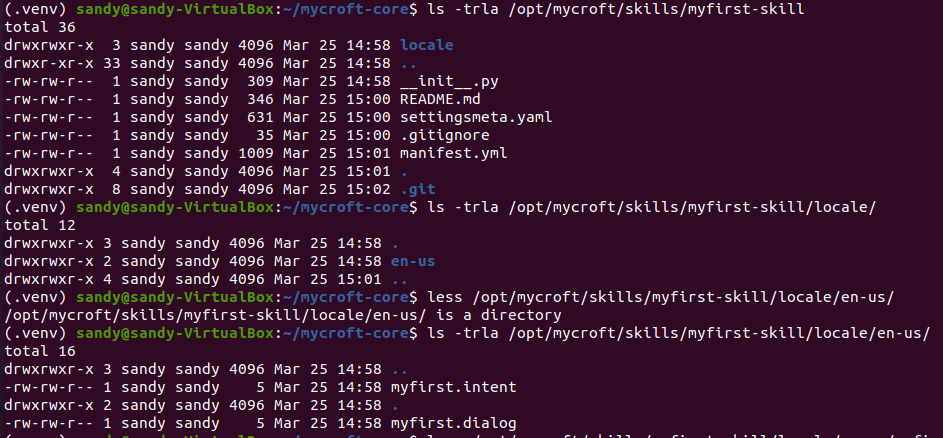
Login your GitHub, in Setting/Developer settings, you can generate token to fill up Personal Access Token field.



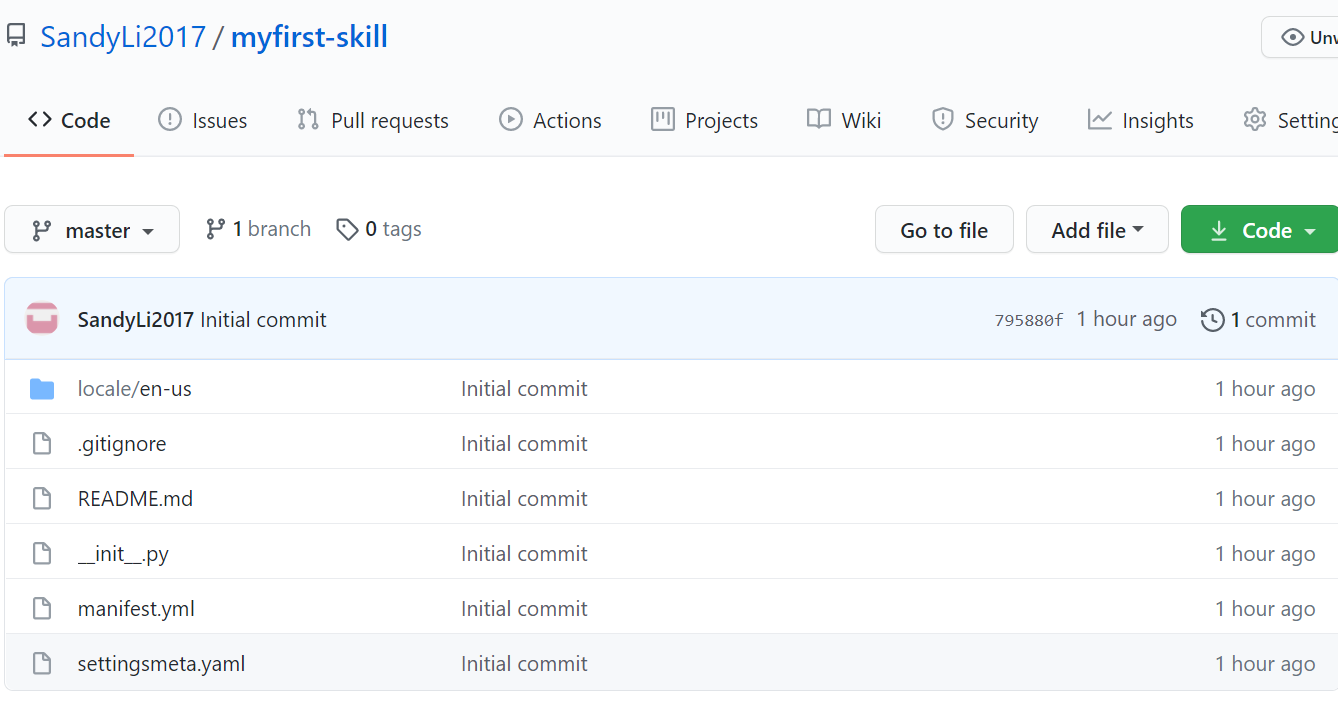


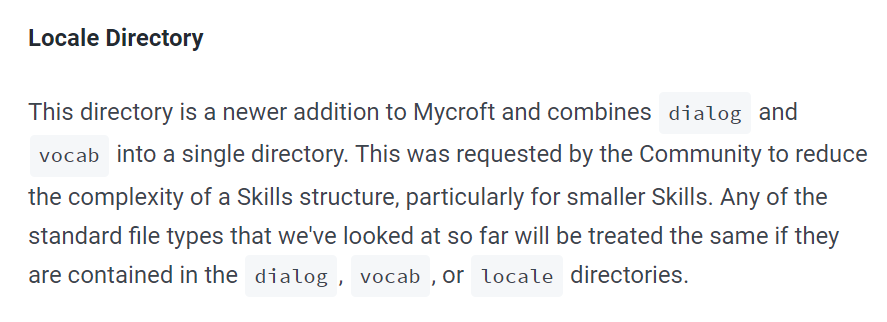


When skill creates successfully, you can find the skill skeleton saved in /opt/mycroft/skills.

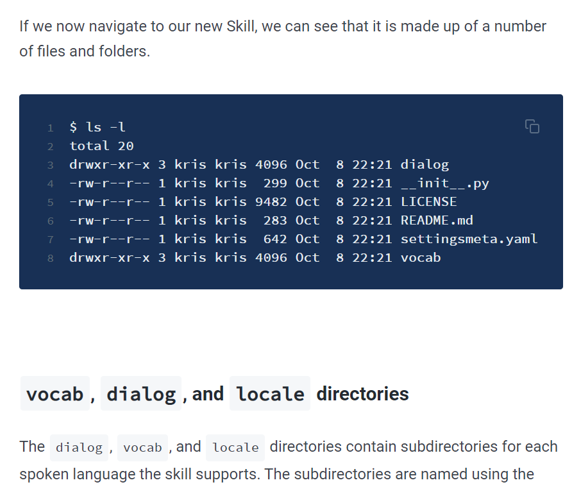


And the same directory hierarchy pushes to your GitHub repo.





Skill created by command “msk create” which is creating locale folder but no more dialog and vocab folders anymore which were using by traditional skills.



# **System Installation**

## **Mycroft Linux VM Installation**

### **Linux VM Installation**

**VM VirtualBox:** Oracle\_VM\_VirtualBox\_6

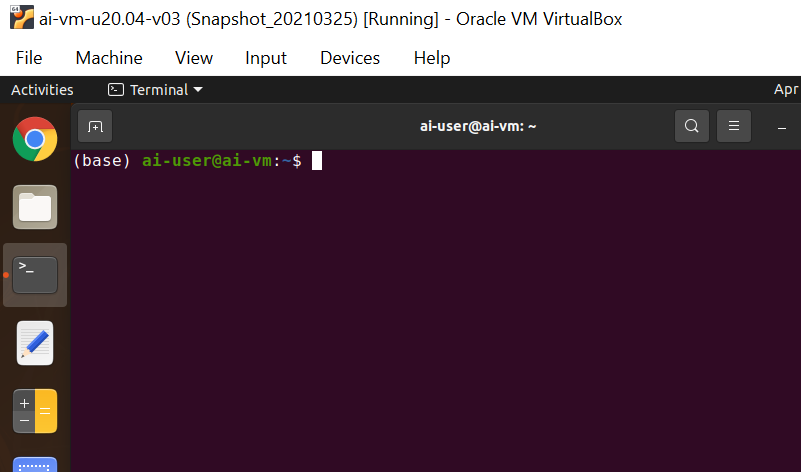
**Linux Image:** Ubuntu 20.04.1

**Project using VM Machine**: ai-vm-u20.04-v03 following the instruction from [telescopeuser/iss-vm](https://github.com/telescopeuser/iss-vm)

When Linux VM installed followed by login, you can open the terminal window to prepare for Mycroft Linux downloading.

Then run below 2 commands:

* sudo apt update
* sudo apt upgrade



### **Mycroft Linux Download**

Click the **Download for Linux** button in Download segment of [Get Started Page](https://mycroft.ai/get-started/#download) to get instructions on how to download Mycroft Linux and run it in the Project VM Machine.

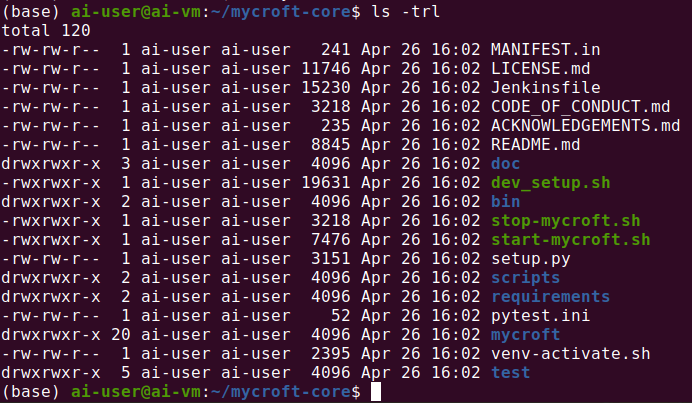
The Instruction link: [MycroftAI/mycroft-core](https://github.com/MycroftAI/mycroft-core)

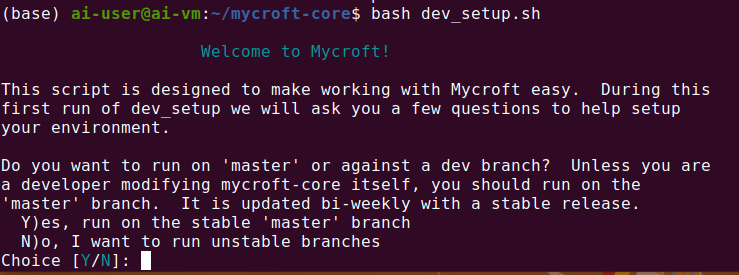
In open terminal window, follow below steps to get code on your device:

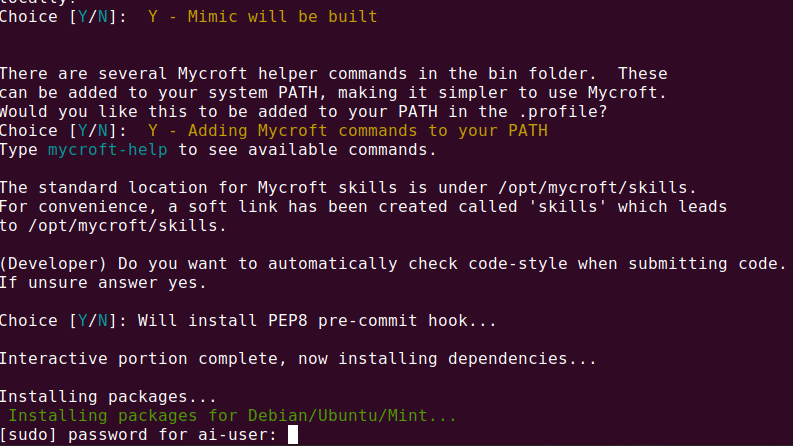
* cd ~/
* git clone <https://github.com/MycroftAI/mycroft-core.git>
* cd mycroft-core
* bash dev\_setup.sh

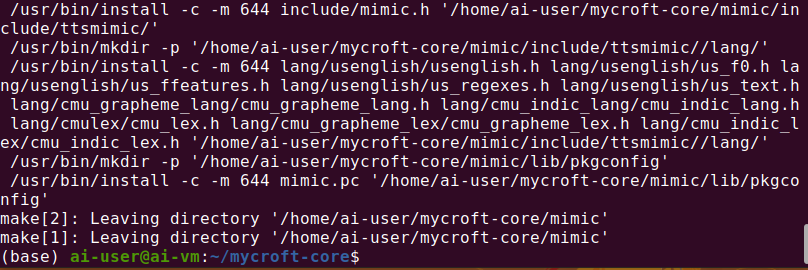
The script dev\_setup.sh sets up dependencies and a [virtualenv](https://virtualenv.pypa.io/en/stable/) to make sure Mycroft Linux running with needed ENV configuration.

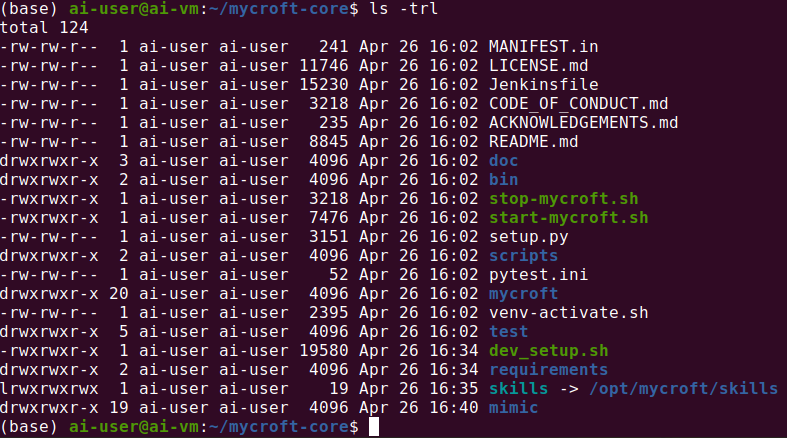










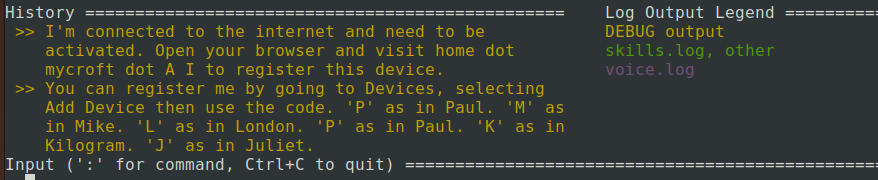


### **3.1.3 Pair Device**

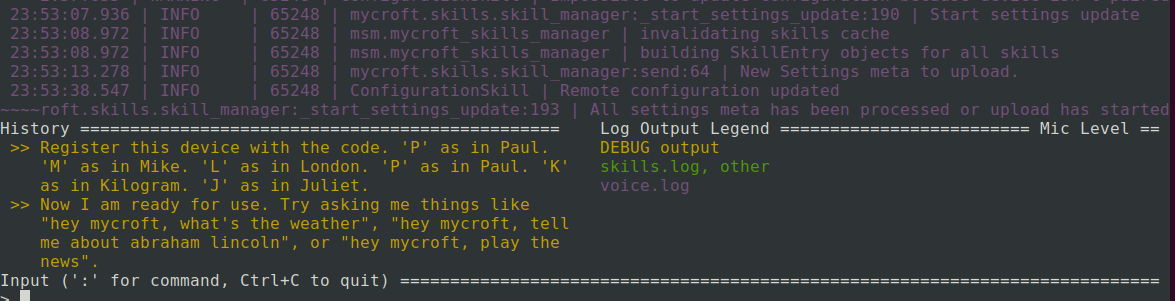
After runningscript dev\_setup.sh in terminal, you can run start-mycroft.sh debug to start your Mycroft device.



For the first time starting your device, there are 6-character Registration Code displaying in CLI which you need to use to finish the pairing against your Mycroft Home page.



Following up 6.1.2 to add the device, your device should pair with your Mycroft. When the wording “Remote configuration updated” appears in the Log Output window, you can start to interact with Mycroft.

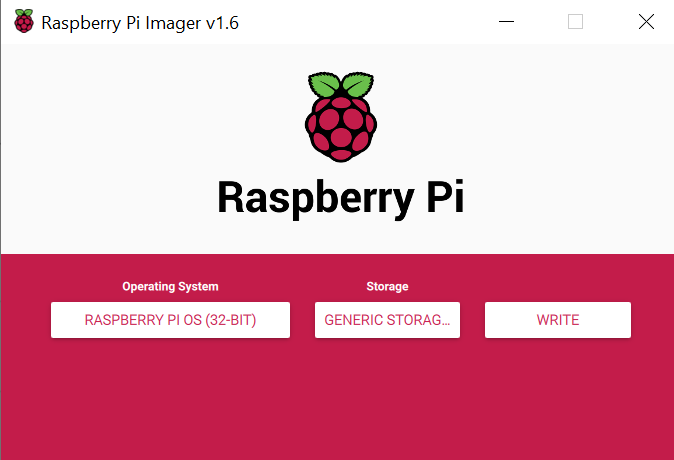
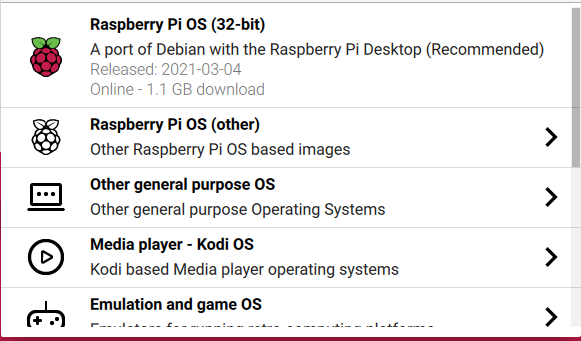


## **Picroft Installation**

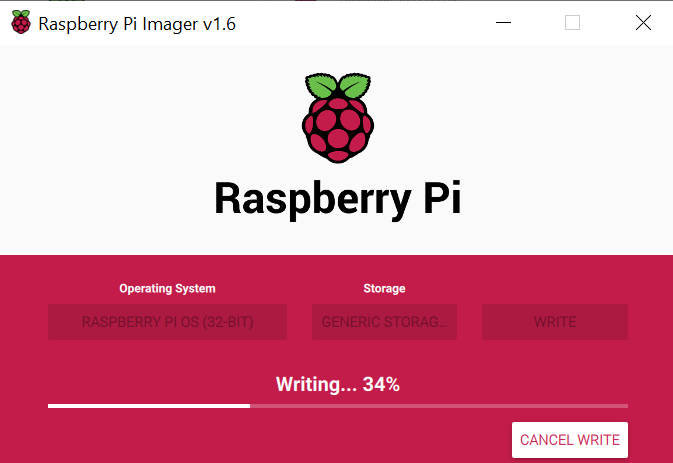
**Raspberry OS setup**

The steps to install Raspberry PI OS are preparation of hardware, download the Raspberry PI Imager and install the OS.

Before the installation, a Raspberry PI 3 Model B (or above) and 16GB micro-SD card are needed. The Raspberry needs to have a HDMI display, USB port to support mouse and keyboard and Ethernet port. Then, download the imager from the Internet and set-up on the desktop with micro-SD slot support. After the imager setup, slot in the micro-SD card, select and run the “Raspberry PI OS” (recommend option).

Once the selection is done, click “Write”. The imager will download the OS version from the internet and write to the SD card.



After the Raspberry PI OS is set-up, slot the SD card into the Raspberry PI and connect the accessories (mouse, keyboard, HDMI to monitor, ethernet cable). Turn on the Raspberry PI, the OS will continue the setup, this will take a couple of minutes.



HDMI

Power



Ethernet

Cable

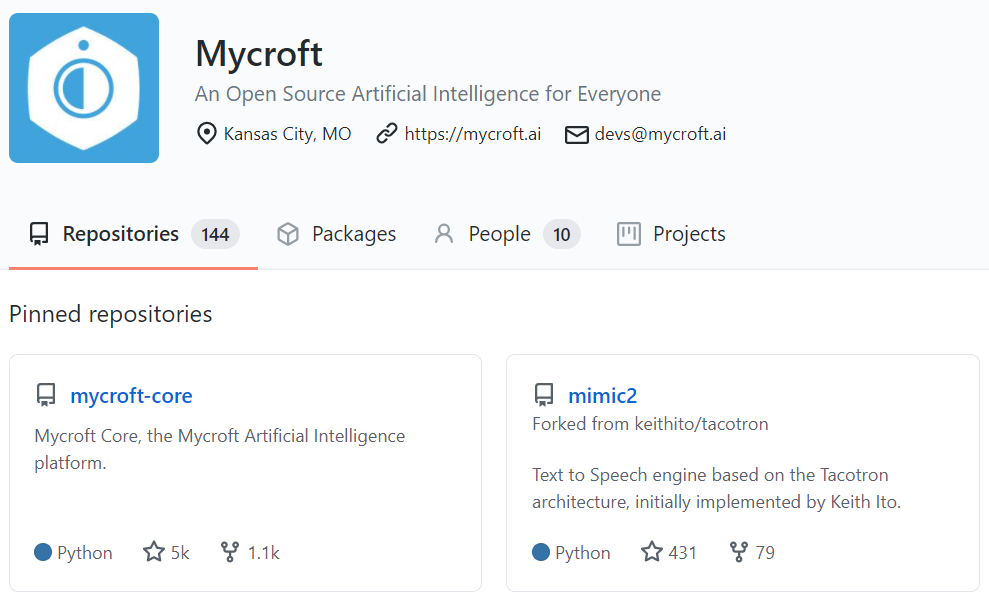
USB keyboard/ mouse/mic

<https://www.raspberrypi.org/blog/raspberry-pi-imager-imaging-utility/>

**Mycroft Core Setup**

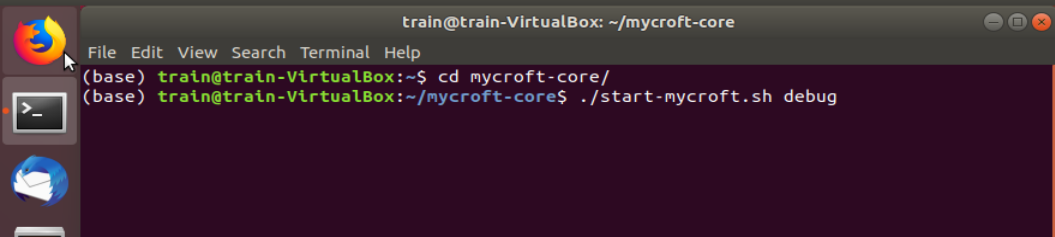
Login to Raspberry with the pre-install OS, the default user is “pi” and password is “raspberry”. The OS should already pre-install Python and Git, proceed to clone the Mycroft-core repository under the command line below:

* cd ~/
* git clone <https://github.com/MycroftAI/mycroft-core>
* cd mycroft-core (go to the “mycroft-core” folder)
* run dev\_setup.sh

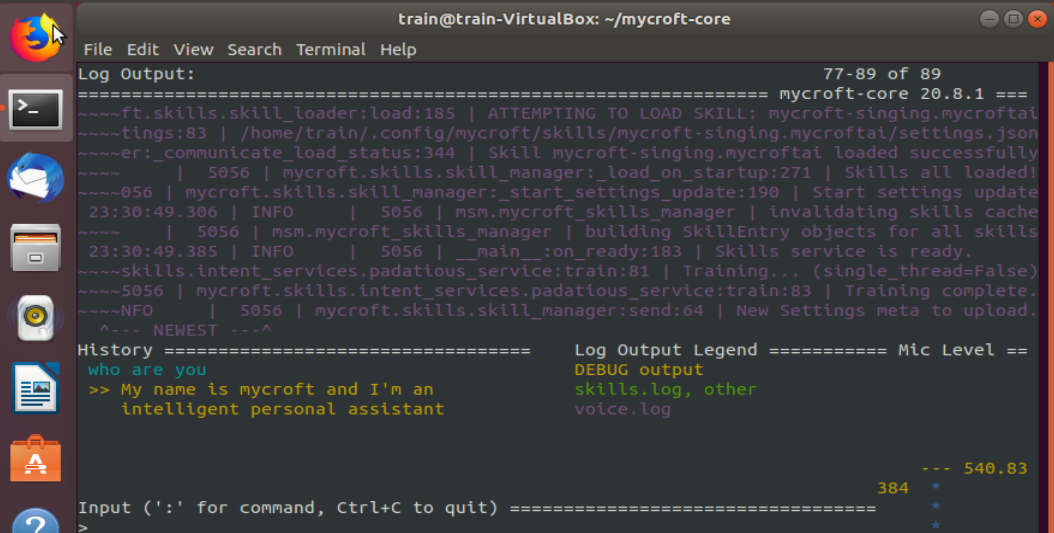


Now the Mycroft-core is installed. Start the Mycroft service.

* cd ~/mycroft-core
* ./start-mycroft.sh debug



To test the core skill, try to talk to Mycroft “**Hey Mycroft, who are you?**”. Below is the response.



For the first time using, you might need to create an account in Mycroft Home, and pair the device referring 2.1.1 and 2.1.5.

## **Mycroft Personal Server Installation and Setup**

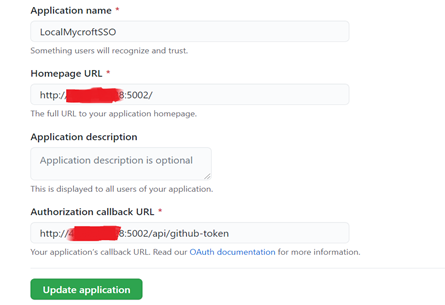
1. For installing Postgres/Redis DB, building database structure and setup Single Sign On API, Account API and Device API, please refer to the link below.

[*https://github.com/MycroftAI/selene-backend*](https://github.com/MycroftAI/selene-backend)

1. Setup GitHub REST API to do basic of authentication

* Go to URL below to register a new OAuth application for SSO Application. It will be assigned a unique Client ID and Client Secret.

[*https://github.com/settings/applications/new*](https://github.com/settings/applications/new)



* Add environment variables containing Client ID, Client Secret, Access Secret, Refresh Secret and Reset Secret.

*export GITHUB\_CLIENT\_ID=<GitHub client id>*

*export GITHUB\_CLIENT\_SECRET=<GitHub client secret>*

*export JWT\_ACCESS\_SECRET=<access secret>*

*export JWT\_REFRESH\_SECRET=<refresh secret>*

*export JWT\_RESET\_SECRET=<reset secret>*

1. Setup the firewall rules on Ubuntu

* Run the command below in terminal

*sudo ufw allow 5000:5100/tcp*

1. Setup Google Cloud Speech-To-Text API

* Follow the link below to get your own API key for Google Cloud Speech-To-Text API

[*http://www.chromium.org/developers/how-tos/api-keys*](http://www.chromium.org/developers/how-tos/api-keys)

* Add API key in environment variable

*export GOOGLE\_STT\_KEY=<Google STT API key>*

1. Perform the steps below to make Mycroft-core connect to the personal selene server

* Download the python code of client setup from GitHub

*cd /opt/selene*

*git clone* [*https://github.com/YYSunshine0912/mycroft-client-setup.git*](https://github.com/YYSunshine0912/mycroft-client-setup.git)

* Access to GitHub REST API with the link below to get GitHub Token via browser.

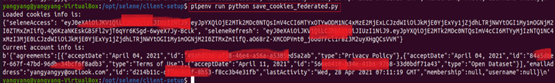
[*https://github.com/login/oauth/authorize?scope=user:email&client\_id=0d6cc5d42fb4158ccaba&state=1*](https://github.com/login/oauth/authorize?scope=user:email&client_id=0d6cc5d42fb4158ccaba&state=1)

* Copy and paste GitHub Token into “github\_token” variable in “save\_cookies\_federated.py” python file



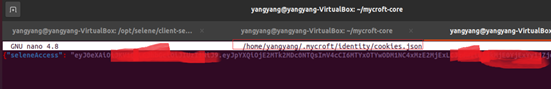
* Run “save\_cookies\_federated.py” python file, it will get JWT access/refresh token from GitHub REST API and store them into the local JSON file after completing GitHub federated authentication.

*pipenv run python save\_cookies\_federated.py*

**

* Run the command below to verify if JWT access/refresh token have been stored into local JSON file

*nano ~/.mycroft/identity/cookies.json*

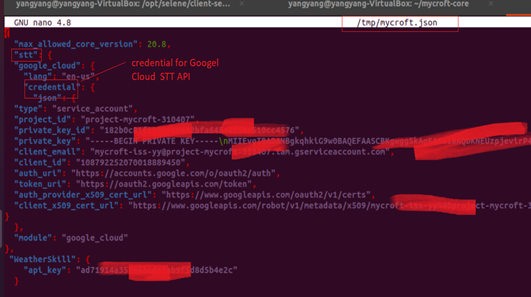
**

* Change IP Address of remote server to point to the personal selene server in Mycroft-Core configuration file.

**

* Run the command below to add credential for the registered Google Cloud STT API in Mycroft-Core configuration file so that Mycroft-Core can perform speech-to-text conversion after connecting to personal selene server

*mycroft-config edit user*

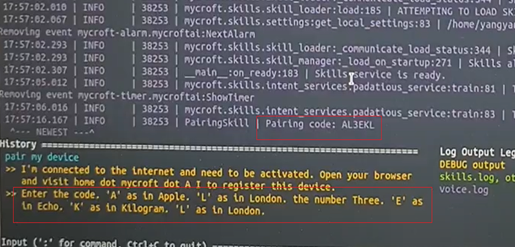
**

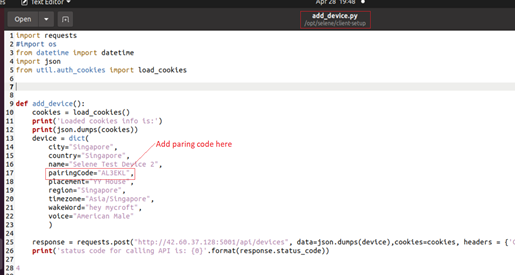
* Start Mycroft-Core to pair device, it will call Device API from personal selene server to get the pairing code

*cd ~/mycroft-core*

*./start-mycroft.sh debug*

* Add the pairing code got from Mycroft-Core in “add\_device.py” python file

**

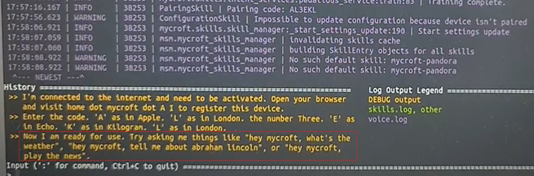
**

* Run “add\_device.py” python file, it will call the Account API to add advice using the saved JWT access/refresh token to pass the authentication.

*pipenv run python add\_device.py*

**

* Mycroft-core can connect to personal selene server and perform speech to text, text to speech and triggering the skill activities

**

## **3.4 Voice assistant for routes optimized parcel delivery Installation**

Step1: Clone the code from <https://github.com/jiangag/vrp>

Step2: Install the packages list inside requirements.txt (pandas, ortools, fastapi, uvicorn, request)

Step3: Run the app.py file

Step4: By default, the API is hosted at <http://127.0.0.1/get_address> and ready to accept API calls

# **4 Use Case Instruction**

## **4.1 Use Case #1 - Picroft Run Voice assistant for routes optimized parcel delivery API Call Skill**

Go to ~/mycroft-core/skills and download the “mycroft-delivery” from GitHub

* cd ~/mycroft-core/skills
* git clone https://github.com/twming/mycroft-delivery.git

Note: You may also run Use Case #3 skill to load the project “mycroft-delivery” from

GitHub

## **4.2 Use Case #2 - Voice assistant for routes optimized parcel delivery API Host Personal Server**

Step1: Clone the code from <https://github.com/jiangag/vrp>

Step2: Install the packages list inside requirements.txt (pandas, ortools, fastapi, uvicorn, request) in the Personal Server

Step3: Run the app.py file

Step4: The API is hosted at [http://x.x.x.x/get\_address](http://127.0.0.1/get_address) and ready to accept API calls from local as well internet (x.x.x.x is Public IP Address of selene server, configurable)

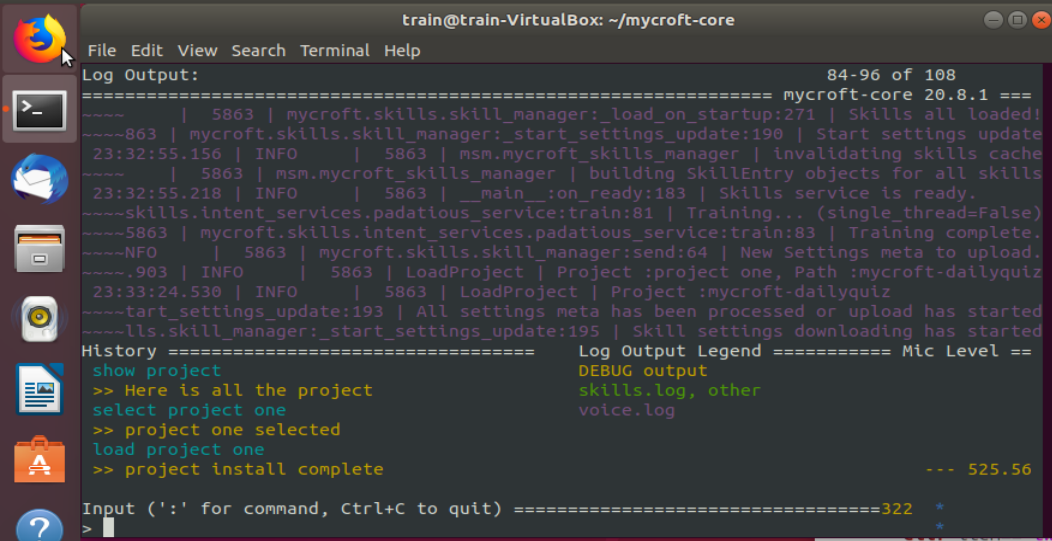
## **4.3 Use Case #3 - Load Project Skill**

Go to ~/mycroft-core/skills and download the load project from GitHub

* cd ~/mycroft-core/skills
* git clone <https://github.com/twming/mycroft-loadproject.git>

After installing the voice assisted project load skill, you could interact with Mycroft to load your selected projects (Daily Quiz, Delivery). The project list is read from the projectlist.txt in the Mycroft-loadproject folder.

* show project
* select project one
* load project/begin project
* remove project



## **4.4 Use Case #4 - DailyQuiz Skill**

Go to ~/mycroft-core/skills and download the “mycroft-dailyquiz” from GitHub

* cd ~/mycroft-core/skills
* git clone https://github.com/twming/mycroft-dailyquiz.git

Note: You may also run Use Case #3 skill to load the project “mycroft-dailyquiz” from GitHub

## **4.5 Use Case #5 – Test ISS Project**

### **4.5.1 ISS API**

ISS API is built using python flask based on the framework of selene. It’s able to host multiple APIs for one project for the different functions. It’s also easy to add new APIs for the new projects. For this use case, we assume there are two ISS group projects to be tested by ISS lectures. The information for the APIs for the two projects is as below.

|  |  |
| --- | --- |
| **Project One Name** | **Iris Demo** |
| **URL** | http://<Public IP Address of selene server:5000>/iss/iris/predict |
| **HTTP Method** | Post |
| **Content-Type** | application/json |
| **Input Parameters** | |  |  |  | | --- | --- | --- | | **S/N** | **Param Name** | **Data Type** | | 1 | sepal\_length | decimal | | 2 | sepal\_width | decimal | | 3 | petal\_length | decimal | | 4 | petal\_width | decimal |   JSON input sample:  {“sepal\_length”:5.5,  “sepal\_width”:3.1,  “petal\_length”:5.8  “petal\_width”:2.0  } |
| **Output Data** | |  |  |  | | --- | --- | --- | | **S/N** | **Output Data** | **Data Type** | | 1 | prediction | string | | 2 | probability | decimal |   JSON output sample:  { “prediction”:”virginica”,  “probability”:0.87  } |
| **Project Path** | The actual project of Iris demo was placed at “ **/opt/selene/selene-backend/shared/iss\_project/project1**” path in personal selene server. |
| **Steps to build API** | 1. Define an endpoint and URL for Iris Demo in ‘api.py’ file at “**/opt/selene/selene-backend/api/iss/iss\_api**” path    2. Create an endpoint named “iss\_project1.py” for Iris Demo at “**/opt/selene/selene-backend/api/iss/iss\_api/endpoints**” path    3. Defile ‘post’ function and ‘\_call\_project\_api’ function in ‘iss\_project1.py’ file. The first entry function of Iris Demo will be called under ‘\_call\_project\_api’ function |
| **Project Two Name** | **Delivery Project** |
| **URL** | http://<Public IP Address of selene server:5000>/iss/delivery/address |
| **HTTP Method** | Post |
| **Content-Type** | application/json |
| **Input Parameters** | |  |  |  | | --- | --- | --- | | **S/N** | **Param Name** | **Data Type** | | 1 | vehicle\_num | int | | 2 | current\_postalCode | int |   JSON input sample:  { “vehicle\_num”:5,  “current\_postalCode”:560324  } |
| **Output Data** | |  |  |  | | --- | --- | --- | | **S/N** | **Output Data** | **Data Type** | | 1 | result | string |   JSON output sample:  { “result”:”there are 5 more locations to go and the next one is 560153”  } |
| **Project Path** | The actual project of Delivery project was placed at “ **/opt/selene/selene-backend/shared/iss\_project/project2**” path in personal selene server. |
| **Steps to build API** | 1. Define an endpoint and URL for Delivery Project in ‘api.py’ file at “**/opt/selene/selene-backend/api/iss/iss\_api**” path  2. Create an endpoint named “iss\_project2.py” for Delivery Project at “**/opt/selene/selene-backend/api/iss/iss\_api/endpoints**” path    3. Defile ‘post’ function and ‘\_call\_project\_api’ function in ‘iss\_project2.py’ file. The first entry function of Delivery Project will be called under ‘\_call\_project\_api’ function |

### **4.5.2 mycroft-iss-project Skill**

Go to ~/mycroft-core/skills and download the “**mycroft-iss-project**” skill from GitHub

*cd ~/mycroft-core/skills*

*git clone https://github.com/YYSunshine0912/mycroft-client-setup.git*

The information for the intended tasks is as below:

|  |  |
| --- | --- |
| **Intended Task One** | **Show ISS Projects Info** |
| **Statements** | The following statements can be used to trigger this intended task  ● ISS project  ● ISS project info  ● How many ISS project(s)  ● How many ISS projects are there  ● How many ISS projects do we have” |
| **Expected Response** | Mycroft Linux or Picroft’s text-to-speech engine will convert the statement below to speech:  “There are two ISS projects. Project one is Iris Demo, Project two is Delivery Demo. |
| **Test Result** |  |
| **Intended Task Two** | **Test Iris Demo** |
| **Statements** | The following statements can be used to trigger this intended task  ● The statement includes “test Iris” key word  ● The statement includes “test project one” key word  ● The statement includes “test demo one” key word |
| **API Involved and Input Parameters** | 1. The URL for the API of Iris Demo is as below:  http://<Public IP Address of selene server:5000>/iss/iris/predict  2. The input parameters for this API are stored in a JSON file named ‘iris\_input.json’ under “input” folder of this Skill folder. The values for these input parameters can be changed so that this intended task is able to get the different results. |
| **Expected Response** | Mycroft-Core or Raspberry PI 3’s text-to-speech engine will convert the statement below to speech:  “The API was called successfully with the returned result is: prediction is {prediction}, probability is {probability}.”  *Note: Variables enclosed by curly brackets are returned data from API* |
| **Test Result** |  |
| **Intended Task Three** | **Test Delivery Project** |
| **Statements** | The following statements can be used to trigger this intended task  ● The statement includes “test delivery” key word  ● The statement includes “test project two” key word  ● The statement includes “test demo two” key word |
| **API Involved And Input Parameters** | 1. The URL for the API of Delivery Project is as below:  http://<Public IP Address of selene server:5000>/iss/delivery/address  2. The input parameters for this API are stored in a JSON file named ‘delivery\_input.json’ under “input” folder of this Skill folder. The values for these input parameters can be changed so that this intended task is able to get the different results. |
| **Expected Response** | Mycroft-Core or Raspberry PI 3’s text-to-speech engine will convert the statement below to speech:  “The API was called successfully with the returned result is: {result}”  *Note: Variables enclosed by curly brackets are returned data from API* |
| **Test Result** |  |