

Edge Impulse op een ESP32-CAM - Image Classification

Inhoudsopgave

- [1. Inleiding](#)
- [2. Benodigde Software](#)
- [3. Benodigde Hardware](#)
- [4. IDE Setup](#)
- [5. Data Collectie](#)
- [6. Edge Impulse](#)
 - [6.1. Project Setup](#)
 - [6.2. Data Uploaden](#)
 - [6.3. Impulse Design](#)
 - [6.4. Model Training](#)
- [7. Deployment](#)

1. Inleiding

Dit document beschrijft hoe je Edge Impulse kunt gebruiken met een ESP32-CAM voor posture detection of een ander classificatie model. We doorlopen stap voor stap het proces van setup tot deployment van het model.

2. Benodigde Software

- Arduino IDE: <https://www.arduino.cc/en/software/>
- Edge Impulse (Online Omgeving): <https://edgeimpulse.com>

3. Benodigde Hardware

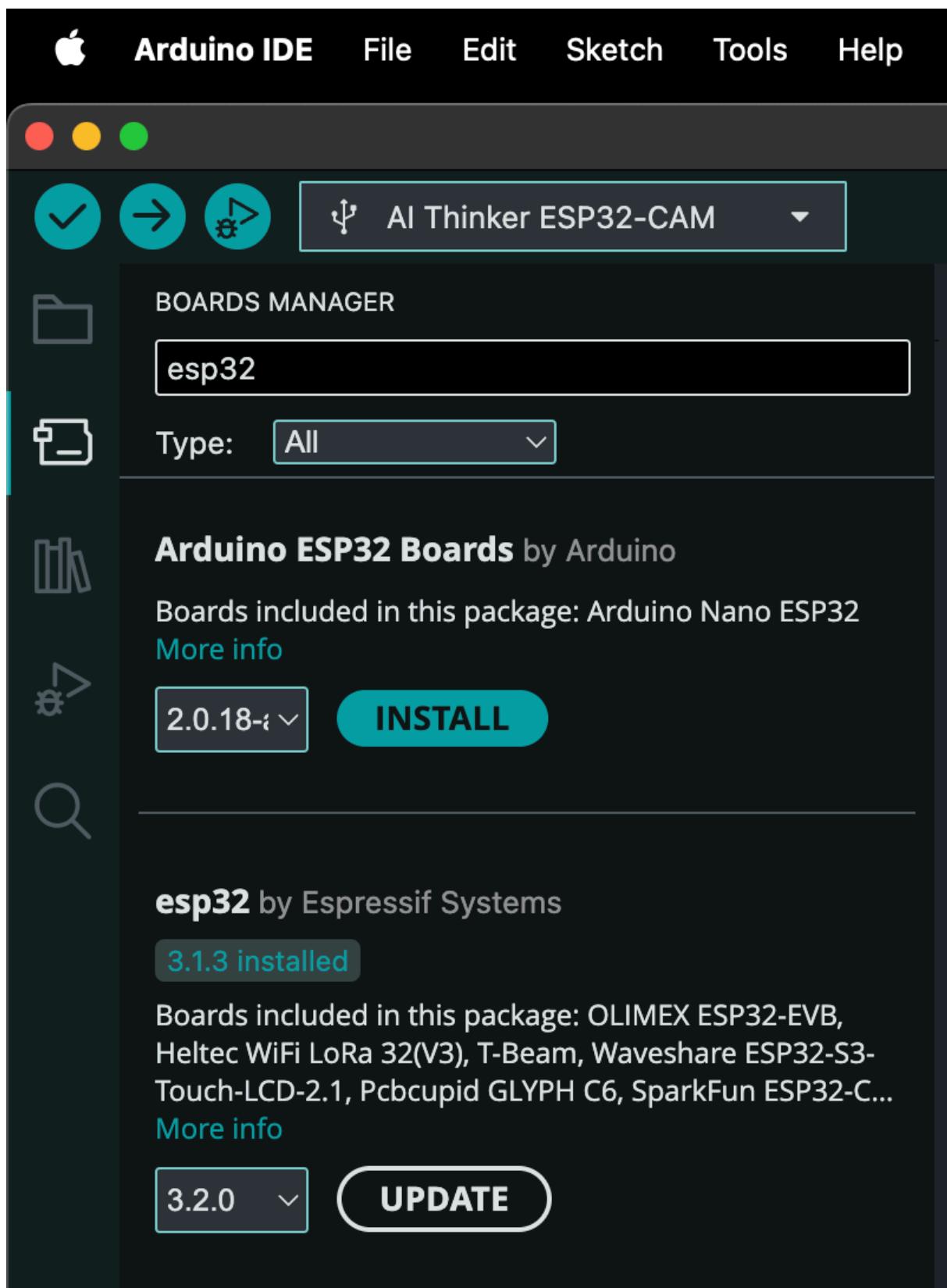
- ESP32-CAM met een OV2640 Camera module (BELANGRIJK!)

- ESP32-CAM-MB Programmer Shield
- Micro-USB Kabel

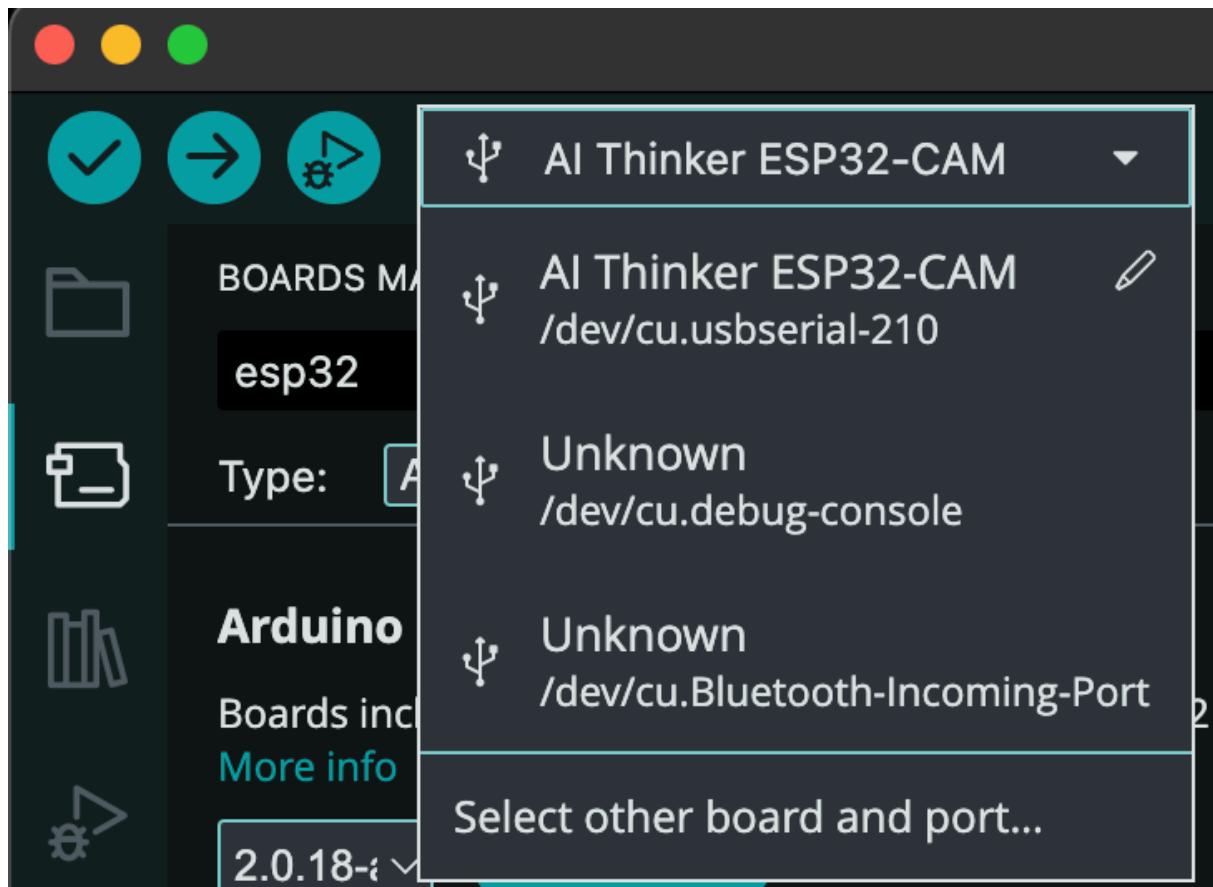


4. IDE Setup

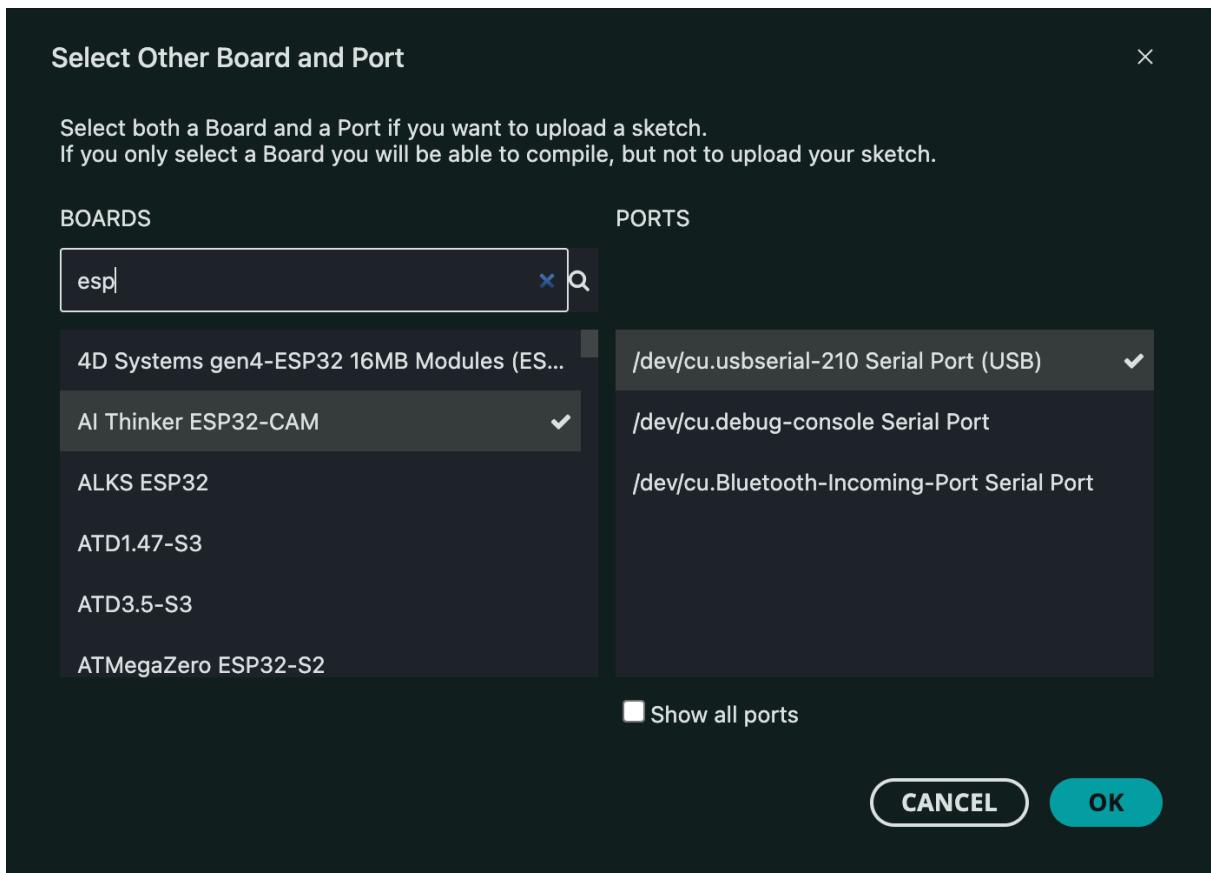
Download de Arduino IDE hier: <https://www.arduino.cc/en/software/>



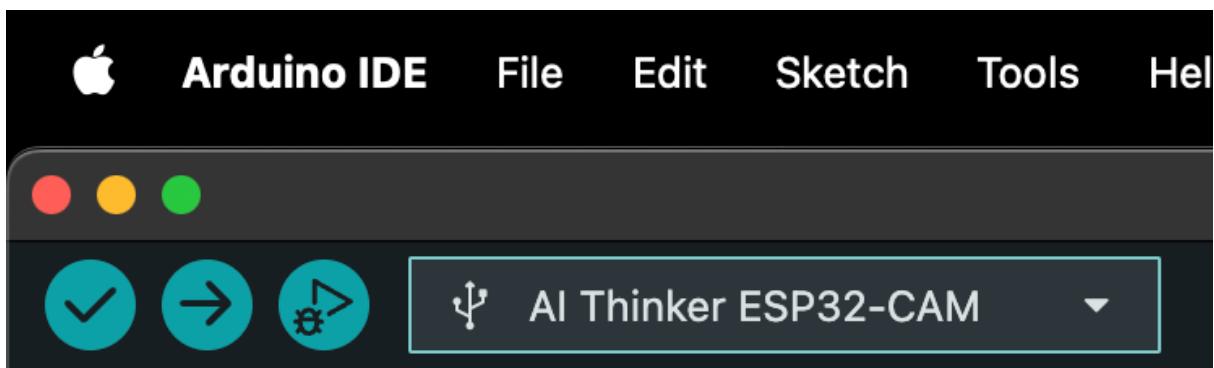
Na het installeren van deze library kan je je board aansluiten en de juiste variant kiezen, in ons geval AI Thinker ESP32-CAM.



Druk op de Drop down en selecteer "Select Other Board and Port"



Selecteer hier vervolgens de "AI Thinker ESP32-CAM" en druk op OK. Nu weet de Arduino IDE welk Board er is aangesloten.

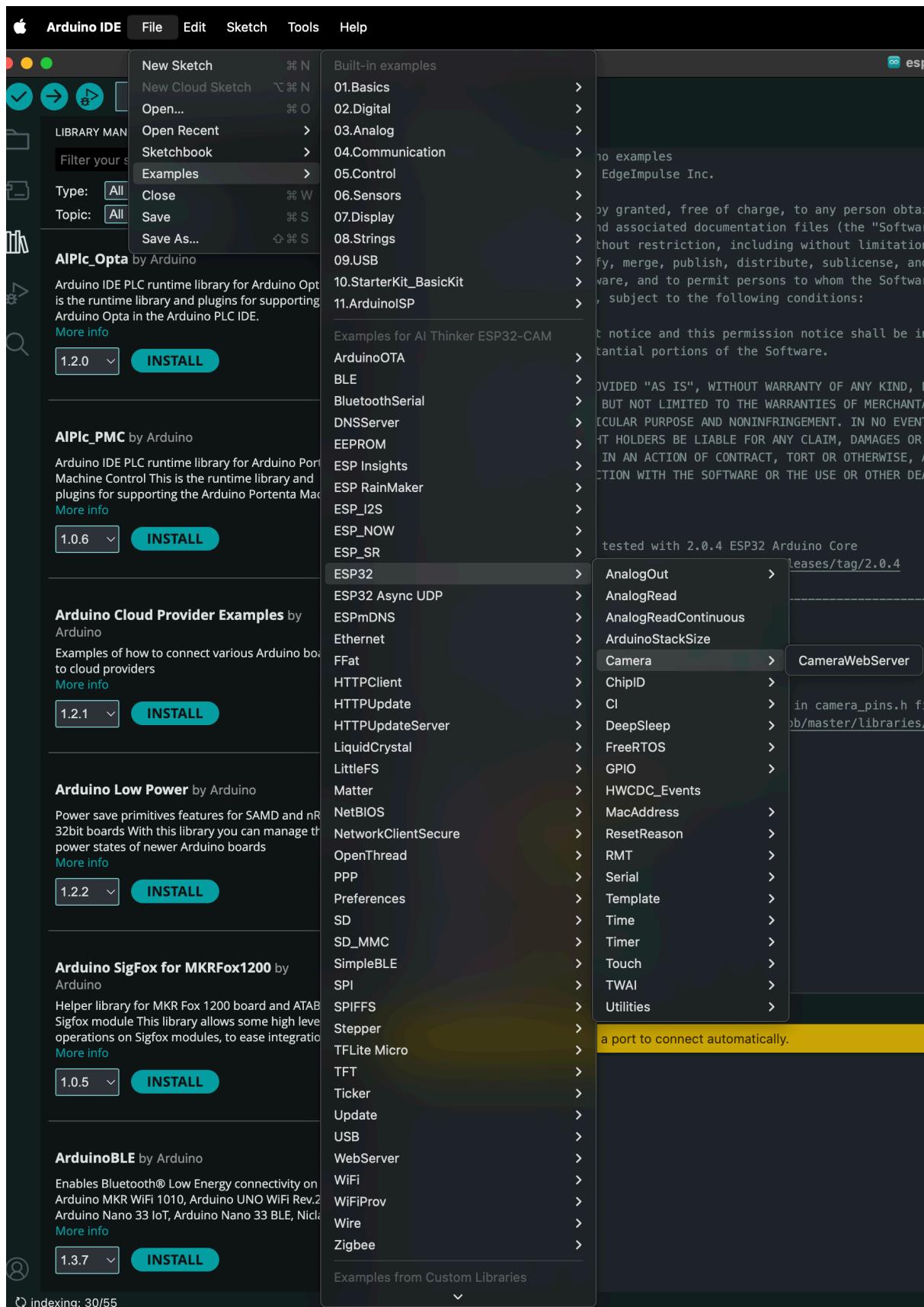


5. Data Collectie

Om een afbeelding classificatie model te trainen heb je natuurlijk fotos nodig!

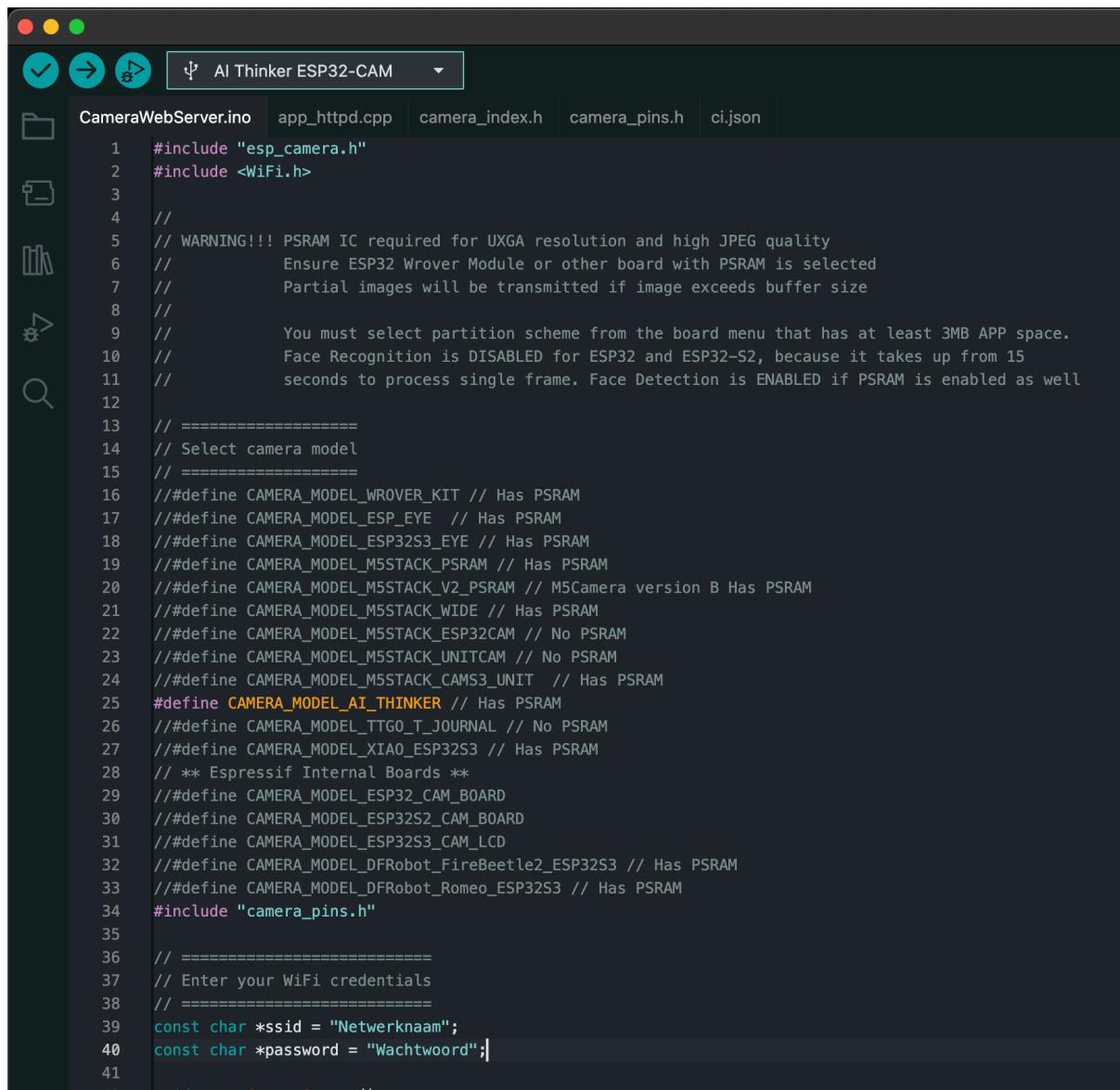
Deze gaan we verzamelen vanaf de ESP32-CAM zelf.

Dit doen we door gebruik te maken van de "CameraWebServer" Example die we in de vorige stap hebben geïnstalleerd via de "esp32" library.



Na het openen van deze sketch moet je een paar instellingen aanpassen:

- Zorg ervoor dat het juiste model is geselecteerd zoals hieronder te zien.
(`#define CAMERA_MODEL_AI_THINKER`)
- Vul het juiste `"ssid"` in. (LET OP: je hebt een 2,4GHZ netwerk nodig, het handigste is een eigen hotspot vanaf je telefoon wanneer je op school bent)
- Vul het correcte `"password"` voor dit netwerk in.



The screenshot shows the Arduino IDE interface with the title bar "AI Thinker ESP32-CAM". The central area displays the code for "CameraWebServer.ino". The code includes comments about PSRAM requirements and camera model selection. It defines the `CAMERA_MODEL_AI_THINKER` constant. At the bottom, it sets WiFi credentials with `const char *ssid = "Netwerknaam";` and `const char *password = "Wachtwoord";`.

```
#include "esp_camera.h"
#include <WiFi.h>

// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
// Ensure ESP32 Wrover Module or other board with PSRAM is selected
// Partial images will be transmitted if image exceeds buffer size

// You must select partition scheme from the board menu that has at least 3MB APP space.
// Face Recognition is DISABLED for ESP32 and ESP32-S2, because it takes up from 15
// seconds to process single frame. Face Detection is ENABLED if PSRAM is enabled as well

// =====
// Select camera model
// =====
#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
#define CAMERA_MODEL_ESP_EYE // Has PSRAM
#define CAMERA_MODEL_ESP32S3_EYE // Has PSRAM
#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
#define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
#define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
#define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
#define CAMERA_MODEL_M5STACK_CAMS3_UNIT // Has PSRAM
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM
#define CAMERA_MODEL_XIAO_ESP32S3 // Has PSRAM

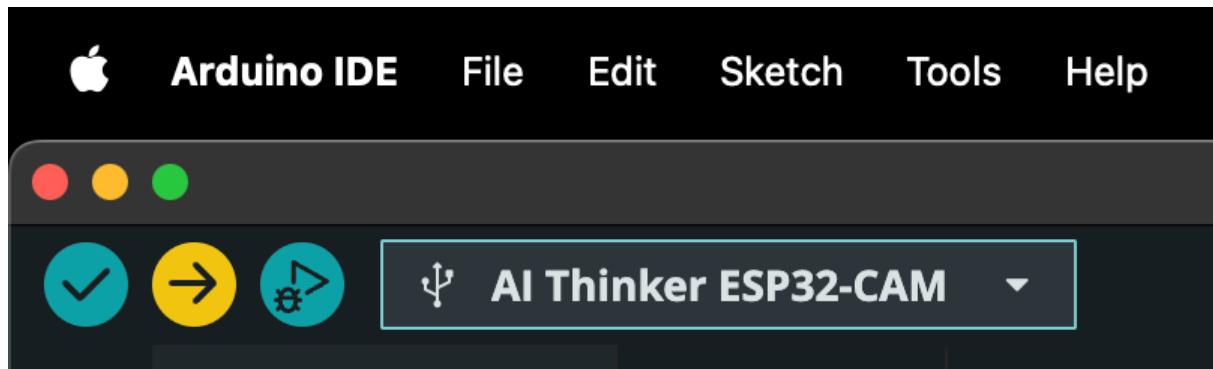
// ** Espressif Internal Boards **
#define CAMERA_MODEL_ESP32_CAM_BOARD
#define CAMERA_MODEL_ESP32S2_CAM_BOARD
#define CAMERA_MODEL_ESP32S3_CAM_LCD
#define CAMERA_MODEL_DFRobot_FireBeetle2_ESP32S3 // Has PSRAM
#define CAMERA_MODEL_DFRobot_Romeo_ESP32S3 // Has PSRAM

#include "camera_pins.h"

// =====
// Enter your WiFi credentials
// =====
const char *ssid = "Netwerknaam";
const char *password = "Wachtwoord";
```

Na het aanpassen is het tijd om de sketch naar de ESP te flashen. Dit doe je door op de gele knop in de onderstaande afbeelding te drukken. LETOP:

Selecteer het juiste bord in de dropdown (AI Thinker ESP32-CAM)



Nu zal het flash process beginnen, je Sketch wordt gecompiled en wachten op de connectie met de ESP. Wanneer "Connecting....." zichtbaar wordt moet je op de knopjes van de esp drukken, dit is soms een beetje friemelen. Wat het beste werkt is de RST knop op de achterkant van de esp 2 seconden in te drukken en daarna de twee zijknopjes in te drukken.

A screenshot of the Arduino IDE Serial Monitor window. The code area at the top shows the following definitions:

```
23 //##define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
24 //##define CAMERA_MODEL_M5STACK_CAMS3_UNIT // Has PSRAM
25 #define CAMERA_MODEL_AI_THINKER // Has PSRAM
26 //##define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM
27 //##define CAMERA_MODEL_XIAO_ESP32S3 // Has PSRAM
28 // ** Espressif Internal Boards **
29 //##define CAMERA_MODEL_ESP32_CAM_BOARD
30 //##define CAMERA_MODEL_ESP32S2_CAM_BOARD
```

The 'Output' section shows the serial port configuration and the start of the connection:

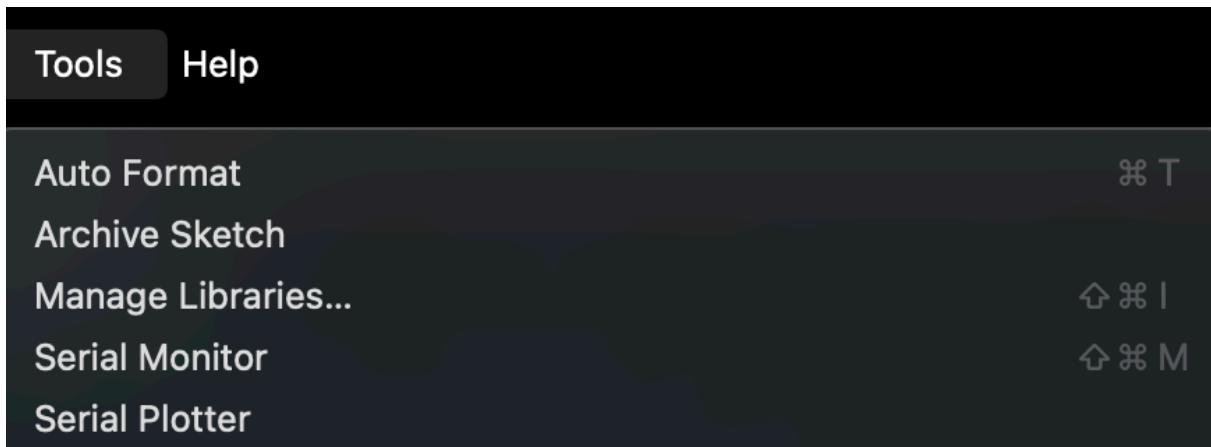
```
Output
Sketch uses 1045008 bytes (33%) of program storage space. Maximum is 3145728 bytes.
Global variables use 61252 bytes (18%) of dynamic memory, leaving 266428 bytes for local variables. Maximum is 327680 bytes.
esptool.py v4.8.1
Serial port /dev/cu.usbserial-210
Connecting.....
```

Wanneer de ESP is geconnect wordt het programma op de ESP geschreven een succesvolle flash ziet er als volgt uit

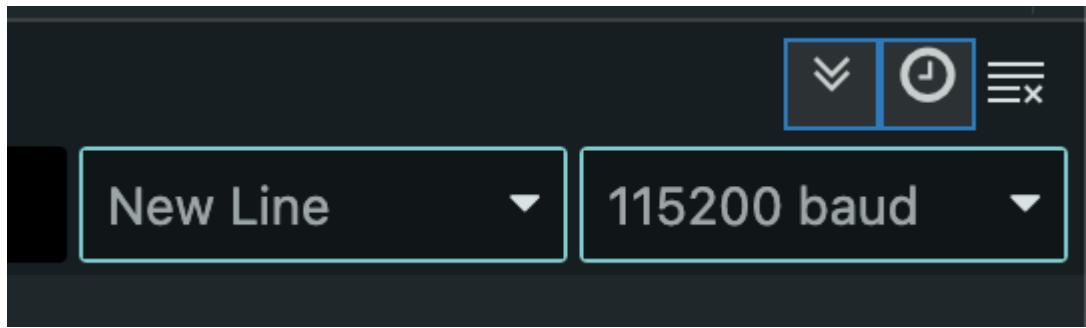
```
Output
Writing at 0x00091cd6... (50 %)
Writing at 0x00097434... (52 %)
Writing at 0x0009d047... (54 %)
Writing at 0x000a2938... (57 %)
Writing at 0x000a81ee... (59 %)
Writing at 0x000ad3dc... (61 %)
Writing at 0x000b270c... (64 %)
Writing at 0x000b7e94... (66 %)
Writing at 0x000bd708... (69 %)
Writing at 0x000c2a03... (71 %)
Writing at 0x000c7f1c... (73 %)
Writing at 0x000cdbab... (76 %)
Writing at 0x000d38e7... (78 %)
Writing at 0x000d8f98... (80 %)
Writing at 0x000e412d... (83 %)
Writing at 0x000e9d41... (85 %)
Writing at 0x000eeeea4... (88 %)
Writing at 0x000f603f... (90 %)
Writing at 0x000fd780... (92 %)
Writing at 0x001031b7... (95 %)
Writing at 0x00108968... (97 %)
Writing at 0x0010eb6c... (100 %)
Wrote 1045120 bytes (672850 compressed) at 0x00010000 in 17.1 seconds (effective 488.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

Druk nu op de RST knop zodat de ESP opnieuw opstart en ga naar de Serial Monitor



Zorg ervoor dat de Serial Monitor op 115200 baud staat, als je gekke tekens in de Serial Monitor ziet dan is dit waarschijnlijk het probleem.



Als alles goed is gegaan zie je nu een ip address in de Serial Monitor staan.

```
Output  Serial Monitor X
Message (Enter to send message to 'AI Thinker ESP32-CAM' on '/dev/cu.usbserial-210')
18:48:38.390 -> ets Jul 29 2019 12:21:46
18:48:38.390 ->
18:48:38.390 -> rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
18:48:38.390 -> configSip: 0, SPIWP:0xee
18:48:38.390 -> clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
18:48:38.390 -> mode:DIO0, clock div:1
18:48:38.390 -> load:0x3fff0030,len:4916
18:48:38.390 -> load:0x40078000,len:16492
18:48:38.390 -> load:0x40080400,len:4
18:48:38.390 -> load:0x40080404,len:3524
18:48:38.390 -> entry 0x400805b8
18:48:39.147 ->
18:48:39.478 -> WiFi connecting.....
18:48:45.984 -> WiFi connected
18:48:45.984 -> Camera Ready! Use 'http://192.168.1.181' to connect
```

Kopieer de http link in je browser om naar de CameraWebServer te gaan.
LETOP je moet op hetzelfde netwerk zitten als dat de ESP-32 Zit (de ingevulde SSID).

Hier zul je het onderstaande menu aantreffen, druk op start stream om een live Feed te bekijken.

≡ Toggle OV2640 settings

XCLK MHz	20	Set
Resolution	QVGA(320x240)	
Quality	4	63
Brightness	-2	2
Contrast	-2	2
Saturation	-2	2
Special Effect	No Effect	
AWB	<input checked="" type="checkbox"/>	
AWB Gain	<input checked="" type="checkbox"/>	
WB Mode	Auto	
AEC SENSOR	<input checked="" type="checkbox"/>	
AEC DSP	<input type="checkbox"/>	
AE Level	-2	2
AGC	<input checked="" type="checkbox"/>	
Gain Ceiling	2x	128x
BPC	<input type="checkbox"/>	
WPC	<input checked="" type="checkbox"/>	
Raw GMA	<input checked="" type="checkbox"/>	
Lens Correction	<input checked="" type="checkbox"/>	
H-Mirror	<input type="checkbox"/>	
V-Flip	<input type="checkbox"/>	
DCW (Downsize EN)	<input checked="" type="checkbox"/>	
Color Bar	<input type="checkbox"/>	
LED Intensity	0	255

Get Still **Stop Stream**

Advanced Settings

≡ Register Get/Set

≡ CLK

≡ Window



Save ×

Hier kan je verschillende instellingen aanpassen zoals de resolutie, het is handig om de hoogst mogelijke resolutie te kiezen, deze kan je later altijd nog aanpassen. Beter teveel dan te weinig.

Nu kan je een Dataset gaan maken!

Edgelmpulse adviseert om minstens 20 foto's per klasse te maken, ook hier geld beter te veel dan te weinig. Let wel op de balans van je klassen, in het optimale geval heb je evenveel fotos van iedere klasse die je hebt.

Het is handig om een map aan te maken voor de verschillende klasse die je hebt, hierdoor kan je later in edge impulse snel je data van een Label voorzien.

6. EDGE Impulse

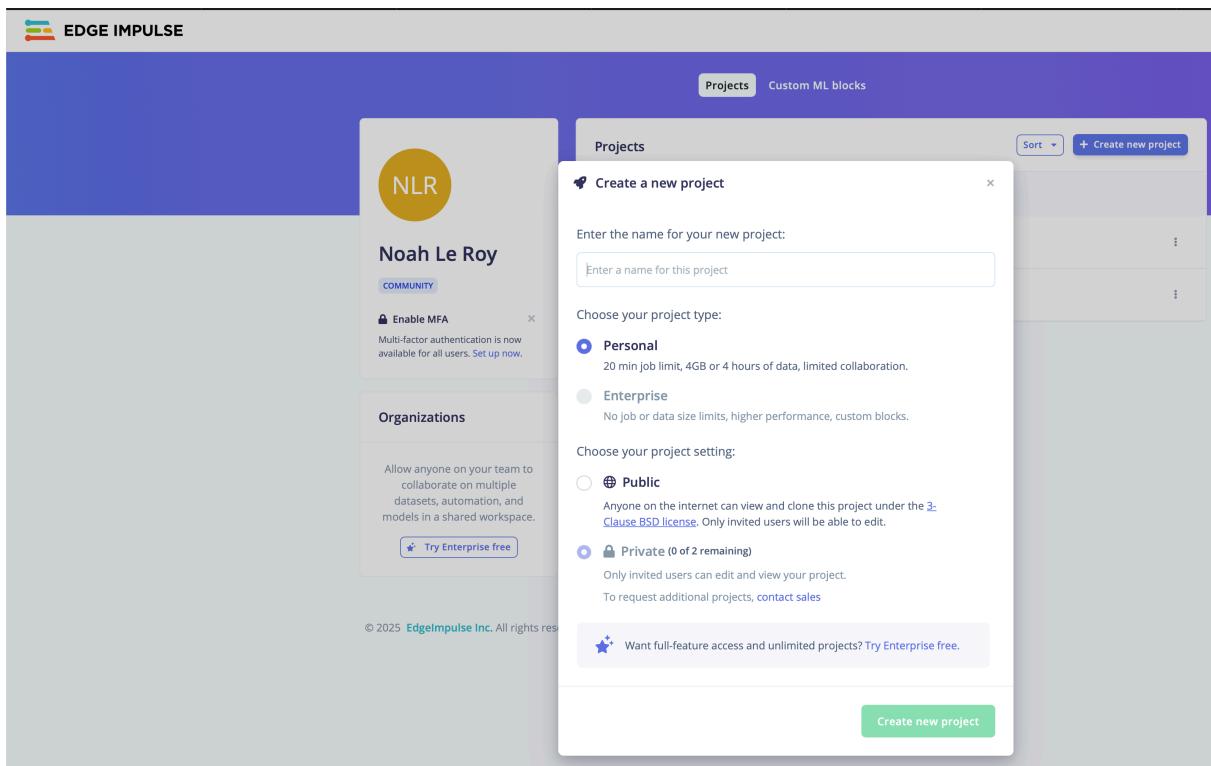
Edge Impulse is een platform waarmee je machine learning-modellen kunt ontwikkelen, trainen en inzetten op edge devices zoals microcontrollers en sensoren. Het richt zich op toepassingen in embedded AI, zoals spraakherkenning, beeldanalyse of trillingsdetectie. Wij gaan het gebruiken om een simpel image classification model mee te maken.

6.1 Project Setup

Maak een Edge Impulse account aan op edgeimpulse.com

wanneer je bent ingelogd zal je het onderstaande zien, druk hier op "Create new project" en geef

het project een naam.



Na het aanmaken van het project kom je op dit scherm.

Rechts Boven aan zie je "Target" staan. druk hier op en selecteer de ESP32-EYE en druk op "Save". Dit is niet exact onze ESP, maar Close enough!

The screenshot shows the Edge Impulse configuration interface. At the top, it says "Noah Le Roy /". Below that, there's a large blue header with the text "Configure your target device and application budget". The main content area is divided into sections:

- Target device**: A dropdown menu listing various target devices. The option "Espressif ESP-EYE (ESP32 240MHz)" is checked.
- Processor family**: A dropdown menu listing processor families. The option "Cortex-M4F 80MHz" is selected.
- Clock rate**: A dropdown menu listing clock rates. The option "Cortex-M7 216MHz" is selected.
- Custom device name (optional)**: A text input field.
- Application budget**: A section for specifying RAM and ROM requirements. It includes fields for "RAM" and "ROM".

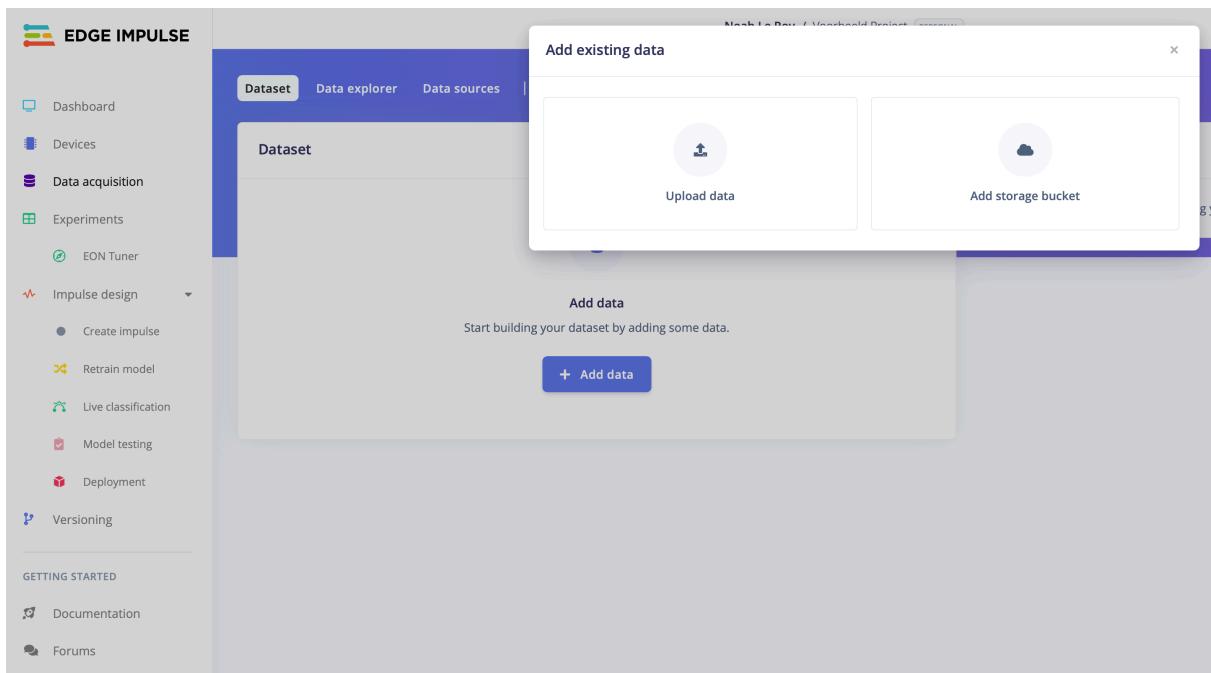
On the right side of the interface, there's a sidebar with the title "Available targets" containing a long list of supported target devices, including:

- Alif HE (Cortex-M55 160MHz + U55-128MACC)
- Alif HP (Cortex-M55 400MHz + U55-256MACC)
- Ambiq Apollo4 EVB (Cortex-M4F 192MHz)
- Ambiq Apollo5 EVB (Cortex-M55 250MHz)
- Arduino Nano 33 BLE Sense (Cortex-M4F 64MHz)
- Arduino Nicla Vision (Cortex-M7 480MHz)
- Arduino Nicla Vision M4 (Cortex-M4 240MHz)
- Arduino Portenta H7 (Cortex-M7 480MHz)
- BrainChip AKD1000 or AKD1500
- BrickML (Cortex-M33 192MHz)
- Cortex-M4F 80MHz
- Cortex-M7 216MHz
- Digi ConnectCore 93 (NXP i.MX 93 - Cortex-A55 1.7GHz + Ethos-U65-256)
- Espressif ESP-EYE (ESP32 240MHz)
- Himax WE-I (ARC DSP 400MHz)
- Himax WiseEye2 (Cortex-M55 400 MHz)
- Himax WiseEye2 (Cortex-M55 400 MHz) + Ethos-U55-64
- IMDT V2H (CPU)
- IMDT V2H (with Renesas' RZ/V2H)
- Infineon PSoC6 CY8C624 (Cortex-M4F 150 MHz)
- Infineon PSoC6 CY8C6347 (Cortex-M4F 150 MHz)
- MacBook Pro 16" 2020 (Intel Core i9 2.4GHz)
- MemryX MX3
- Microchip SAMA7G54 Evaluation Kit
- Nordic nRF52840 DK (Cortex-M4F 64MHz)
- Nordic nRF5340 DK (Cortex-M33 128MHz)
- Nordic nRF9151 DK (Cortex-M33 64MHz)
- Nordic nRF9160 DK (Cortex-M33 64MHz)
- Nordic nRF9161 DK (Cortex-M33 64MHz)
- Nvidia Jetson Nano
- Nvidia Jetson Orin NX
- Nvidia Jetson Orin Nano
- OpenMV Cam H7 Plus
- Particle Boron
- Particle Photon 2
- Qualcomm Dragonwing RB3 Gen 2 Development Kit

6.2 Data Uploaden

Nu zijn we klaar om onze Dataset te uploaden.

Ga naar "Data acquisition" aan de linker kant en druk vervolgens op "Add data"



Druk op "Upload data". Hier kan je per klasse een map uploaden en meteen van een Label voorzien zoals hieronder weergegeven.

Upload data

x

You can upload CBOR, JSON, CSV, Parquet, WAV, JPG, PNG, AVI or MP4 files. You can also upload an annotation file named "info.labels" with your data to assign bounding boxes, labels, and/or metadata. View [Uploader docs](#) to learn more. Alternatively, you can use our [Python SDK](#) to programmatically ingest data in various formats, such as pandas or numpy.

For CSV and Parquet files, [configure the CSV Wizard](#) to define how your files should be processed before uploading files.

Upload mode

Select individual files [?](#)

Select a folder [?](#)

Select files

Kies bestanden  36 bestanden

Upload into category

Automatically split between training and testing [?](#)

Training

Testing

Label

Infer from filename [?](#)

Leave data unlabeled [?](#)

Enter label:

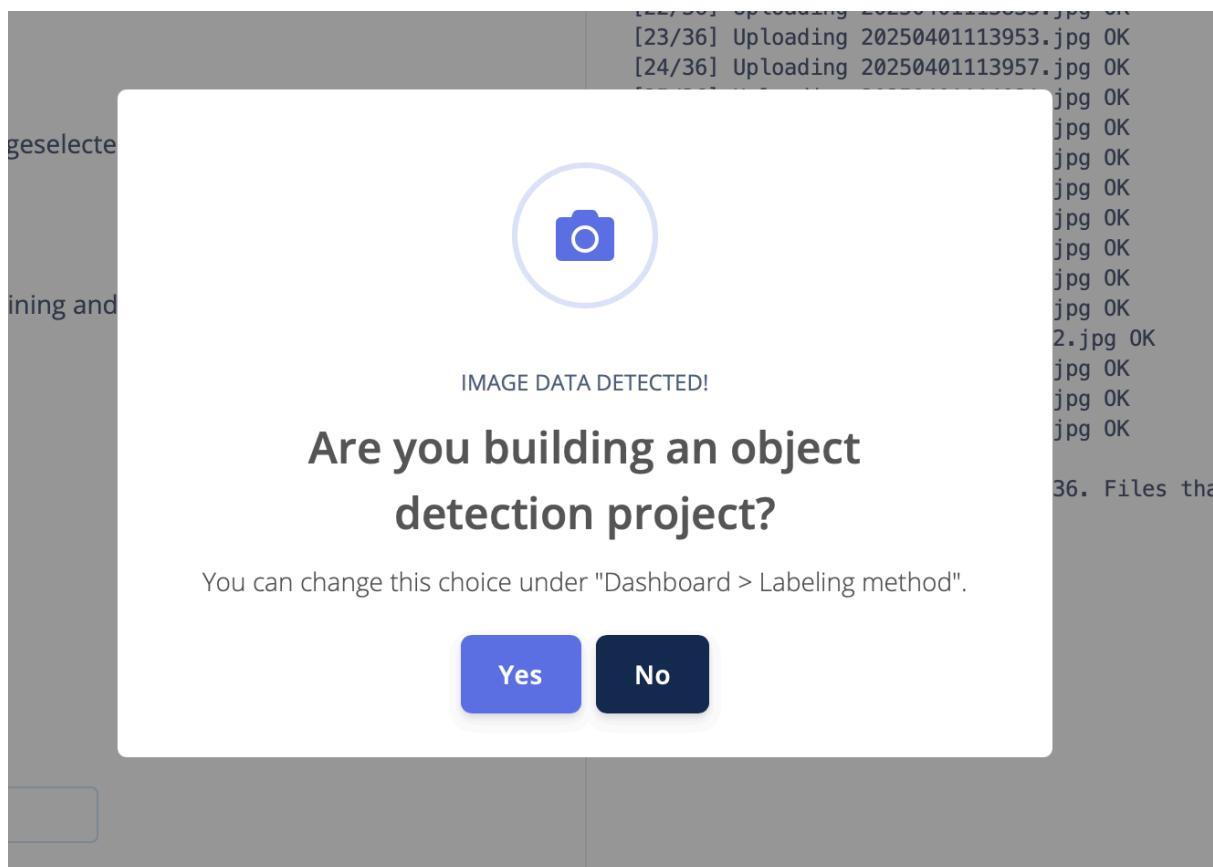
BAD

< Back

Upload data

Je kan hier zelf aangeven hoe je wil dat de Split van de data gebeurd, Automatisch is prima tenzij je zelf een Train en Test set hebt gemaakt.

Druk op "Upload data"



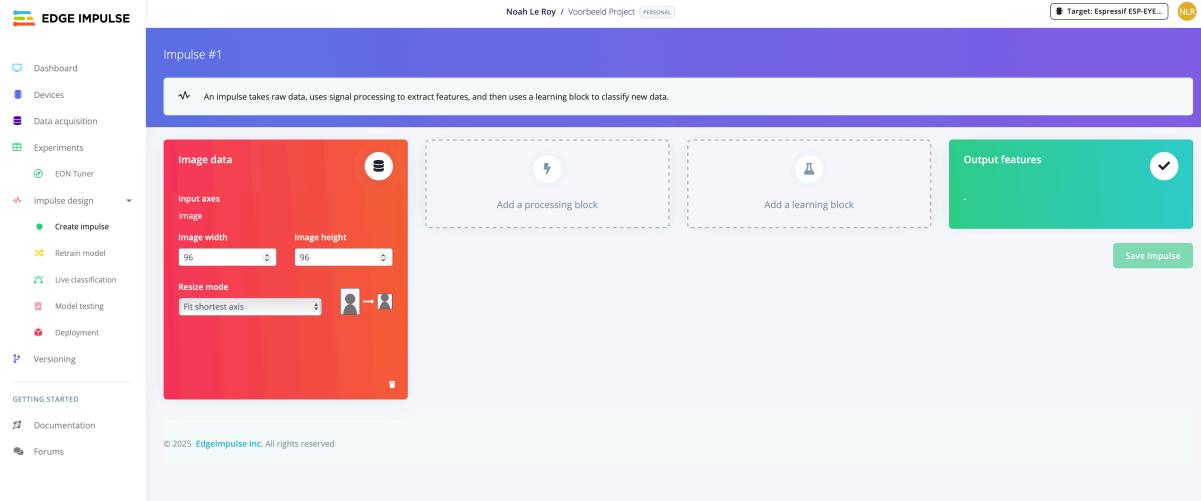
Selecteer hier "No" wij maken een image classification model, geen object detection model.

Upload nu op dezelfde manier je andere classes, Let op dat je ze het juiste Label toe kent.

6.3 Impulse Design

Nu de data is geüpload kunnen we aan het model beginnen.

ga naar "Impulse design" aan de linker kant.



Hier gaan we ons model opzetten. We werken van links naar rechts.

Eerst resizen we de afbeelding voordat deze het model in gaat, dit zorgt ervoor dat het model sneller wordt. Hoe groter de afbeelding, hoe langer het model er over doet om een nieuwe afbeelding te classificeren. ook drukken we de afbeelding samen (Squash) om zo veel mogelijk informatie te behouden als de afbeelding niet overeenkomt met de opgegeven verhouding.

Image data



Input axes

image

Image width

96

Image height

96

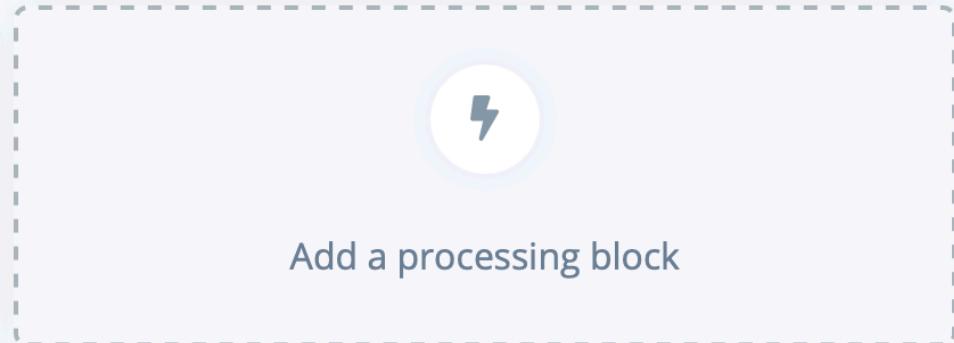
Resize mode

Squash



Druk nu op "Add a processing block"

essing to extract features, and then uses a learning block to classify new data.



Selecteer hier "Image"

⚡ Add a processing block

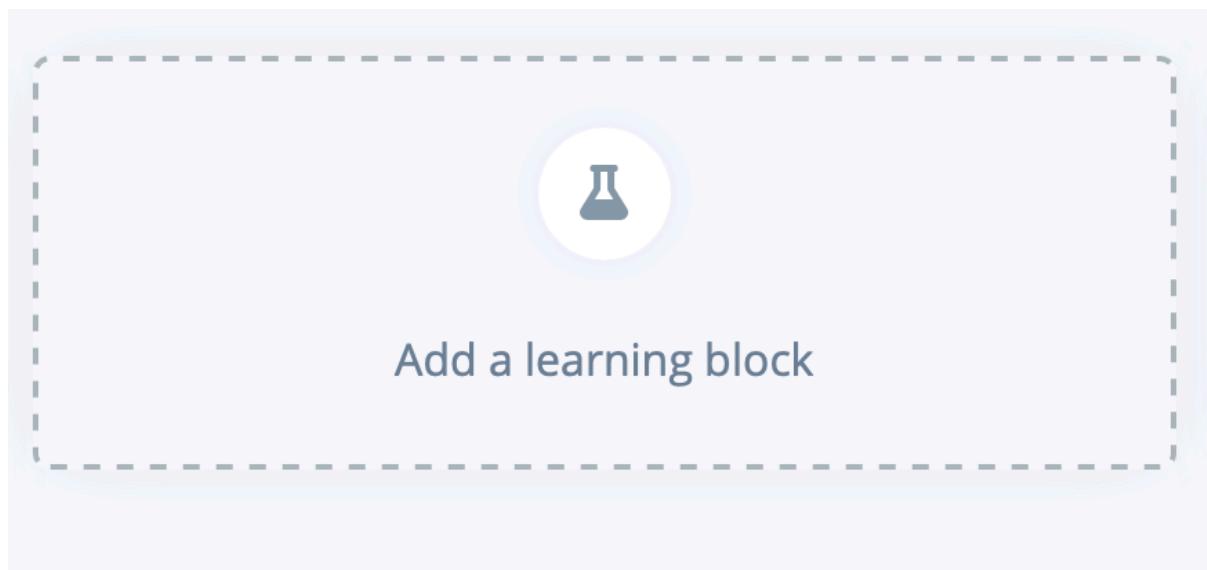
X

Did you know? You can bring your own DSP code.

DESCRIPTION	AUTHOR	RECOMMENDED
Image <small>OFFICIALLY SUPPORTED</small> Preprocess and normalize image data, and optionally reduce the color depth.	Edge Impulse	 Add
Raw Data <small>OFFICIALLY SUPPORTED</small> Use data without pre-processing. Useful if you want to use deep learning to learn features.	Edge Impulse	Add

Some processing blocks have been hidden based on the data in your project. [Show all blocks anyway](#)

Druk nu op "Add a learning block"



Selecteer hier "Transfer learning (Images)"

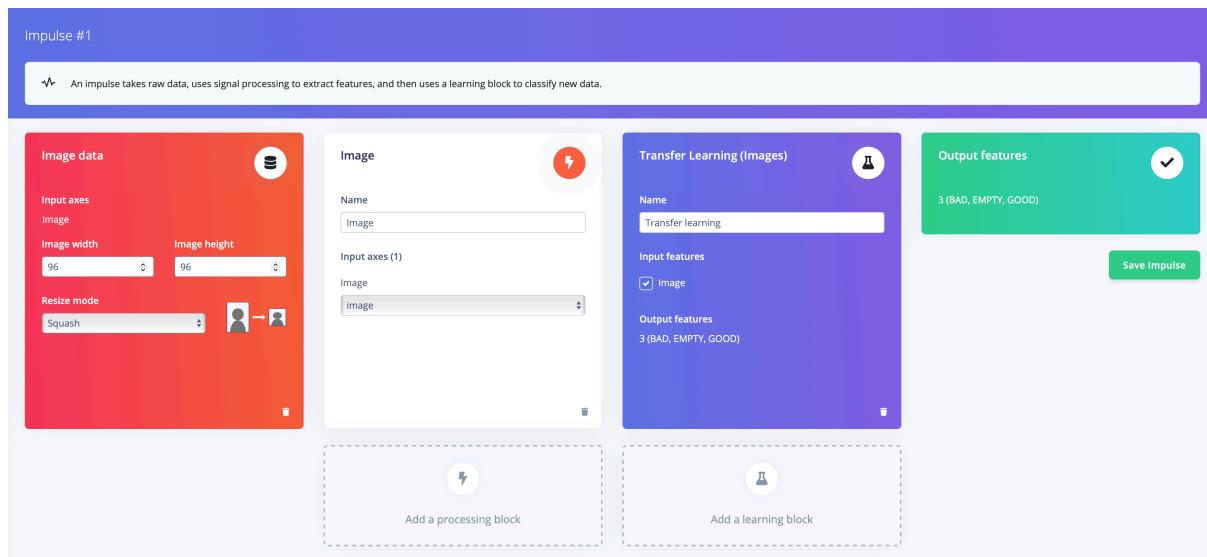
⚠ Add a learning block

X

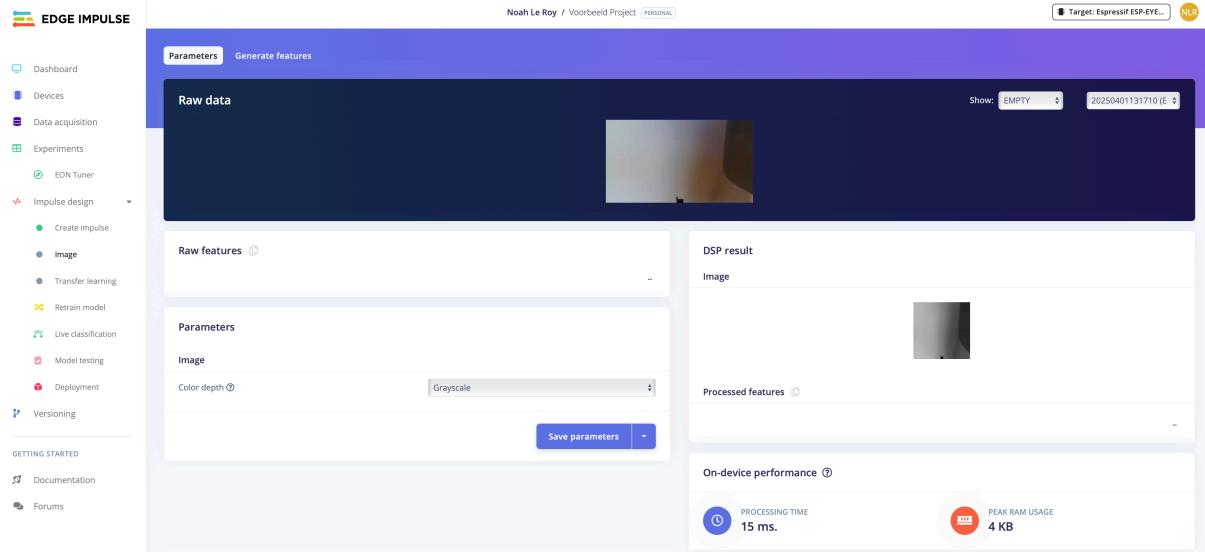
Did you know? You can bring your own model in PyTorch, Keras or scikit-learn.

DESCRIPTION	AUTHOR	RECOMMENDED
Transfer Learning (Images) <small>OFFICIALLY SUPPORTED</small> Fine tune a pre-trained image classification model on your data. Good performance even with relatively small image datasets.	Edge Impulse	★ <button>Add</button>
Classification <small>OFFICIALLY SUPPORTED</small> Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.	Edge Impulse	<button>Add</button>
Regression <small>OFFICIALLY SUPPORTED</small> Learns patterns from data, and can apply these to new data. Great for predicting numeric continuous values.	Edge Impulse	<button>Add</button>

Je Impulse is nu klaar en ziet er als het goed is zo uit

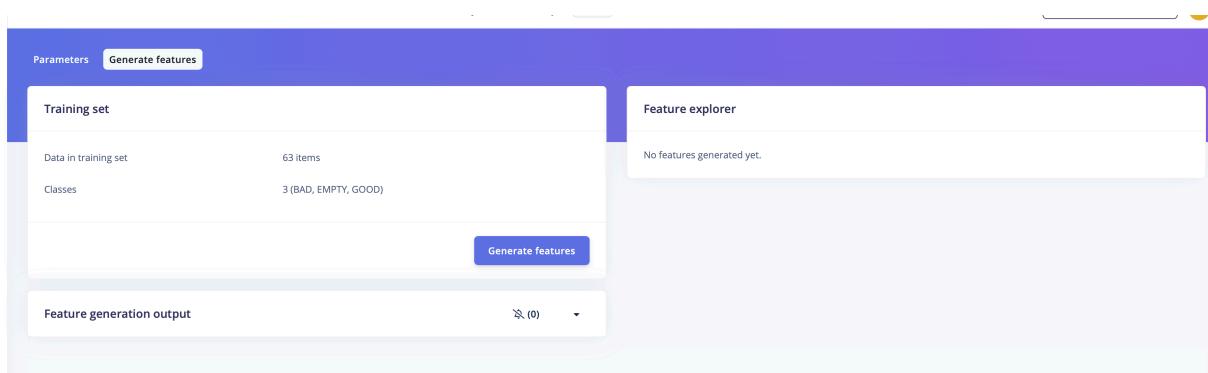


Druk nu op "Image" aan de linker kant onder "Create impulse"



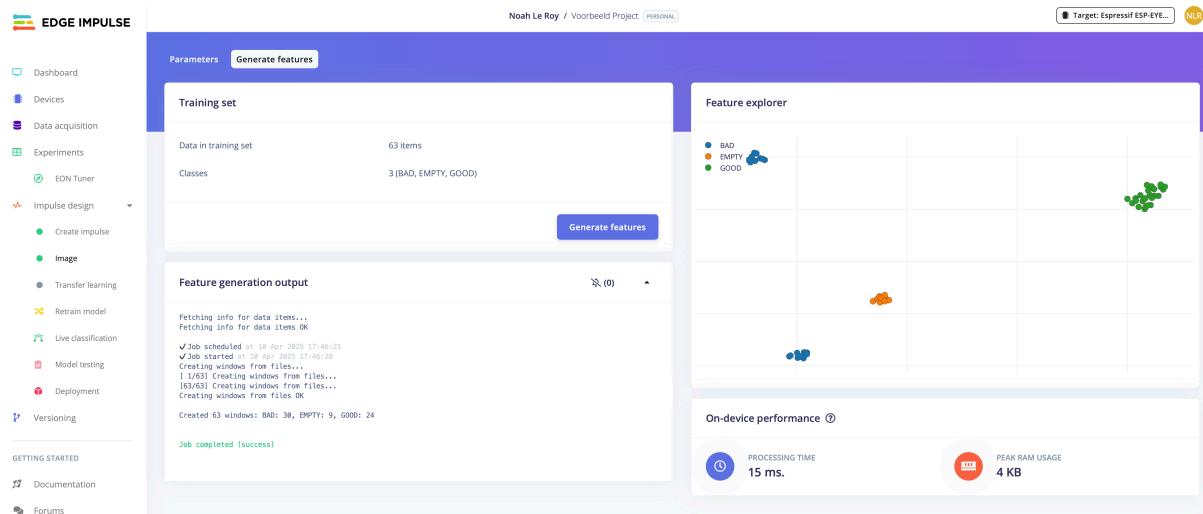
Hier kunnen we de colorspace aanpassen. "Grayscale" zal sneller werken op de ESP maar als kleur een belangrijk onderdeel is van je classificatie kan je hem op "RGB" zetten.

Druk vervolgens op "Save parameters"



Je wordt doorgestuurd naar dit scherm, hier ga je features genereren. Check of al je classes er tussen staan. in dit geval "GOOD", "BAD" en "EMPTY".

Druk op "Generate features"



Na het genereren van de Features zie je als het goed is een aantal clusters aan de rechterkant. Dit is goed!

Klik nu op "Transfer learning" aan de linker kant onder "Image".

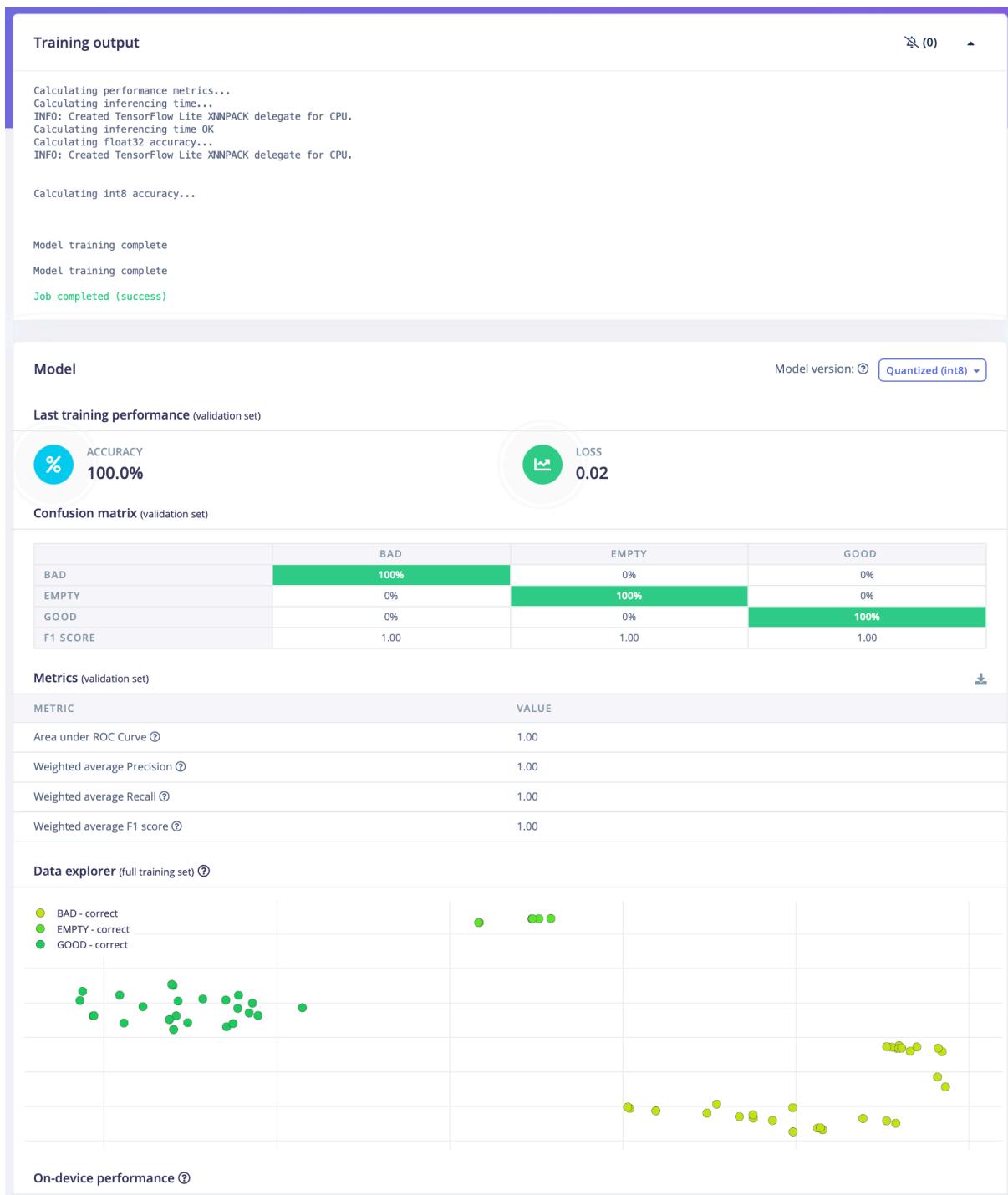
6.4 Model Trainen

In dit menu gaan we het model trainen. Je kan gewoon beginnen met de standaard instellingen, als je ziet dat het model niet genoeg leert kan je altijd proberen te spelen met de "Number of training cycles" en de "Learning rate"

Het is handig om "Data augmentation" te selecteren, dit zal je images over een aantal assen spiegelen zodat het model meer data heeft om van te leren.

Druk op "Save & train"

Het model wordt nu getraind, wacht rustig af totdat de output zegt dat het klaar is.



Hier zie je de performance op de validatie set, zoals te zien scoort dit model erg goed omdat er een duidelijk verschil in de afbeeldingen zit. Aan de clusters is te zien dat het model goed in staat is om de verschillende classes van elkaar te scheiden.

Ga nu naar "Live clasification" aan de linker kant onder "Retrain model"

EDGE IMPULSE

Noah Le Roy / Voorbeeld Project PERSONAL

Target: Express ESP-EYE

Dashboard Devices Data acquisition Experiments ION Tuner Impulse design Create impulse Image Transfer learning Retrain model Live classification Model testing Deployment Versioning

Classification result

Summary

Name	20250401131718
Label	EMPTY
CATEGORY	COUNT
BAD	0
EMPTY	1
GOOD	0
uncertain	0

Detailed result

BAD	EMPTY	GOOD
0.01	0.97	0.02

Show only unknowns

RAW DATA
20250401131718

Image

Raw features

Processed features

Upgrad Plan

Hier kan je een foto selecteren uit je dataset en kijken of deze goed geklassificeerd wordt. Je kan ook naar "Model testing" gaan om het model los te laten op de gehele test set en de resultaten te bekijken. De Test set is aangemaakt tijdens het uploaden van de data (Train Test Split).

EDGE IMPULSE

This lists all test data. You can manage this data through Data acquisition.

Test data

Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse.

SAMPLE NAME	EXPECTED OUTCOME	ACCURACY	RESULT
2025401113718	EMPTY	100%	1 EMPTY
2025401113726	EMPTY	100%	1 EMPTY
2025401113743	EMPTY	100%	1 EMPTY
2025401113706	GOOD	100%	1 GOOD
2025401113710	GOOD	100%	1 GOOD
2025401113747	GOOD	100%	1 GOOD
2025401113719	GOOD	100%	1 GOOD
2025401113737	GOOD	100%	1 GOOD
2025401113735	GOOD	100%	1 GOOD
2025401113932	BAD	100%	1 BAD
2025401114021	BAD	100%	1 BAD
2025401114018	BAD	100%	1 BAD
2025401113959	BAD	100%	1 BAD
202540111401-2	BAD	100%	1 BAD
2025401114013	BAD	100%	1 BAD

Model testing output

Classifying data for `float32 model_1`.
Job scheduled: 2025-01-10T10:00:18Z
Job started: 2025-01-10T10:00:18Z
Model: TensorFlow Lite MNPACK delegate for GPU.

Classifying data for Transfer learning DK

Generating model testing summary...
Generating transfer learning summary...
Job completed [Success]

Results

Model version: `Unprinted (first)`

Metrics for Transfer learning

METRIC	VALUE
Area under ROC Curve ⓘ	1.00
Weighted average Precision ⓘ	1.00
Weighted average Recall ⓘ	1.00
Weighted average F1 score ⓘ	1.00

Confusion matrix

	BAD	EMPTY	GOOD	UNCERTAIN
BAD	100%	0%	0%	0%
EMPTY	0%	100%	0%	0%
GOOD	0%	0%	100%	0%
F1 SCORE	1.00	1.00	1.00	

Feature explorer ⓘ

Legend: BAD - correct, EMPTY - correct, GOOD - correct

Als je tevreden bent met het resultaat is het tijd om het model te deployen op de ESP!

Klik op "Deployment" aan de linker kant onder "Model Testing"

7. Deployment

The screenshot shows the Edge Impulse web interface. On the left, there is a sidebar with various project management and development tools. The main area is titled "Configure your deployment". It contains a search bar and a list of deployment options:

- C++ library**: A portable C++ library with no external dependencies, which can be compiled with any modern C++ compiler.
- Arduino library**: An Arduino library with examples that runs on most Arm-based Arduino development boards. This option is selected, indicated by a checked checkbox.
- BrainChip MetaTF Model for Akida - AKD1000 and AKD1500**: A MetaTF converted model (.fbz) for use with the BrainChip Akida™ runtime.
- Cube.MX CMSIS-PACK**: A STM32Cube.MX CMSIS-PACK, for fast inferencing on many ST MCUs.
- Custom block**: A ZIP file containing everything your custom deploy block will receive.
- DRP-AI Library**: Generate machine learning models to use the DRP-AI accelerator on Renesas RZ/ products.

Below the list, there are two tables comparing different deployment configurations:

	LATENCY	IMAGE	TRANSFER LEARNING	TOTAL
Selected (float32)	15 ms. RAM 4.0K FLASH - ACCURACY	1,663 ms. 334.6K 585.5K	-	1,678 ms. 334.6K
Unoptimized (float32)	LATENCY RAM 4.0K FLASH -	4,141 ms. 893.7K 1.6M	-	4,156 ms. 893.7K

Druk op de zoekbalk en selecteer "Arduino library"

Configure your deployment

You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption. [Read more.](#)

Arduino library X

SELECTED DEPLOYMENT
Arduino library
 An Arduino library with examples that runs on most Arm-based Arduino development boards.

MODEL OPTIMIZATIONS
 Model optimizations can increase on-device performance but may reduce accuracy.

EON™ Compiler
 Same accuracy, 17% less RAM, 15% less ROM.

Quantized (int8)		IMAGE	TRANSFER LEARNING	TOTAL
<input checked="" type="checkbox"/> Selected	LATENCY	15 ms.	1,663 ms.	1,678 ms.
	RAM	4.0K	334.6K	334.6K
	FLASH	-	585.5K	-
	ACCURACY	-	-	-

Unoptimized (float32)		IMAGE	TRANSFER LEARNING	TOTAL
<input type="checkbox"/> Select	LATENCY	15 ms.	4,141 ms.	4,156 ms.
	RAM	4.0K	893.7K	893.7K
	FLASH	-	1.6M	-
	ACCURACY	-	-	100.00%

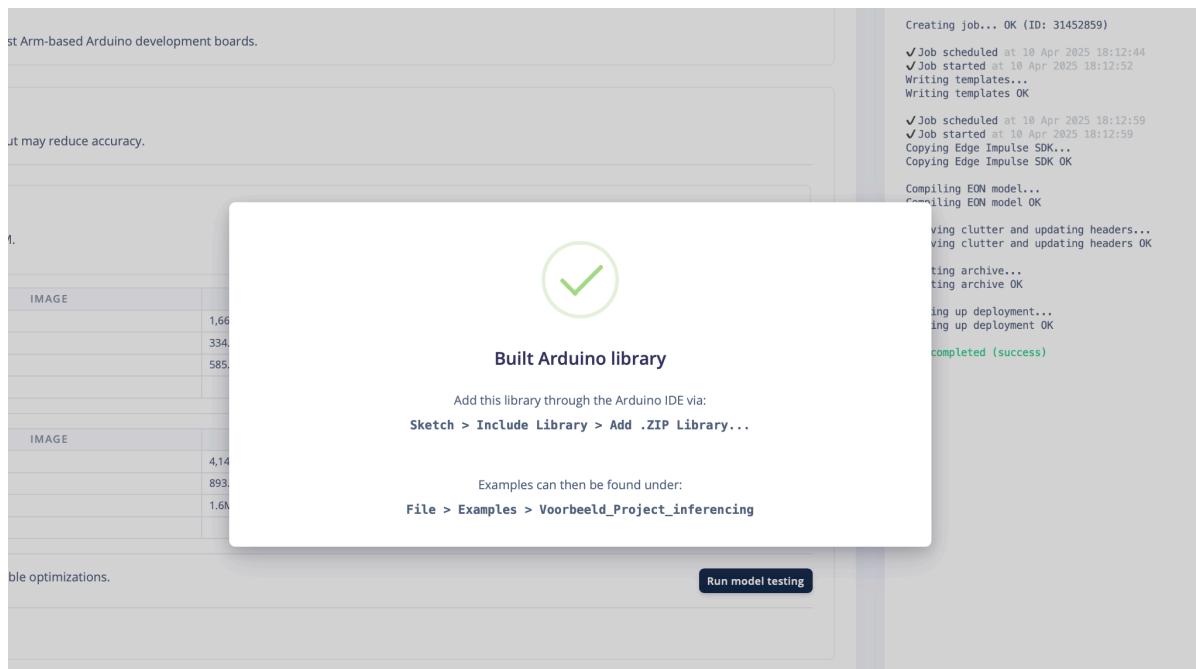
To compare model accuracy, run model testing for all available optimizations. Run model testing

Estimate for Espressif ESP-EYE (ESP32 240MHz) - [Change target](#)

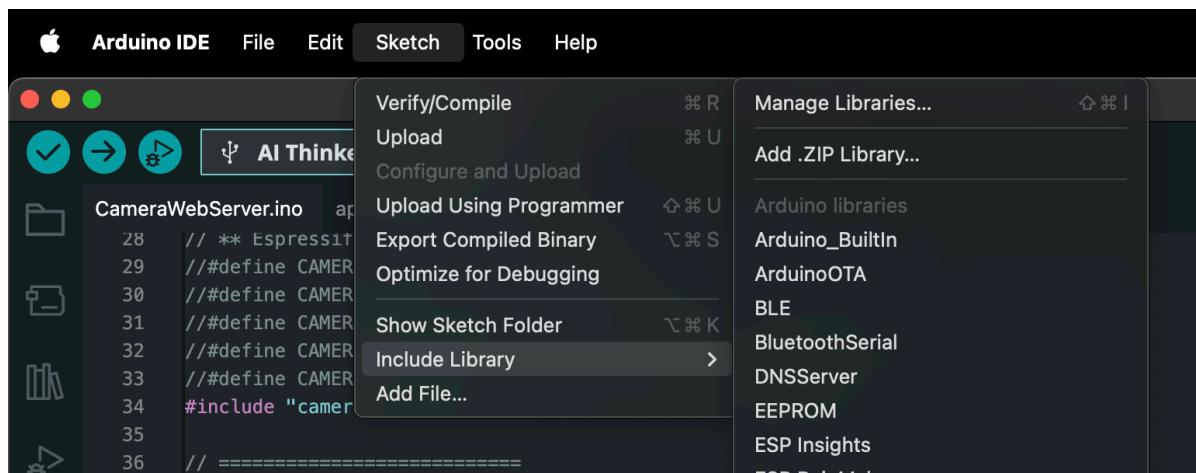
Build

Hier kan je kiezen of je het model wil Quantizen of niet. Een Quantized model zal sneller draaien op de ESP maar mogelijk minder accuraat zijn.

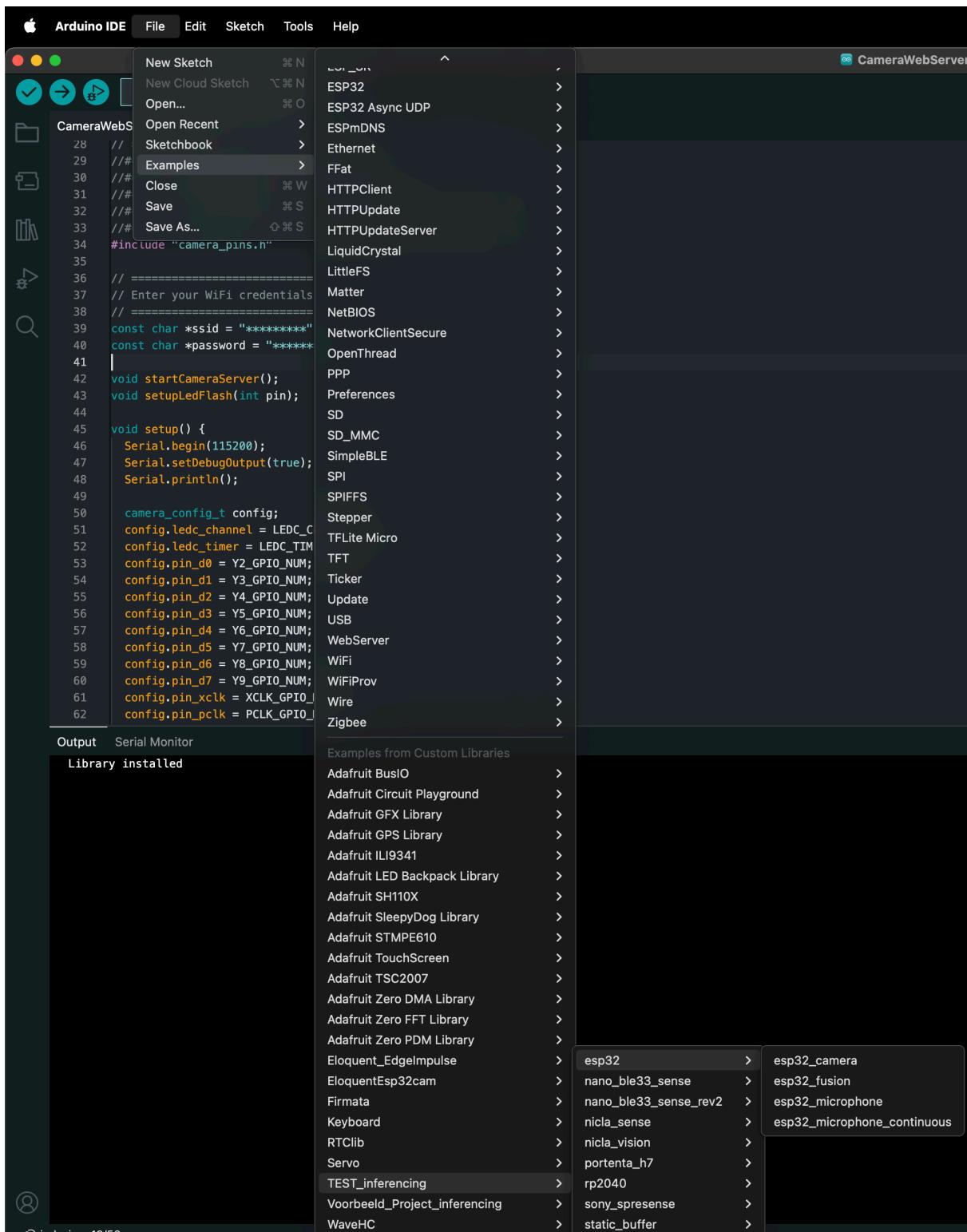
Druk op "Build", er zal nu een .ZIP gedownload worden die je kan importeren in de Arduino IDE



Doe wat Edge Impulse zegt en druk op Add .ZIP Library in het menu



Na het importeren van de Library kan je een Example inladen onder:
 File > Examples > Naam van je project (Helemaal onderaan) > esp32 > esp32_camera



De nieuwe Sketch zal nu openen.

LET OP: op line 36 moeten we nu de juiste ESP selecteren. Dit doe je door

" // " voor `#define CAMERA_MODEL_ESP_EYE` te zetten en deze slashes weg te halen
VOOR `#define CAMERA_MODEL_AI_THINKER`

```
esp32_camera.ino
1  /* Edge Impulse Arduino examples
2   * Copyright (c) 2022 EdgeImpulse Inc.
3   *
4   * Permission is hereby granted, free of charge, to any person obtaining a copy
5   * of this software and associated documentation files (the "Software"), to deal
6   * in the Software without restriction, including without limitation the rights
7   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
8   * copies of the Software, and to permit persons to whom the Software is
9   * furnished to do so, subject to the following conditions:
10  *
11  * The above copyright notice and this permission notice shall be included in
12  * all copies or substantial portions of the Software.
13  *
14  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
15  * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
16  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
17  * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
18  * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
19  * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
20  * SOFTWARE.
21  */
22
23 // These sketches are tested with 2.0.4 ESP32 Arduino Core
24 // https://github.com/espressif/arduino-esp32/releases/tag/2.0.4
25
26 /* Includes ----- */
27 #include <Voorbeeld_Project_inferencing.h>
28 #include "edge-impulse-sdk/dsp/image/image.hpp"
29
30 #include "esp_camera.h"
31
32 // Select camera model - find more camera models in camera_pins.h file here
33 // https://github.com/espressif/arduino-esp32/blob/master/libraries/ESP32/examples/Camera/CameraWebServer/camera\_pins.h
34
35 //#define CAMERA_MODEL_ESP_EYE // Has PSRAM
36 #define CAMERA_MODEL_AI_THINKER // Has PSRAM
37
```

That's it! Nu kunnen we de Sketch flashen naar de ESP zoals we gedaan hebben met de CameraWebServer. Het model zal nu de voorspellingen weergeven in de Serial Monitor.

Output Serial Monitor X

Message (Enter to send message to 'AI Thinker ESP32-CAM' on '/dev/cu.usbserial-210')

New Line 115200 baud

```

20:30:22.239 -> GOOD: 0.169/
20:30:23.554 -> Predictions (DSP: 6 ms., Classification: 1123 ms., Anomaly: 0 ms.):
20:30:23.554 -> Predictions:
20:30:23.554 -> BAD: 0.17578
20:30:23.554 -> EMPTY: 0.68750
20:30:23.554 -> GOOD: 0.13672
20:30:24.866 -> Predictions (DSP: 6 ms., Classification: 1123 ms., Anomaly: 0 ms.):
20:30:24.866 -> Predictions:
20:30:24.866 -> BAD: 0.14453
20:30:24.866 -> EMPTY: 0.73438
20:30:24.866 -> GOOD: 0.12109
20:30:26.183 -> Predictions (DSP: 6 ms., Classification: 1123 ms., Anomaly: 0 ms.):
20:30:26.183 -> Predictions:
20:30:26.183 -> BAD: 0.17969
20:30:26.183 -> EMPTY: 0.66406
20:30:26.183 -> GOOD: 0.15625
20:30:27.500 -> Predictions (DSP: 6 ms., Classification: 1123 ms., Anomaly: 0 ms.):
20:30:27.500 -> Predictions:
20:30:27.500 -> BAD: 0.31641
20:30:27.500 -> EMPTY: 0.57031
20:30:27.500 -> GOOD: 0.11328

```

Ln 84, Col 1 AI Thinker ESP32-CAM on /dev/cu.usbserial-210 2

```

185     return;
186 }
187
188 // print the predictions
189 ei_printf("Predictions (DSP: %d ms., Classification: %d ms., Anomaly: %d ms.): \n",
190             result.timing.dsp, result.timing.classification, result.timing.anomaly);
191
192 #if ET_CLASSIFIER_OBJECT_DETECTION == 1
193

```

Deze output wordt geprint op line 190 binnen de `void loop()` van de onaangepaste file (kan mogelijk veranderen).

Binnen de `void loop()` kan je nu aan de slag met het " `result`" om eventueel acties te ondernemen aan de hand van de output van het model.

Gefeliciteerd! Je hebt nu een werkend model op je ESP32-CAM gemaakt met Edge Impulse.