Annexos

1 Coordinador ZigBee

El coordinador ZigBee és el encarregat de controlar tota la xarxa i els camins de comunicació amb tots els dispositius que utilitzen aquest protocol. Ha de existir un dispositiu d'aquest tipus en cada xarxa. Per fer funcionar el coordinador utilitzarem el modul ZigBee de Home Assistant, també anomenat ZHA.

ZHA és una implementació de la biblioteca Zigpy integrada al nucli de Home Assistant. La integració ens permet connectar els dispositius ZigBee disponibles amb algun del mòduls de Zigpy per poder controlar-los. La biblioteca tradueix els protocol de comunicació de diferents fabricants de maquinari per oferir a una API comuna a totes elles. Al ser de codi obert, qualsevol desenvolupador pot afegir compatibilitat en maquinari no suportat.

Nosaltres utilitzarem el xip CC2531 de Texas Instruments com a coordinador, el qual necessitem fer *flashing.* Per poder instal·lar el programari al xip, necessitarem també un CC-Debugger, un petit programador i depurador per a xips de RF de baixa potència com el nostre CC2531. Per a la instal·lació utilitzarem un ordinador amb Linux Mint 20. Hem de connectar el xip al CC-Debuger, i ambdós al ordinador a dos entrades USB.

Segons la documentació oficial necessitarem els següents paquets, que instal·larem així:

```
apt-get install dh-autoreconf libusb-1.0 libboost-all-dev libglib2.0-dev
```

Seguidament cal construir CC-Tool amb:

```
git clone https://github.com/dashesy/cc-tool.git
cd cc-tool
./bootstrap
./configure
make
```

Si la comanda *configure* ens mostra un error amb *Boost headers,* necessitarem també el paquet *build-essential.*

```
apt-get install build-essential
```

Ara necessitarem descarregar Z-Stack-firmware i descomprimir el microprogramari:

```
wget https://github.com/Koenkk/Z-Stack-firmware/raw/master/coordinator/Z-Stack_Home_1.2/bin/default/CC2531_DEFAULT_20190608.zip
unzip CC2531_DEFAULT_20190608.zip
```

Finalment procedim a la instal·lació al xip. L'opció -e serveix per a esborrar la memòria del xip, i l'opció -w és per escriure el arxiu especificat.

```
./cc-tool -e -w CC2531ZNP-Prod.hex
```

Una volta ha finalitzat el procés, ja tenim el xip CC2531 amb el programari necessari per funcionar com a coordinador de la nostra xarxa ZigBee.

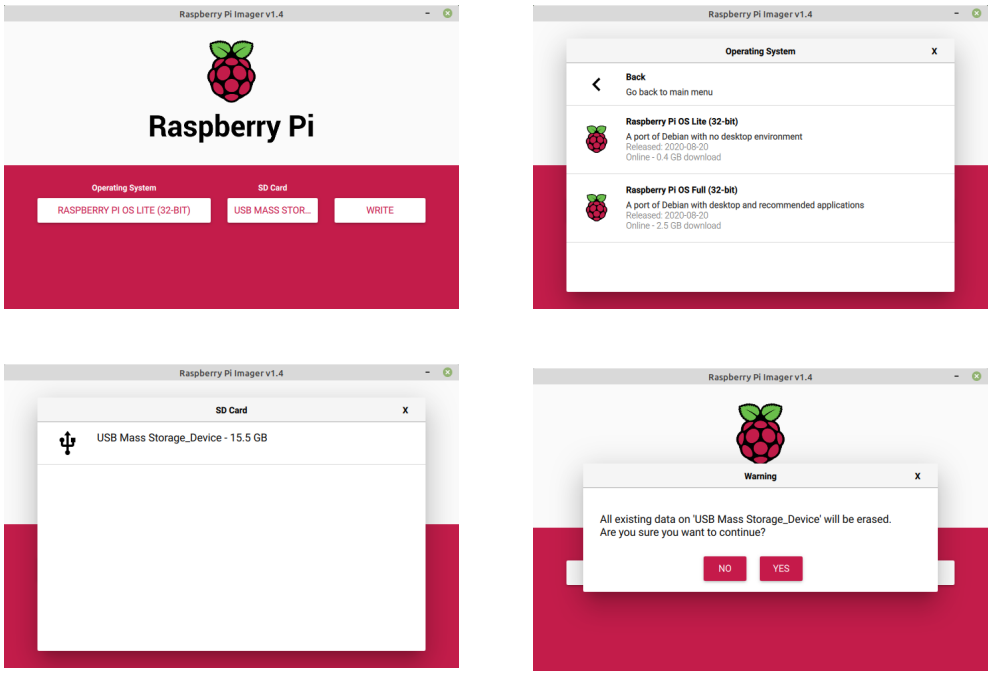
2 Instal·lació de Raspberry Pi OS

A continuació descriurem els passos a seguir per a la instal·lació del sistema operatiu. Hem de preparar tots els elements necessaris com la targeta SD amb un adaptador USB, i també hem d'instal·lar l'eina Raspberry Pi Imager al nostre ordinador personal. Aquesta eina facilita molt la instal·lació perquè descarrega i escriu a la targeta el sistema operatiu desitjat en molts pocs passos.

Per instal·lar l'eina descrita farem ús del terminal al nostre ordinador :

```
sudo apt install rpi-imager
```

Obrim Raspberry Pi Imager i seleccionem el sistema operatiu Raspberry Pi OS Lite (32 bits sense Desktop ni programari recomanat). Seleccionem la targeta SD a la qual voleu escriure la imatge, i polsem en *write* per escriure les dades a la targeta SD. El procediment d'escriptura té una durada d'uns pocs minuts.



D'esquerra a dreta i de dalt a baix.

Finestra principal Raspberry Pi Manager

Finestra selecció de sistema operatiu en Raspberry Pi Manager

Finestra selecció de targeta SD en Raspberry Pi Manager

Finestra confirmació d'escriptura en Raspberry Pi Manager

Una volta finalitza el procés, hem de configurar la connexió a Internet (WIFI) i habilitar el servici SSH. Hem de crear un arxiu específic a l'arrel de la partició *boot* de la targeta SD per habilitar a la arrancada de cadascuna de les dues coses.

Arxius de configuració de xarxa i ssh del SO

wpa_supplicant.conf	ssh
country=es *update_config=1* *ctrl_interface=/var/run/wpa_supplicant* *network={* *scan_ssid=1* *ssid="LaMevaXarxa"* *psk="LaMevaContrasenya"* *}*	*El contingut d'aquest arxiu és indiferent. Només hi ha que crear el arxiu amb en el nom ssh sense extensió.*

Incorporarem la targeta a la nostra Raspberry Pi i encendrem el dispositiu. A la primera arrancada del sistema configurarà la xarxa WIFI i habilitarà el servidor SSH. Posteriorment, s'elimina de forma automàtica els arxius creats i el sistema queda configurat.

Raspberry assigna per defecte al host el nom 'raspberrypi', que serà el nom que utilitzarem per connectar-nos des de el nostre ordinador. El usuari per defecte és 'pi' i la contrasenya és 'raspberry'.

```
ssh pi@raspberrypi
```

De les primeres coses que cal realitzar és actualitzar el sistema operatiu amb els paquets més recents.

```
sudo apt-get update
sudo apt-get upgrade
```

Acte seguit, canviarem el contrasenya del usuari que venia per defecte:

```
pi@raspberrypi:~ $ passwd
S'està canviant la contrasenya de pi.
Current password:
Nova contrasenya de :
Torneu a escriure la nova contrasenya de :
passwd: s'ha actualitzat la contrasenya satisfactòriament
```

Ara forçarem al sistema a demanar la contrasenya de superusuari sempre que es faça ús de la comanda *sudo* per evitar usos indeguts dels permisos d'administrador. Amb un editor canviarem el contingut del arxiu '/etc/sudoers.d/010_pi-nopasswd' per:

```
pi ALL=(ALL) PASSWD: ALL
```

Per a la gestió de Iptables, el nostre tallafocs que ja es troba al sistema per defecte, instal·larem l'eina Uncomplicated Firewall (ufw). Aquesta ferramenta ens ajuda al procés de configuració del tallafocs.

```
sudo apt install ufw
```

Després de la instal·lació, deixarem sols obert el port SSH.

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow ssh
sudo ufw enable
```

Per a la prevenció d'intrusos utilitzarem Fail2ban. Aquesta aplicació escrita en Python detecta i bloqueja intents d'accés per força bruta.

```
sudo apt install fail2ban
```

Primer, farem una copia de la configuració general per afegir-li les modificacions pròpies.

```
sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Deixarem la configuració per defecte a excepció de la entrada *maxretry a* que la reduirem a 3 intents:

```
maxretry = 3
```

Al finalitzar, cal reiniciar el servei

```
sudo /etc/init.d/fail2ban restart
```

Finalment, canviarem el nom del host. Nosaltres li posarem 'tfg'. Necessitem modificar dos arxius: '/etc/hostname', que conté el nom de la maquina, i '/etc/hosts' que associa noms en adreces IP.

Configuració del nom de la màquina.

hostname	hosts
tfg	*...* *127.0.1.1 tfg*

Cal reiniciar i ja tenim el sistema operatiu configurat.

3 Instal·lació de Docker i Portainer

A continuació es detallen les comandes necessàries per instal·lar Docker al nostre dispositiu. Docker és un projecte de codi obert per a la automatització del desplegament d'aplicacions. Així, es proporciona una capa extra d'abstracció i automatitza la virtualització de les aplicacions. Amb Docker aconseguim entorns aïllats per executar els programes sense que interfereixi en el sistema operatiu.

El primer pas serà actualitzar el nostre sistema operatiu:

```
sudo apt update && sudo apt dist-upgrade && sudo apt autoremove
```

Docker proporciona un guions per a la instal·lació en dispositius amb Raspberry OS. Aquests guions detecten la nostra distribució i versió de Linux i configura el nostre sistema. A més, instal·la totes les dependències i recomanacions del gestor de paquets. Altrament, no proporciona cap paràmetre d'instal·lació ni opcions per especificar la versió a instal·lar.

```
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
```

Obtenim aquesta sortida amb detalls de les versions instal·lades:

```
Client: Docker Engine – Community
 Version:           19.03.13
 API version:       1.40
 Go version:        go1.13.15
 Git commit:        4484c46
 Built:             Wed Sep 16 17:07:02 2020
 OS/Arch:           linux/arm
 Experimental:      false

Server: Docker Engine – Community
 Engine:
  Version:          19.03.13
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.15
  Git commit:       4484c46
  Built:            Wed Sep 16 17:00:52 2020
  OS/Arch:          linux/arm
  Experimental:     false
 containerd:
  Version:          1.3.7
  GitCommit:        8fba4e9a7d01810a393d5d25a3621dc101981175
 runc:
  Version:          1.0.0-rc10
  GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
```

El següent pas es incloure al nostre usuari dintre dels usuaris de Docker, al nostre cas 'pi', perquè no hi hagi problemes de permisos i de gestió.

```
sudo usermod -a -G docker pi
```

Finalment, reiniciem el sistema perquè s'apliquen els canvis.

Arribat a aquest punt, ja tenim operatiu Docker i podem afegir-li contenidors. El primer que crearem és Portainer. Aquest contenidor simplifica la gestió de Docker amb una interfície gràfica web molt fàcil d'utilitzar. Container necessita d'un volum on consumir i emmagatzemar dades que hem de crear en primer lloc. I després ja podem crear el contenidor on especifiquem el volum creat i el port amb el que escoltarà la interfície d'usuari.

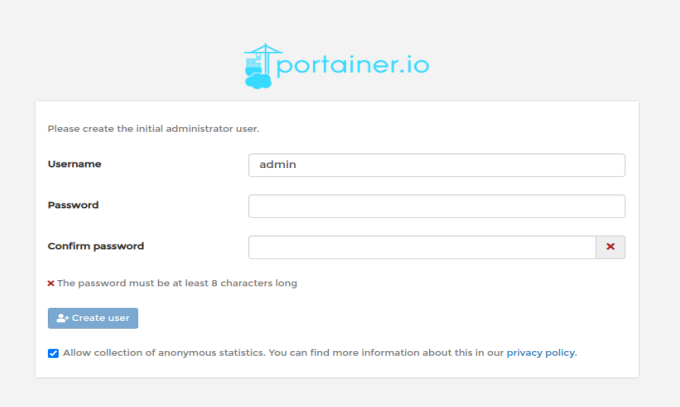
```
sudo docker volume create portainer_data

docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -v
        /var/run/docker.sock:/var/run/docker.sock -v
        portainer_data:/data portainer/portainer-ce
```

Obtenim aquesta sortida i el contenidor ja està creat:

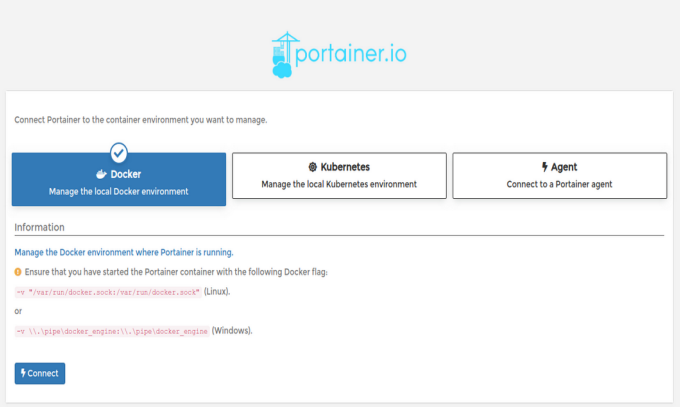
```
Unable to find image 'portainer/portainer-ce:latest' locally
latest: Pulling from portainer/portainer-ce
d1e017099d17: Pull complete
eccff216cb75: Pull complete
Digest: sha256:0ab9d25e9ac7b663a51afc6853875b2055d8812fcaf677d0013eba32d0bf0e0d
Status: Downloaded newer image for portainer/portainer-ce:latest
d3400fc9c1dd65cd2bb977eb1ce7913d4a2242b9af24d5bfea23ba141845769c
```

Cal reiniciar i haurem de configurar Portainer. Per a la configuració cal accedir amb un navegador a la direcció 'http://tfg:9000/' on ens sol·licitarà la creació d'un nou usuari per gestionar la ferramenta.



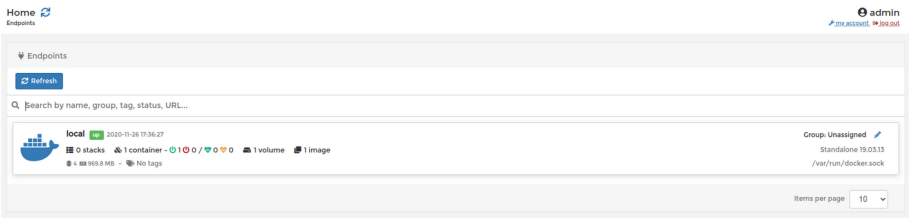
Finestra de Portainer per a la creació de nou usuari

A continuació ens demanarà vincular Portainer a Docker. Ho seleccionem i polsem en connectar.

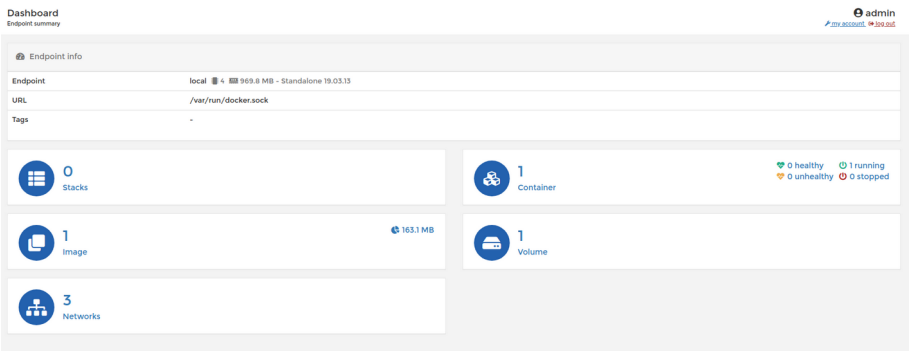


Finestra de connexió de Portainer amb Docker

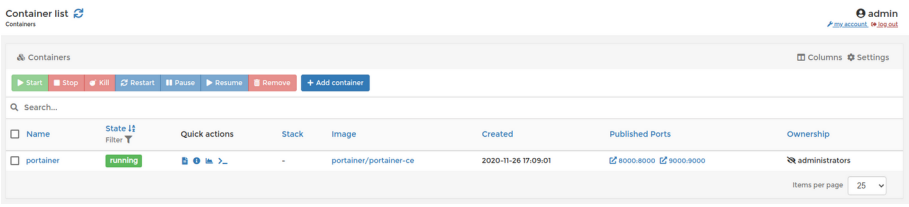
Arribat aquest punt ja tenim configurat Portainer. A partir d'ara ja podem gestionar els contenidor des de la interfície web.



Finestra principal de Portainer



Panell de Docker en Portainer



Panell de contenidors en Portainer

4 Instal·lació de Home Assistant

La versió que instal·larem és Home Assistant Supervised. Aquesta versió incorpora tot els components de Home Assistant a excepció del sistema operatiu. El mòdul supervisor és un dispositiu que gestiona tot el sistema, amb capacitat de netejar-se, reparar-se o restablir-se si fos necessari.

Per a la instal·lació necessitem del sistema operatiu seleccionat i de Docker operatius. El primer que farem, serà crear una carpeta que contindrà totes les dades del controlador domòtic. Al nostre cas, crearem la carpeta al nostre espai d'usuari personal.

```
mkdir -m 777 /home/pi/Docker/HA
```

A continuació descarregarem el guió d'instal·lació del sistema i el executarem. Hem d'especificar el següents arguments al guió: -m per a la maquinari on s'instal·la el programari i -d per especificar la ruta de la carpeta de dades creat anteriorment. Per a la execució necessitarem de permisos de superusuari.

```
$:sudo su
$:curl -Lo installer.sh
'https://raw.githubusercontent.com/home-assistant/supervised-installer/master/
installer.sh'
$:bash installer.sh -m raspberrypi3 -d /home/pi/Docker/HA/
```

**Si s'experimenten problemes a la instal·lació del guió amb connexió sense fils, es recomana la instal·lació de Home Assistant amb cable ethernet, ja que durant el procés es reinicia diverses voltes les connexions.*

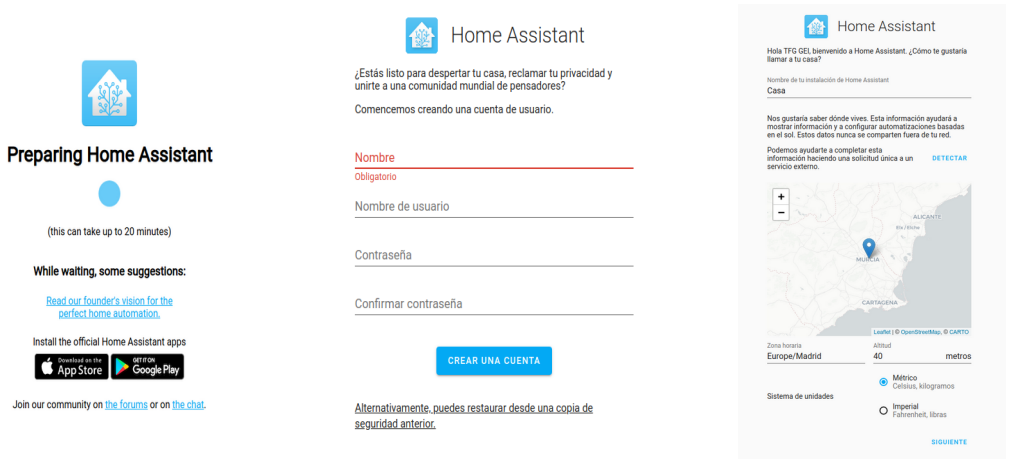
Arribat a aquest punt, començarà a instal·lar-se la resta de contenidors de forma automàtica. Home Assistant es forma amb 7 contenidors en total i que podem consultar amb la següent comanda:

```
sudo docker ps -a
```

```
CONTAINER ID        IMAGE                                                COMMAND                  CREATED
1a4bfb80e8d2        homeassistant/armv7-hassio-observer:2020.10.1        "/init"                  4 minutes ago
2cf81805e5a8        homeassistant/raspberrypi3-homeassistant:landingpage "/init"                  16 minutes ago
2b83d8004008        homeassistant/armv7-hassio-multicast:3               "/init"                  16 minutes ago
44b9a48e7d28        homeassistant/armv7-hassio-cli:2020.10.1             "/init /bin/bash -c …"   18 minutes ago
4eaffbe636bb        homeassistant/armv7-hassio-audio:17                  "/init"                  19 minutes ago
ec04fc4f08e7        homeassistant/armv7-hassio-dns:2020.11.0             "/init"                  20 minutes ago
64df01fed76e        homeassistant/armv7-hassio-supervisor                "/init"                  21 minutes ago
d3400fc9c1dd        portainer/portainer-ce                               "/portainer"             5 days ago
```

Llistat de contenidors de Home Assistant

Una volta s'han creat tots els contenidors, ja podem accedir amb un navegador indicant la ip o nom del host i el port 8123. Nosaltres utilitzarem *'http://tfg:8123/'*. La instal·lació té una durada molt llarga, i una volta finalitza apareix una pagina per a la creació del usuari del controlador domèstic. El següent pas és assignar-li un nom a la plataforma i definir la zona horària i el sistema de unitats. Aquesta configuració podrem modificar-la posteriorment amb els arxius de configuració.



D'esquerra a dreta. Finestra d'instal·lació. Finestra de creació d'usuari. Finestra de configuració inicial

Arribat aquí, ja tenim el sistema instal·lat i aconsellem reiniciar el dispositiu. Aquesta és la informació resumida de Home Assistant:

System Health

version: 0.118.4
installation_type: Home Assistant Supervised
dev: false
hassio: true
docker: true
virtualenv: false
python_version: 3.8.6
os_name: Linux
os_version: 5.4.72-v7+
arch: armv7l
timezone: Europe/Madrid

Hass.io

host_os: Raspbian GNU/Linux 10 (buster)
update_channel: stable
supervisor_version: 2020.11.0
docker_version: 19.03.13
disk_total: 13.9 GB
disk_used: 3.8 GB
healthy: true
supported: failed to load: Unsupported
supervisor_api: ok
version_api: ok
installed_addons:

5 Instal·lació i configuració inicial de Rhasspy

Per a la instal·lació de Rhasspy hem de preparar el directori on es guardarà els nostre perfil, la qual estarà dintre del directori personal.

```
mkdir -m 777 /home/pi/Docker/RH
```

La instal·lació és molt senzilla, i únicament hem d'iniciar la imatge del Docker amb la comanda següent

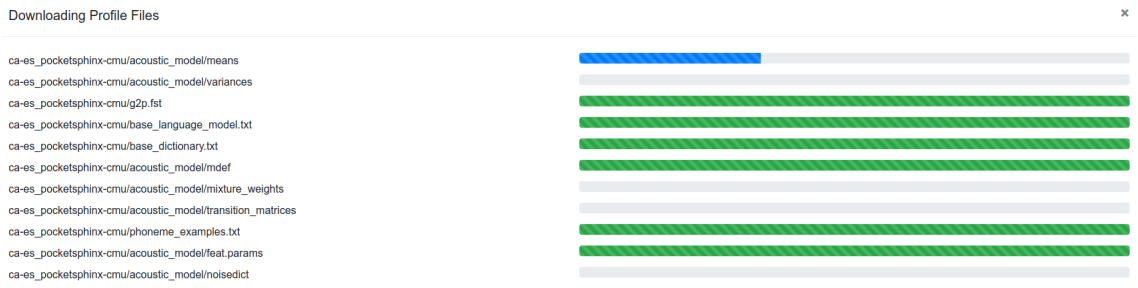
```
docker run -d -p 12101:12101 \
    --name rhasspy \
    --restart unless-stopped \
    -v "$HOME/Docker/RH/rhasspy/profiles:/profiles" \
    -v "/etc/localtime:/etc/localtime:ro" \
    --device /dev/snd:/dev/snd \
    rhasspy/rhasspy \
    --user-profiles /profiles \
    --profile ca
```

Cal destacar que hem especificat la ruta creada anteriorment i hem configurat el perfil amb el idioma català. Una volta iniciada la imatge, podem accedir a la interfície web amb el port 12101.

El primer que farem, serà accedir a la secció de configuració. Inicialment, cap component és troba habilitat, i per tant, hem d'habilitar els components que desitgem. Com hem explicat a la secció '10 Rhasspy', aquest són els serveis seleccionats:

- Enregistrament i reproducció d'àudio: Arecord i Aplay
- Paraula vigília: Pocketsphinx
- Transcripció d'ordres de veu: Pocketsphinx
- Reconeixement d'intencions: Fsticuffs
- Conversió de text a veu: Espeak

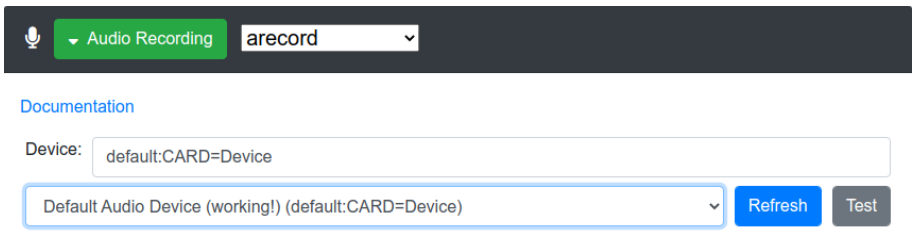
Una volta seleccionats hem de guardar la configuració i reiniciar el programari Rhasspy. Al reiniciar-se ens notificarà que és necessari descarregar uns arxius, que són els models acústics necessaris per a Pocketsphinx. Quan la descarrega està completada, el sistema es reinicia de nou i procedim a la configuració.



Finestra de descarrega dels perfils de Rhasspy

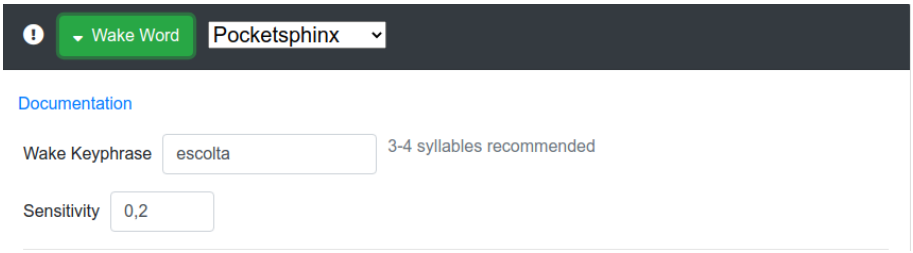
A continuació es necessari la configuració d'alguns dels components:

- Enregistrament d'àudio: Hem de seleccionar el nostre dispositiu de la llista de disponibles. Hi ha l'opció de testejar automàticament els micròfons i apareixerà la paraula *working* als dispositius operatius.



Finestra de selecció d'enregistrador d'àudio

- Paraula vigília: Hem configurat la paraula 'escolta' com a vigília, i l'hem configurat amb una sensibilitat de 0,2 després de diverses proves.

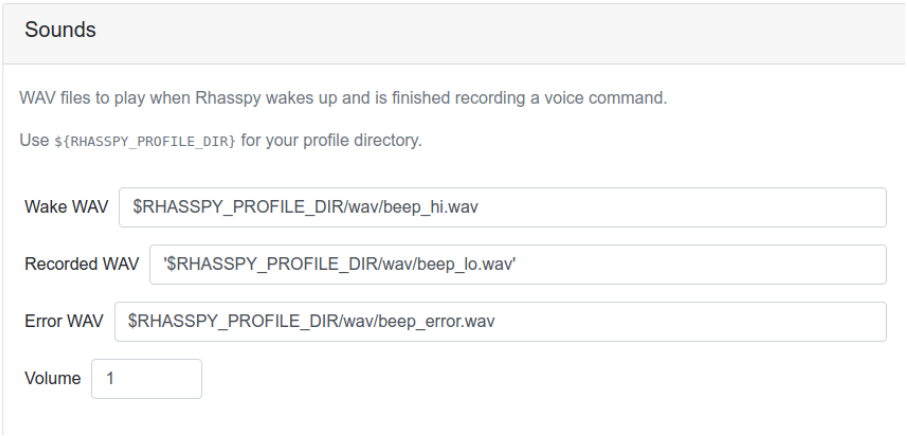


Finestra de selecció de paraula vigília

- Manipulació d'intencions: per a la connexió amb Home Assistant es necessari un testimoni d'accés i la direcció IP del servidor.
- El gestor de diàlegs ha d'estar activat amb el component Rhasspy perquè gestione les sessions iniciades per la paraula vigília.

- La resta de components es quedaran amb la configuració per defecte.

A la secció inferior, es possible personalitzar els sons que emetrà Rhasspy quan desperte i escolta les ordres. Per facilitar la configuració hem copiat els arxius de so necessaris a la carpeta del perfil i utilitzat la variable '$RHASSPY_PROFILE_DIR' per accedir-hi.



Configuració del sons de Rhasspy

També hem modificat la velocitat de la reproducció de so de Espeak que consideràvem ràpida, i l'hem reduït de 180 a 150.

```
"text_to_speech": {
      "espeak": {
         "arguments": [
            "-s",
            "150"
         ]
      },
      "system": "espeak"
   },
```

Per finalitzar, guardem els canvis i reiniciem.

6 Arxius de Home Assistant

configuration.yaml

```
default_config:

http:
   ssl_certificate: /ssl/fullchain.pem
   ssl_key: /ssl/privkey.pem

homeassistant:
   customize: !include customize.yaml
   latitude: !secret home_latitude
   longitude: !secret home_longitude
   elevation: 50
   unit_system: metric
   time_zone: Europe/Madrid
   name: Dani TFG

zone:
  - name: Home
    latitude: !secret home_latitude
    longitude: !secret home_longitude
    radius: 10

device_tracker:
  - platform: bluetooth_tracker
    interval_seconds: 60
    new_device_defaults:
      track_new_devices: false

person:
  - name: Daniel
    id: dani_person
    device_trackers:
      - device_tracker.huawei_p_smart
      - device_tracker.dani_rock

sensor:
  - platform: template
    sensors:
      prediccio:
        value_template: "no hi ha dades"


  - platform: time_date
    display_options:
      - 'time'
      - 'date'
python_script:
intent:

intent_script: !include intent_script.yaml

group: !include groups.yaml
automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml
```

automations.yaml

```
- id: escalfador_on
  alias: escalfador_on
  description: Encendre escalfador d'aigua a mitjanit
  trigger:
  - platform: time
    at: 00:00
  action:
  - service: homeassistant.turn_on
    entity_id: switch.endoll
  mode: single

- id: escalfador_off
  alias: escalfador_off
  description: Apagar escalfador d'aigua al matí
  trigger:
  - platform: time
    at: 08:00
  action:
  - service: homeassistant.turn_off
    entity_id: switch.endoll
  mode: single

- id: es_fa_de_nit
  alias: es_fa_de_nit
  mode: single
  description: encendre els llums al despatx quan es fa de nit
  trigger:
  - platform: sun
    event: sunset
    offset: -00:25:00
  condition:
  - condition: time
    weekday:
    - mon
    - tue
    - wed
    - thu
    - fri
  - condition: state
    state: home
    entity_id: device_tracker.dani_rock
  action:
  - scene: scene.treball
```

scenes.yaml

```
  - name: treball
    entities:
      light.llum_despatx:
        state: true
        brightness: 254
  - name: lectura
    entities:
      light.llum_despatx:
        state: true
        brightness: 178
```

intent_script.yaml

```
  GetTemperatureInt:
    speech:
      text: Tenim {{ states.sensor.sonoff_temperature.state }} graus i una humitat de
{{ states.sensor.sonoff_humidity.state }} per cent.

  GetTemperatureExt:
    speech:
      text: >
        {%- if states.weather.casa.state == "sunny" -%}Avui està  assolellat
        {%- elif states.weather.casa.state == "cloudy" -%}Avui està ennuvolat
        {%- elif states.weather.casa.state == "clear-night" -%}Tenim nit clara
        {%- elif states.weather.casa.state == "fog" -%}Avui tenim boira
        {%- elif states.weather.casa.state == "hail" -%}Avui tenim granís
        {%- elif states.weather.casa.state == "lightning" -%}Avui tenim llamps
        {%- elif states.weather.casa.state == "lightning-rainy" -%}Avui tenim llamps i plutja
        {%- elif states.weather.casa.state == "partlycloudy" -%}Avui està parcialment ennuvolat
        {%- elif states.weather.casa.state == "pouring" -%}Avui tenim pluges torrencials
        {%- elif states.weather.casa.state == "rainy" -%}Avui plou
        {%- elif states.weather.casa.state == "snowy" -%}Avui neva
        {%- elif states.weather.casa.state == "snowy-rainy" -%}Avui plou i neva
        {%- elif states.weather.casa.state == "windy" -%}Avui fa vent
        {%- elif states.weather.casa.state == "windy-variant" -%}Avui fa vent
        {%- elif states.weather.casa.state == "exceptional" -%}Temps desconegut
        {%- endif -%}
        . Tenim {{ states.weather.casa.attributes.temperature }} graus, una humitat de
{{ states.weather.casa.attributes.humidity }} per cent i una velocitat del vent de
{{ states.weather.casa.attributes.wind_speed }} metres per segon

  ChangeLightStateDespatx:
    action:
    - service_template: >
        {%- if light_state == 'Encén' -%}
          homeassistant.turn_on
        {%- else -%}
          homeassistant.turn_off
        {%- endif -%}
      data_template:
        entity_id: light.llum_despatx
    speech:
      text: >
        Fet. {{ light_state }} despatx

  ActiveScene:
    action:
      - service: scene.turn_on
        data_template:
          entity_id: >
            scene.{{escena}}

  Prediccio:
    action:
      - service: python_script.prediccio
    speech:
      text: >
       {%- if states.sensor.prediccio == "no hi ha dades" -%} No hi ha dades
        {%- else -%} {{ states.sensor.prediccio.attributes }}
        {%- endif -%}
```

```
  PrediccioDia:
    action:
      - service: python_script.prediccio
    speech:
      text: >
       {%- if state_attr('sensor.prediccio', dia)!= None -%}}
         {{ state_attr('sensor.prediccio', dia) }} .
       {%- else -%}
         No hi ha predicció
       {%- endif -%}

  GetTime:
    speech:
      text: Ara són les {{ states.sensor.time.state }}
```

prediccio.py

```
  d = {"sunny": "estarà asolejat",
       "partlycloudy": "estarà parcialment ennuvolat",
       "cloudy":"estarà ennuvolat",
       "clear-night":"tindrem nit clara",
       "fog": "tindrem boira",
       "hail": "tindrem granís",
       "lightning":"tindrem llamps",
       "lightning-rainy": "tindrem llamps i pluja",
       "pouring": "tindrem pluges torrencials",
       "rainy":"plourà",
       "snowy": "nevarà",
       "snowy-rainy":"plourà i nevarà",
       "windy": "farà vent",
       "windy-variant": "farà vent",
       "exceptional": ""
}
days =["dilluns", "dimarts", "dimecres", "dijous", "divendres", "dissabte", "diumenge"]

x=hass.states.get("weather.casa").attributes['forecast']
sensor_prediccio=hass.states.get("sensor.prediccio").attributes
nova_prediccio={}
i=0

for v in x:
    z=v['datetime']
    fech=str(z)
    cadena=str("T")
    fecha, hora=fech.split(cadena)
    wk=datetime.datetime.strptime(fecha, '%Y-%m-%d').weekday()
    nova_prediccio[days[wk]]= "  " + d[v['condition']] + " amb una temperatura màxima de " +
str(v['temperature']) + "."
    i=i+1
if len(nova_prediccio)==0:
   nova_prediccio=sensor_prediccio
   hass.states.set("sensor.prediccio", "no hi ha dades", nova_prediccio)
else:
   hass.states.set("sensor.prediccio", "hi ha dades", nova_prediccio)
```

7 Arxius de Rhasspy

scenes.yaml

```
{
   "dialogue": {
      "system": "rhasspy"
   },
   "handle": {
      "system": "hass"
   },
   "home_assistant": {
      "access_token": "ey[resta del token]yRQ",
      "url": "https://tfg:8123"
   },
   "intent": {
      "fsticuffs": {
         "fuzzy": false
      },
      "system": "fsticuffs"
   },
   "microphone": {
      "arecord": {
         "device": "default:CARD=Device"
      },
      "system": "arecord"
   },
   "sounds": {
      "aplay": {
         "device": "sysdefault:CARD=Headphones",
         "volume": "1"
      },
      "error": "$RHASSPY_PROFILE_DIR/wav/beep_error.wav",
      "recorded": "$RHASSPY_PROFILE_DIR/wav/beep_lo.wav",
      "system": "aplay",
      "wake": "$RHASSPY_PROFILE_DIR/wav/beep_hi.wav"
   },
   "speech_to_text": {
      "system": "pocketsphinx"
   },
   "text_to_speech": {
      "espeak": {
         "arguments": [
            "-s",
            "150"
         ]
      },
      "system": "espeak"
   },
   "wake": {
      "pocketsphinx": {
         "keyphrase": "escolta",
         "threshold": 1e-14
      },
      "system": "pocketsphinx"
   }
}
```

sentences.ini

```
[Acudit]
contam un acudit $colors

[GetTemperatureInt]
quina és la temperatura a casa
[GetTemperatureExt]
quina és la temperatura afora
[GetTime]
quina hora és
dime l'hora

[Prediccio]
Quina és la predicció del temps
[PrediccioDia]
Quin temps farà (dilluns | dimarts | dimecres | dijous | divendres | dissabte | diumenge){dia}

[ActiveScene]
activa escena de ($scene) {escena}

[ChangeLightStateDespatx]
states = (Encén | Apaga ) {light_state}
<states> el despatx

[HassTurnOn]
Encén la (llum del despatx){name}
Encén l'(endoll){name}
[HassTurnOff]
Apaga la (llum del despatx){name}
Apaga l'(endoll){name}
[HassLightSet]
Posa [el] (llum del despatx | llum de la mesita){name} al (màxim:100 | mínim:10 | mig:50)
{brightness}
```