Lecture #7
AI Assistants

Spring 2024

What we will build

- A custom Google Assistant device.
- It will use a button to trigger a microphone and playback the Assistant answer on the speaker.

Image source: https://aiyprojects.withgoogle.com

What we will learn

- Record and playback audio from I2S devices.
- Use the Google Assistant gRPC API.

Image source: https://dpi.wi.gov

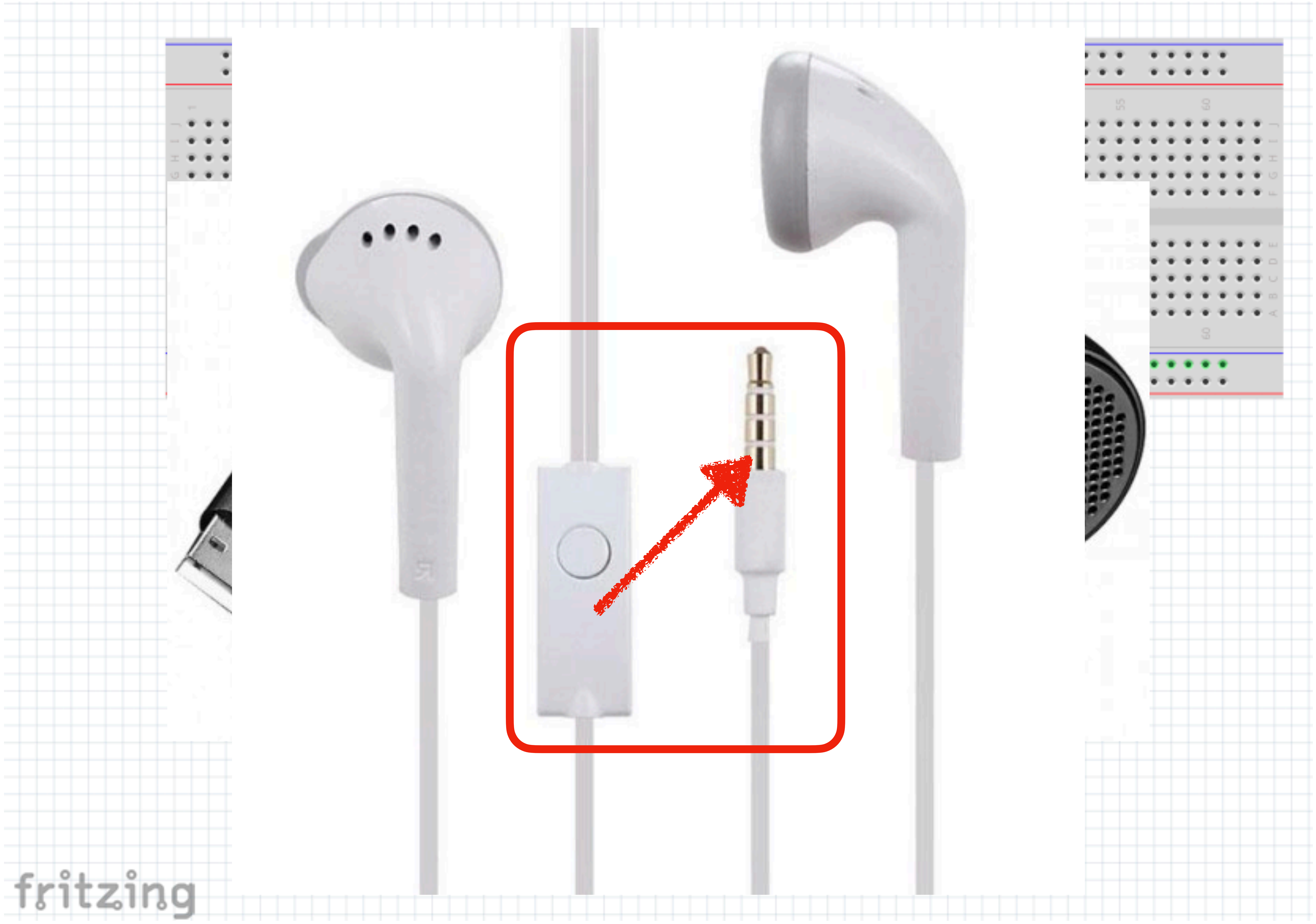
- ✓ Development board that is running Android Things.
- ✓ Microphone and speaker.
- ✓ WiFi/Ethernet connection.
- ✓ Android Studio 3.0+.
- ✓ A LED.
- ✓ A Google/Gmail account.

Image source: https://www.dreamstime.com

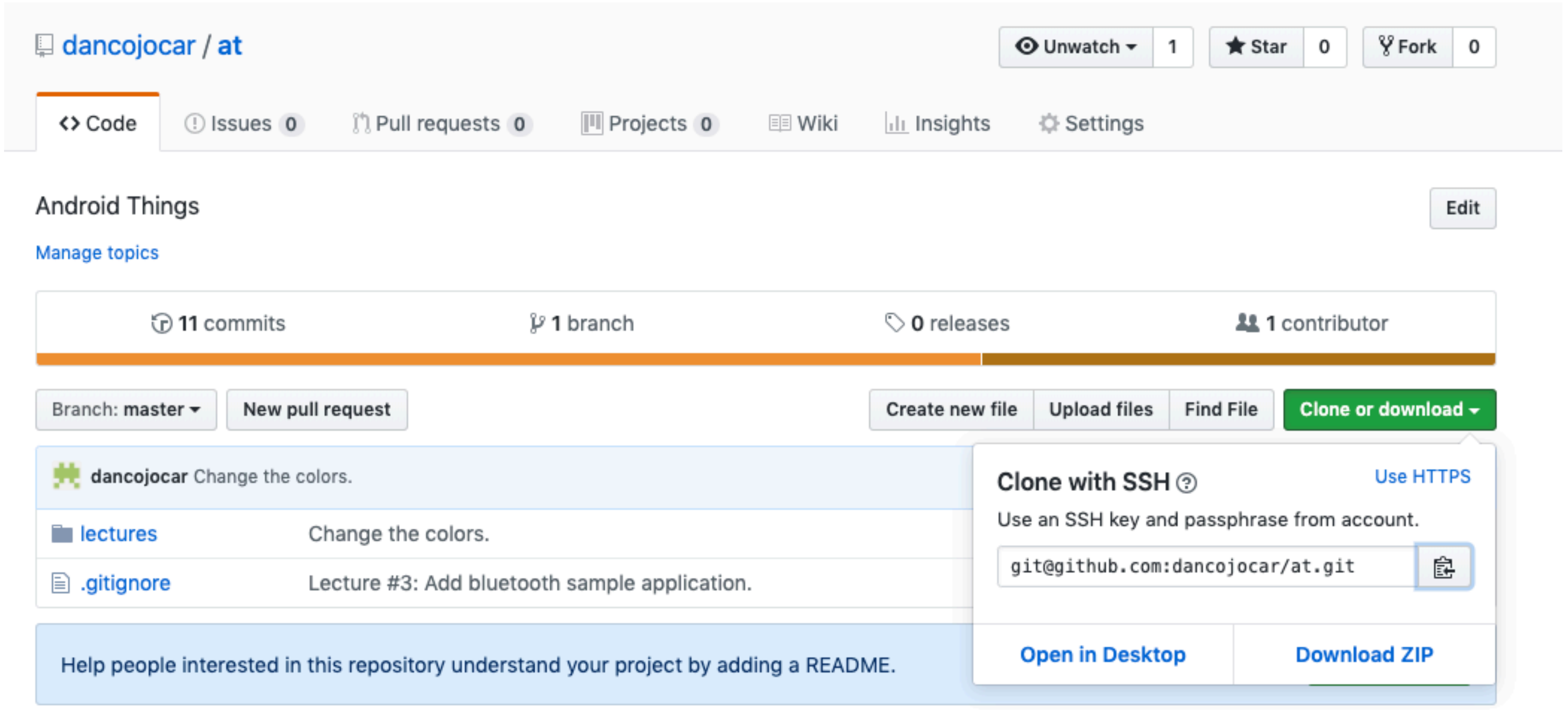
Voice Kit

https://aiyprojects.withgoogle.com/voice/

Assemble the Hardware



Get the Sample Code


dancojocar / at

Unwatch 1 Star 0 Fork 0

Code · Issues 0 · Pull requests 0 · Projects 0 · Wiki · Insights · Settings

Android Things

Manage topics

11 commits · 1 branch · 0 releases · 1 contributor

Branch: master · New pull request · Create new file · Upload files · Find File · Clone or download

dancojocar Change the colors.

lectures	Change the colors.
.gitignore	Lecture #3: Add bluetooth sample application.

Help people interested in this repository understand your project by adding a README.

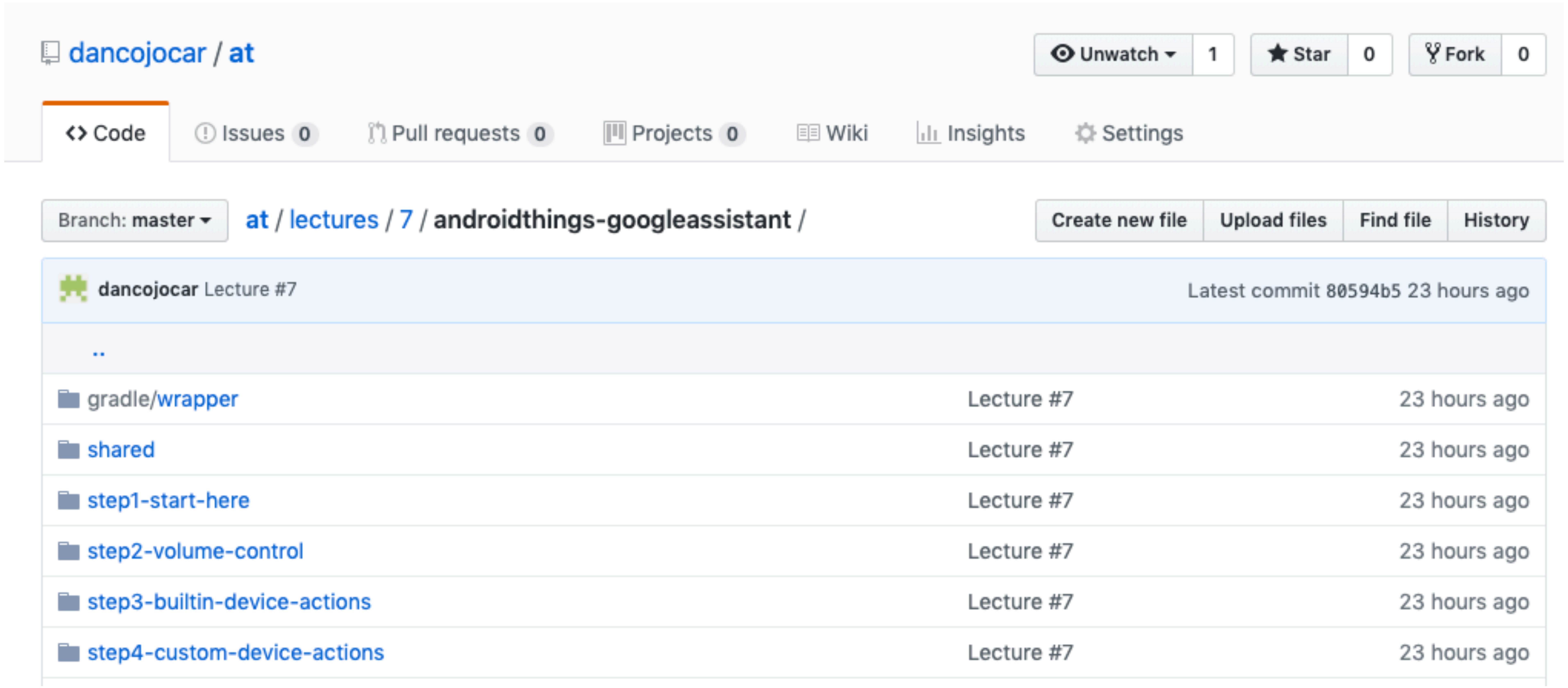
Clone with SSH · Use HTTPS

Use an SSH key and passphrase from account.

```
git@github.com:dancojocar/at.git
```

Open in Desktop · Download ZIP

Get the Sample Code



dancojocar / at

at / lectures / 7 / androidthings-googleassistant /

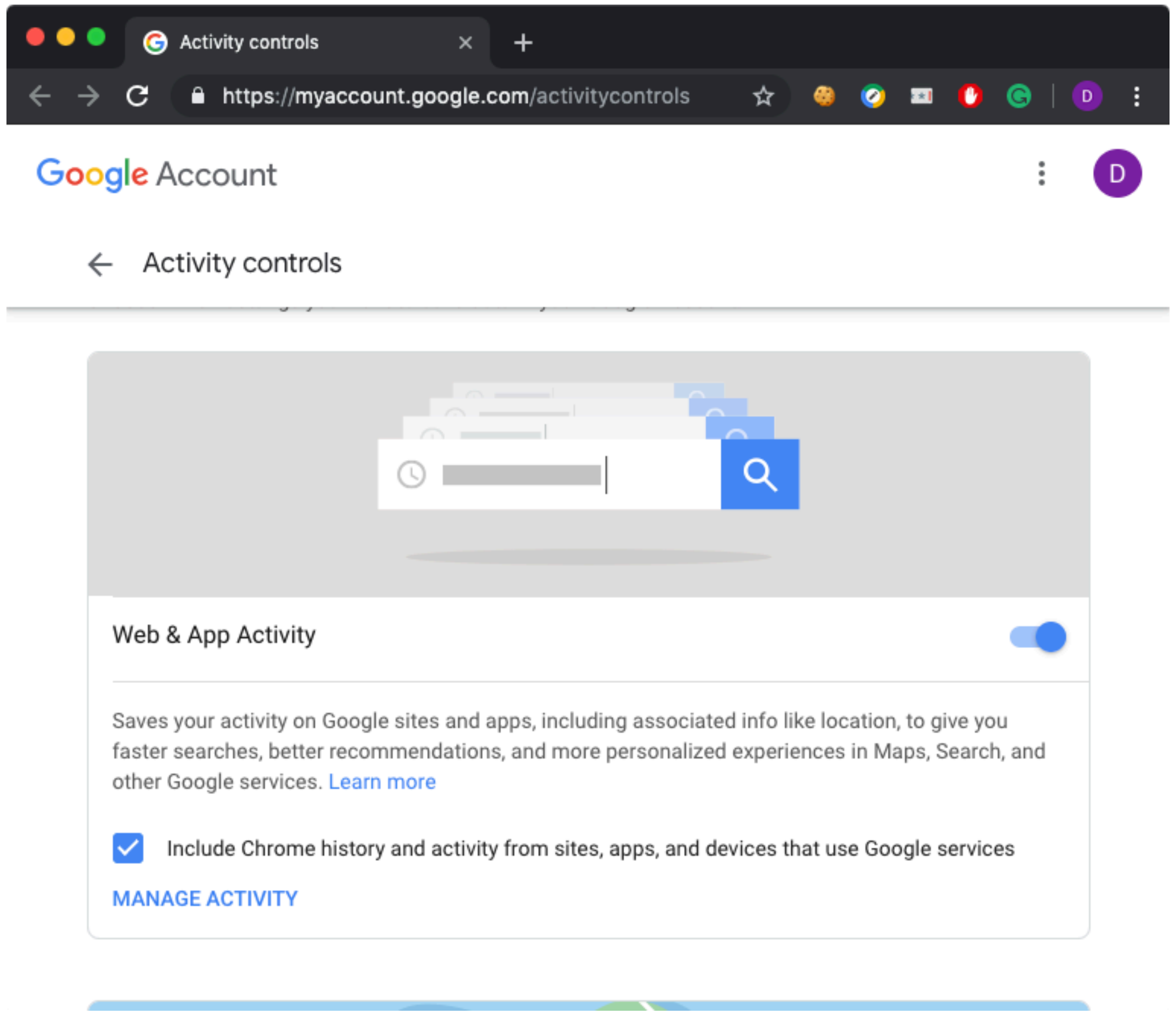
dancojocar Lecture #7 Latest commit 80594b5 23 hours ago

..

gradle/wrapper	Lecture #7	23 hours ago
shared	Lecture #7	23 hours ago
step1-start-here	Lecture #7	23 hours ago
step2-volume-control	Lecture #7	23 hours ago
step3-builtin-device-actions	Lecture #7	23 hours ago
step4-custom-device-actions	Lecture #7	23 hours ago

Configure the Credentials

- In Activity controls enable:
  - Web & App Activity
  - Device Information
  - Voice & Audio Activity



Web & App Activity

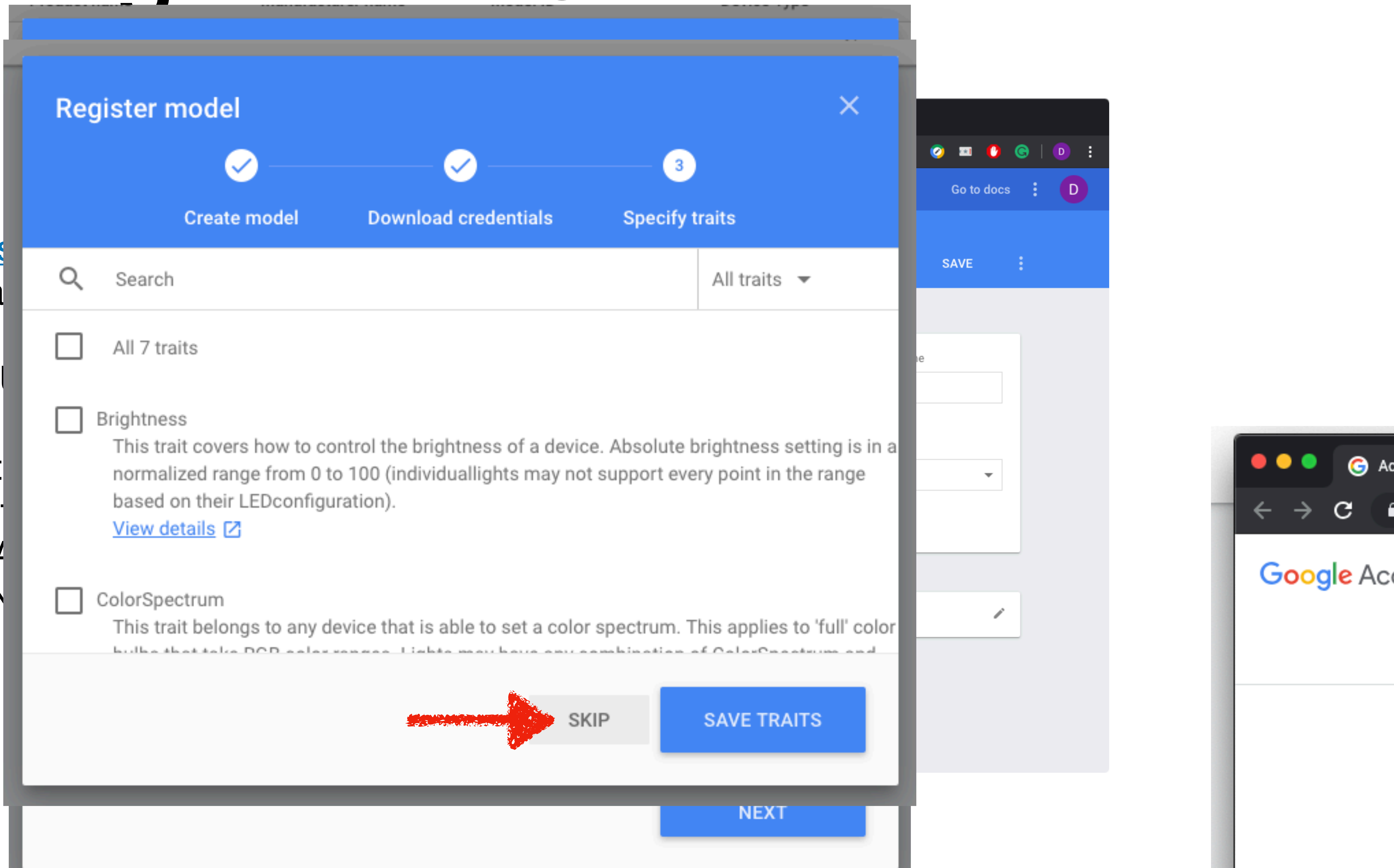
Saves your activity on Google sites and apps, including associated info like location, to give you faster searches, better recommendations, and more personalized experiences in Maps, Search, and other Google services. Learn more

☑ Include Chrome history and activity from sites, apps, and devices that use Google services

MANAGE ACTIVITY

Configure the Credentials

- In Actions register a
  - Create/
  - Select t registra (under A OPTION navbar.



Register model

Create model · Download credentials · Specify traits

Search · All traits

All 7 traits

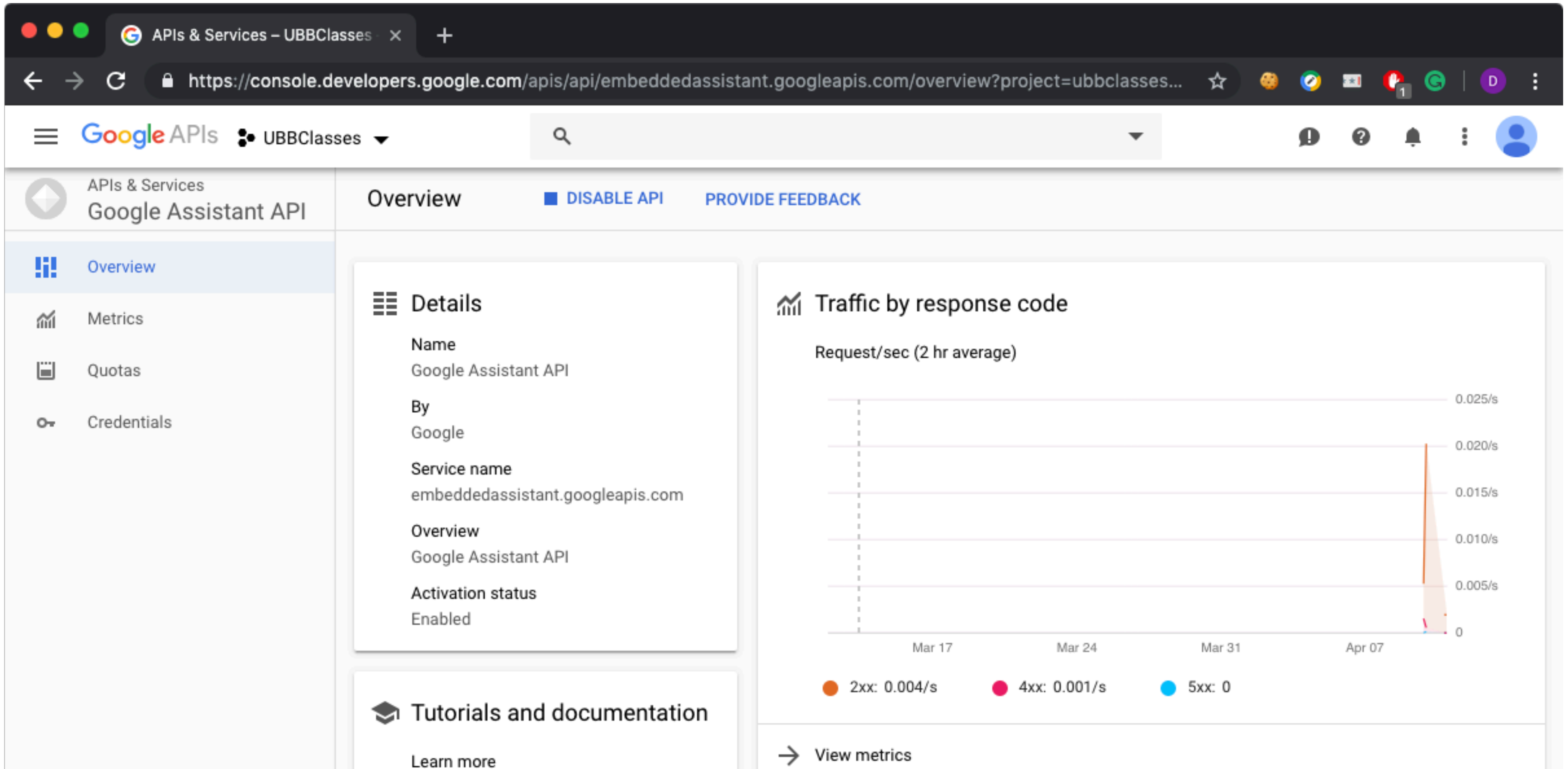
Brightness
This trait covers how to control the brightness of a device. Absolute brightness setting is in a normalized range from 0 to 100 (individuallights may not support every point in the range based on their LEDconfiguration).
View details

ColorSpectrum
This trait belongs to any device that is able to set a color spectrum. This applies to 'full' color

SKIP · SAVE TRAITS

Enable the Google Assistant API

- Enable Google Assistant API in the Cloud Console



APIs & Services – UBBClasses
https://console.developers.google.com/apis/api/embeddedassistant.googleapis.com/overview?project=ubbclasses...
Google APIs
UBBClasses
APIs & Services
Google Assistant API
Overview
DISABLE API
PROVIDE FEEDBACK
Overview
Metrics
Quotas
Credentials

Details

Name
Google Assistant API

By
Google

Service name
embeddedassistant.googleapis.com

Overview
Google Assistant API

Activation status
Enabled

Traffic by response code

Request/sec (2 hr average)

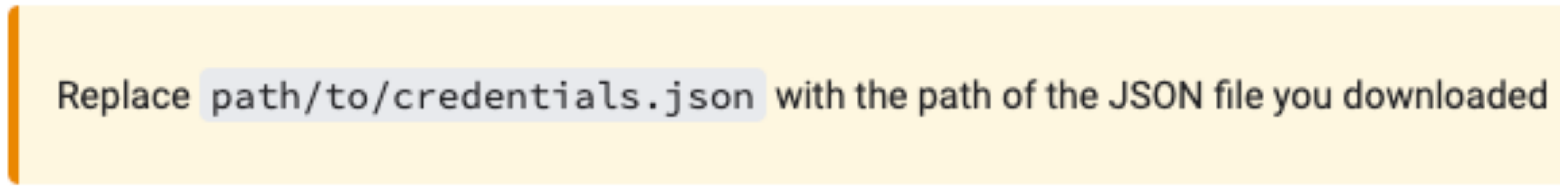
● 2xx: 0.004/s ● 4xx: 0.001/s ● 5xx: 0

Configure a new Python Virtual Environment

```
$ python3 -m venv env
$ source env/bin/activate
(env) $ pip install --upgrade pip setuptools wheel
(env) $ pip install --upgrade “google-auth-oauthlib[tool]”
```

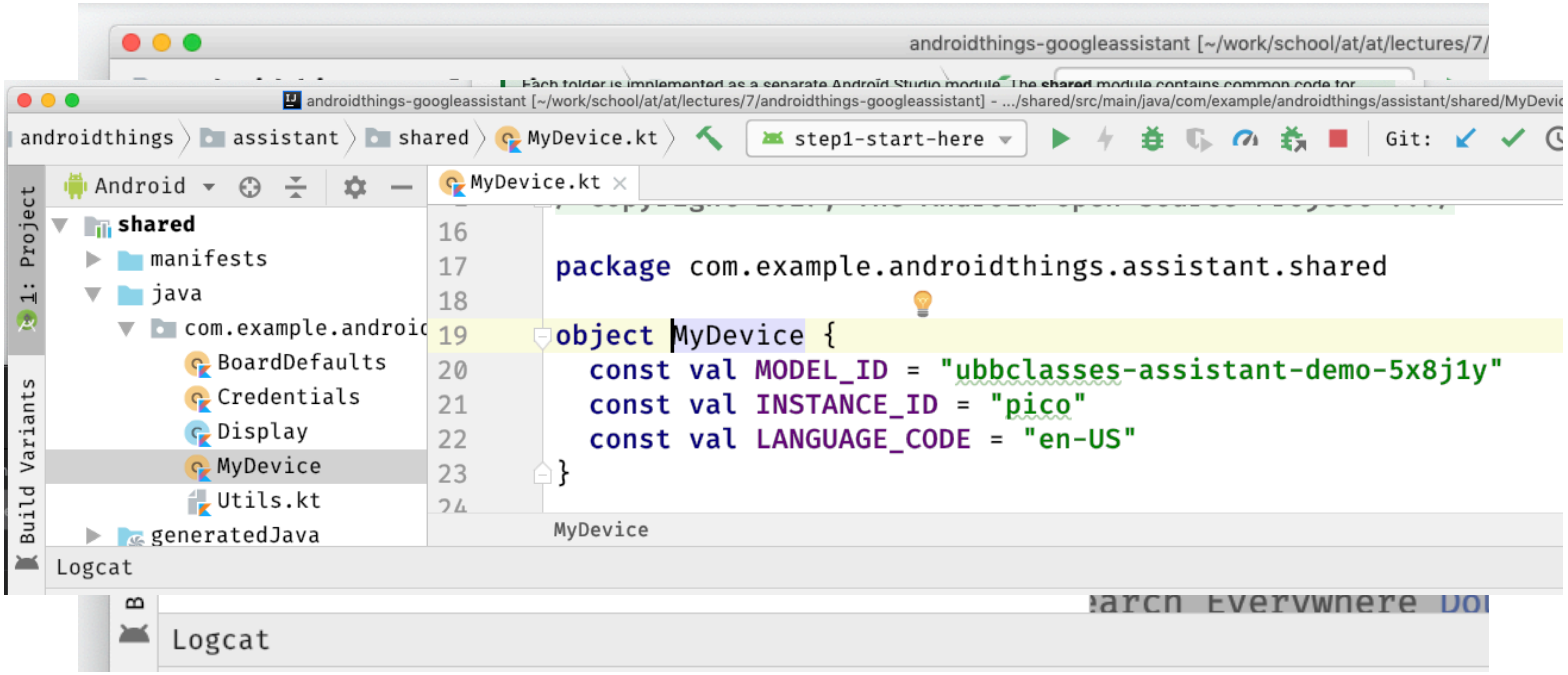
Generate the credentials

```
(env) $ google-oauthlib-tool --client-secrets path/to/credentials.json \
    --credentials shared/src/main/res/raw/credentials.json \
    --scope https://www.googleapis.com/auth/assistant-sdk-prototype \
    --save
```



Replace `path/to/credentials.json` with the path of the JSON file you downloaded

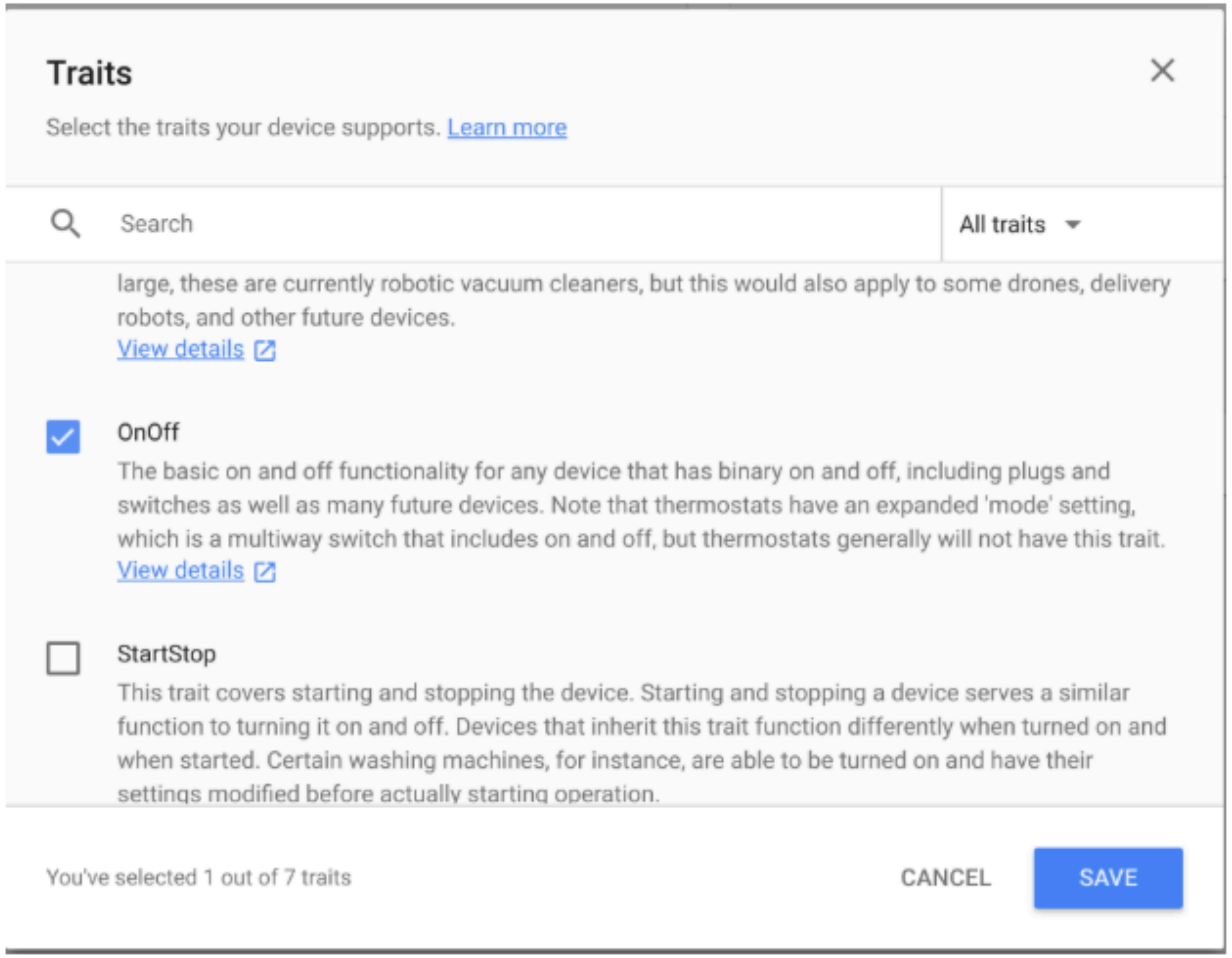
Run the step1-start-here module



```kotlin
package com.example.androidthings.assistant.shared

object MyDevice {
  const val MODEL_ID = "ubbclasses-assistant-demo-5x8j1y"
  const val INSTANCE_ID = "pico"
  const val LANGUAGE_CODE = "en-US"
}
```

Add Built-In Device Actions



Traits

Select the traits your device supports. Learn more

Search

All traits

large, these are currently robotic vacuum cleaners, but this would also apply to some drones, delivery robots, and other future devices.
View details

[✓] OnOff
The basic on and off functionality for any device that has binary on and off, including plugs and switches as well as many future devices. Note that thermostats have an expanded 'mode' setting, which is a multiway switch that includes on and off, but thermostats generally will not have this trait.
View details

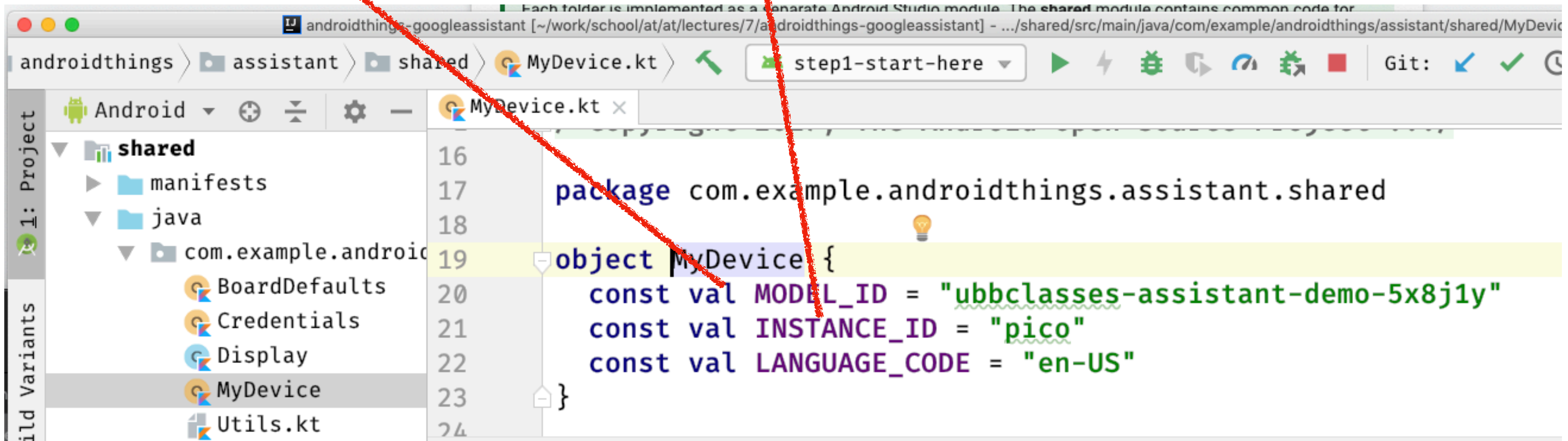
[ ] StartStop
This trait covers starting and stopping the device. Starting and stopping a device serves a similar function to turning it on and off. Devices that inherit this trait function differently when turned on and when started. Certain washing machines, for instance, are able to be turned on and have their settings modified before actually starting operation.

You've selected 1 out of 7 traits

CANCEL SAVE

Register the Device

```
(env) $ google-oauthlib-tool --client-secrets path/to/credentials.json \
   --scope https://www.googleapis.com/auth/assistant-sdk-prototype \
   --save
(env) $ pip install google-assistant-sdk
(env) $ googlesamples-assistant-devicetool --project-id PROJECT_ID list --model
...
(env) $ googlesamples-assistant-devicetool --project-id PROJECT_ID register-device \
--model MODEL_ID --device DEVICE_INSTANCE_ID --client-type SERVICE
```



```
package com.example.androidthings.assistant.shared

object MyDevice {
    const val MODEL_ID = "ubbclasses-assistant-demo-5x8j1y"
    const val INSTANCE_ID = "pico"
    const val LANGUAGE_CODE = "en-US"
}
```

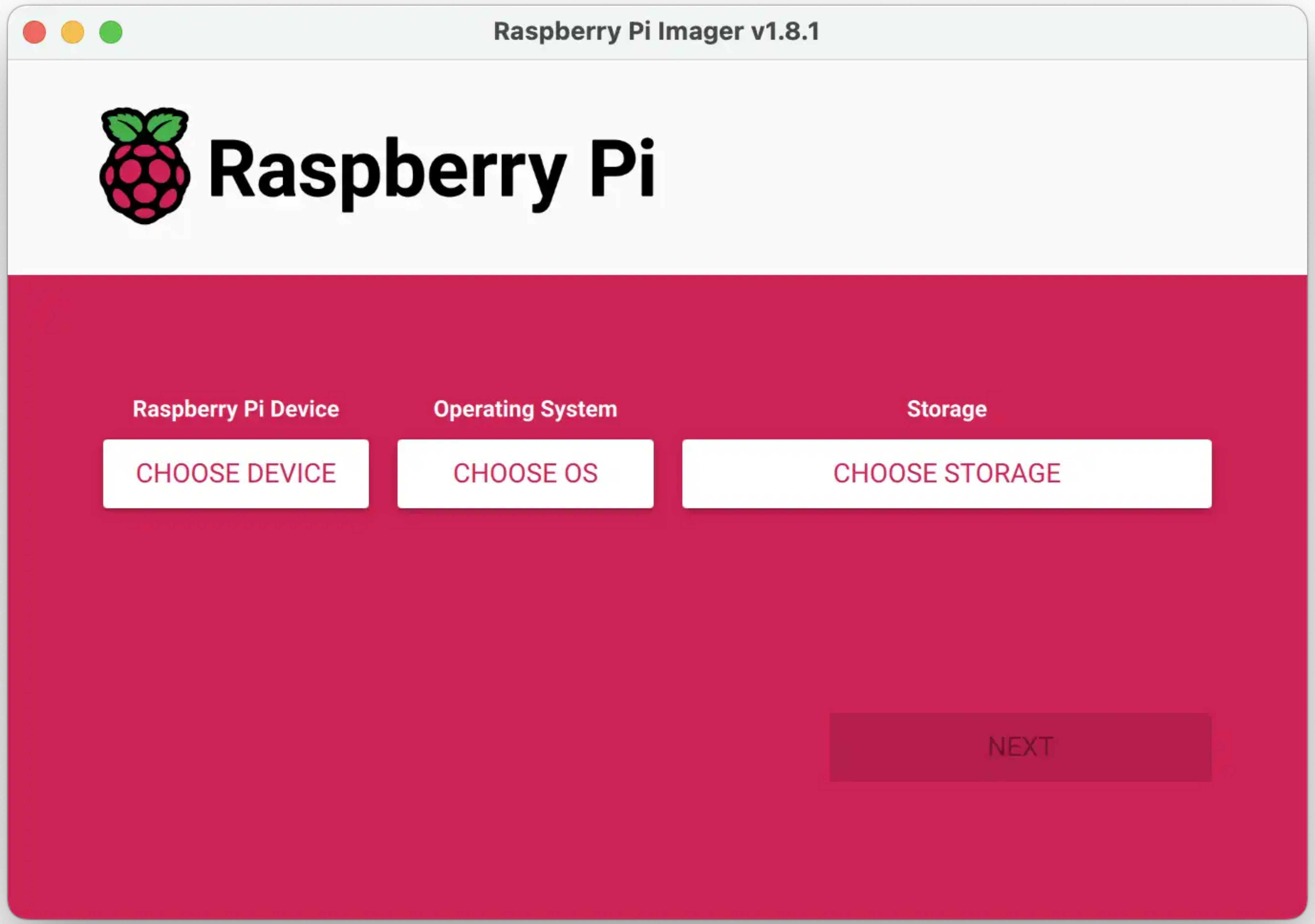
Add Custom Device Actions

```
{
  "manifest": {
    "displayName": "Display",
    "invocationName": "Display",
    "category": "PRODUCTIVITY"
  },
  "actions": [
    {
      "name": "com.example.actions.Display",
      "availability": {
        "deviceClasses": [
          {
            "assistantSdkDevice": {}
          }
        ]
      },
```

Add Custom Device Actions

```
(env) $ gactions test --action_package actions.json --project project_id
```

Raspberry Pi OS



Setting Up Your Audio for Google Assistant

Locate your USB microphone by utilizing the following command:

```
arecord -l
```

Now to locate your speaker:

```
aplay -l
```

Setting Up Your Audio for Google Assistant

```
Create: /home/pi/.asoundrc

pcm.!default {
  type asym
  capture.pcm "mic"
  playback.pcm "speaker"
}
pcm.mic {
  type plug
  slave {
    pcm "hw:<card number>,<device number>"
  capture.pcm "mic"
  playback.pcm "speaker"
}
pcm.mic {
  type plug
  slave {
    pcm "hw:<card number>,<device number>"
    rate 48000
  }
}
pcm.speaker {
  type plug
  slave {
    pcm "hw:<card number>,<device number>"
  }
}
```

Testing Your Speakers and Microphone

```
#test speaker
speaker-test -t wav

#test mic
arecord --format=S16_LE --duration=5 --rate=16000
      –file-type=raw out.raw

#play recording
aplay --format=S16_LE --rate=16000 out.raw

#settings
alsamixer
```

Setting Up Google Assistant

```
#update the packages
sudo apt update

#setup credentials from google
~/googleassistant/credentials.json

#update php deps
sudo apt install python3-dev python3-venv python3-
pip libssl-dev libffi-dev libportaudio2
```

Create virtual env

```
#python env
python3 -m venv env

#update env
env/bin/python3 -m pip install --upgrade pip
setuptools --upgrade

#usetup
source env/bin/activate
```

Install Google Assistant

```
python3 -m pip install --upgrade google-assistant-library
python3 -m pip install --upgrade google-assistant-sdk[samples]
```

Install Google Assistant

```
python3 -m pip install --upgrade google-assistant-library
python3 -m pip install --upgrade google-assistant-sdk[samples]
```

Install Google Assistant

```
python3 -m pip install --upgrade google-auth-oauthlib[tool]

google-oauthlib-tool --client-secrets ~/googleassistant/credentials.json \
--scope https://www.googleapis.com/auth/assistant-sdk-prototype \
--scope https://www.googleapis.com/auth/gcm \
--save --headless
```

Register Google Assistant

```
googlesamples-assistant-pushtotalk --project-id <projectid>
             —-device-model-id <deviceid>
```

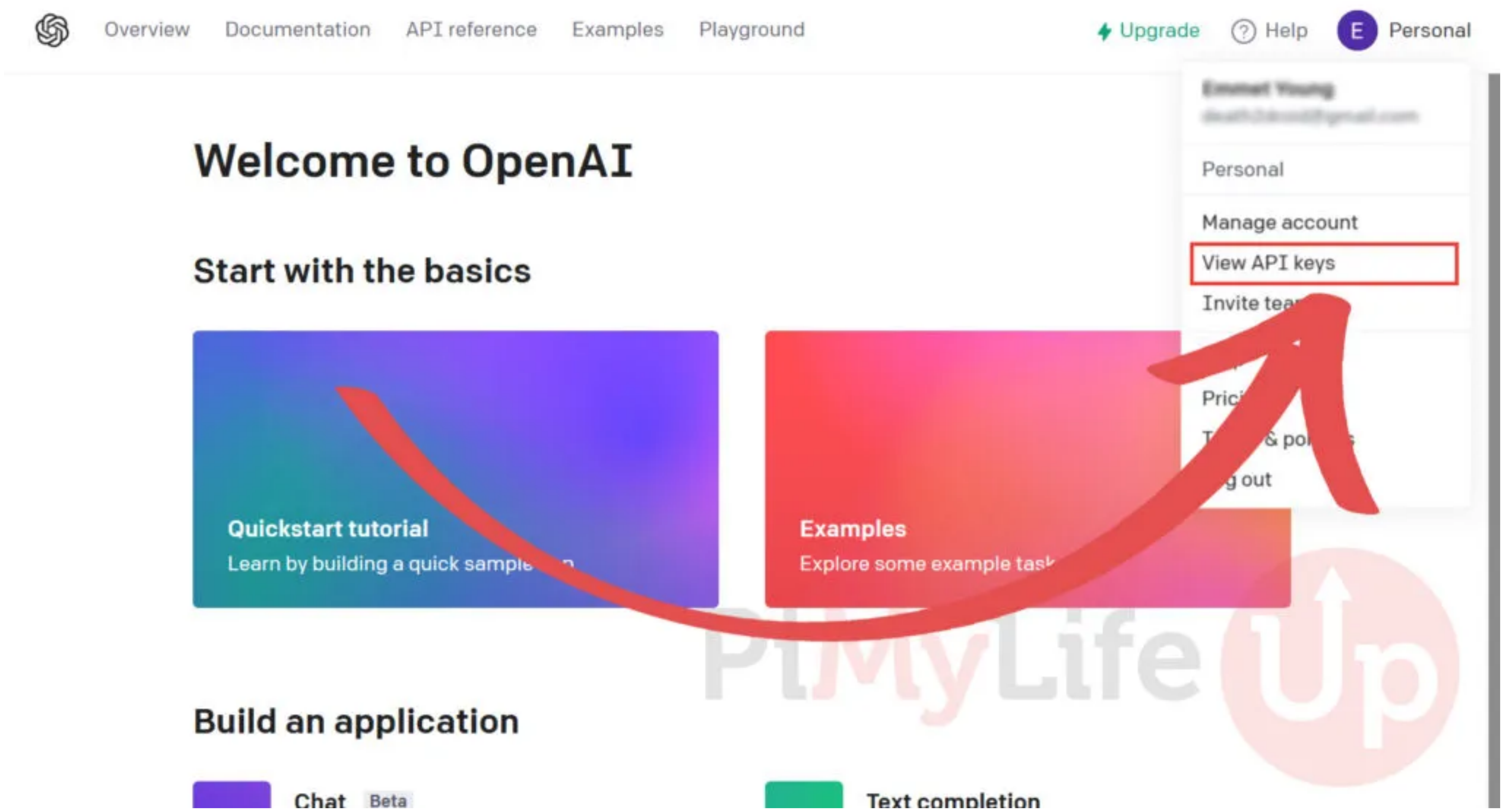
Execute Google Assistant

```
googlesamples-assistant-pushtotalk

#hotword
googlesamples-assistant-hotword
    --device-model-id <deviceid>
```

ChatGPT

Get the API key



Welcome to OpenAI

Start with the basics

Build an application

Install deps

```
python3 -m pip install openai
```

Install deps

```
from openai import OpenAI

client = OpenAI(
    api_key = "SECRETKEY"
)

messages = [
    {
        "role": "system",
        "content": "You are a helpful assistant"
    }
]

while True:
    message = input("You: ")

    messages.append(
        {
            "role": "user",
            "content": message
        },
    )
```

```
messages = [
    {
        "role": "system",
        "content": "You are a helpful assistant"
    }
]

while True:
    message = input("You: ")

    messages.append(
        {
            "role": "user",
            "content": message
        },
    )

    chat = client.chat.completions.create(
        messages=messages,
        model="gpt-3.5-turbo"
    )

    reply = chat.choices[0].message

    print("Assistant: ", reply.content)

    messages.append(reply)
```

Lecture outcomes

- Integrate Google Assistant.
- Define custom actions.
- Connect to OpenAi.