



**Knight Foundation  
School of Computing  
and Information Sciences**

# Using AI with a Low-Cost Camera to Detect Harmful Algae in Natural Water

Team Members: Kiran Brahmawari, Cristian Cruz, Justin Prater

Product Owner: Dr. Antao Chen, Beckman Coulter

Instructor: Masoud Sadjadi

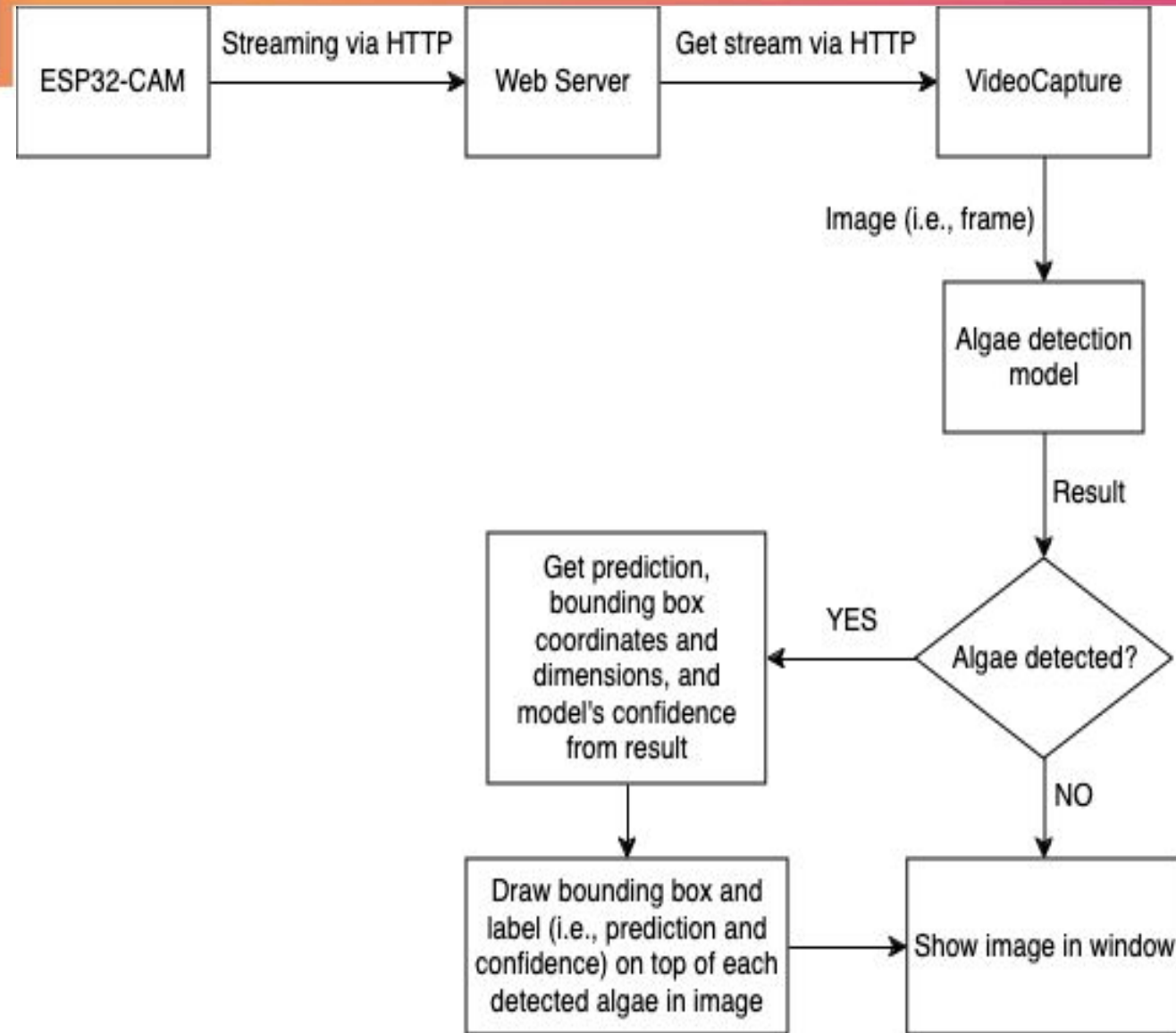
# Motivation

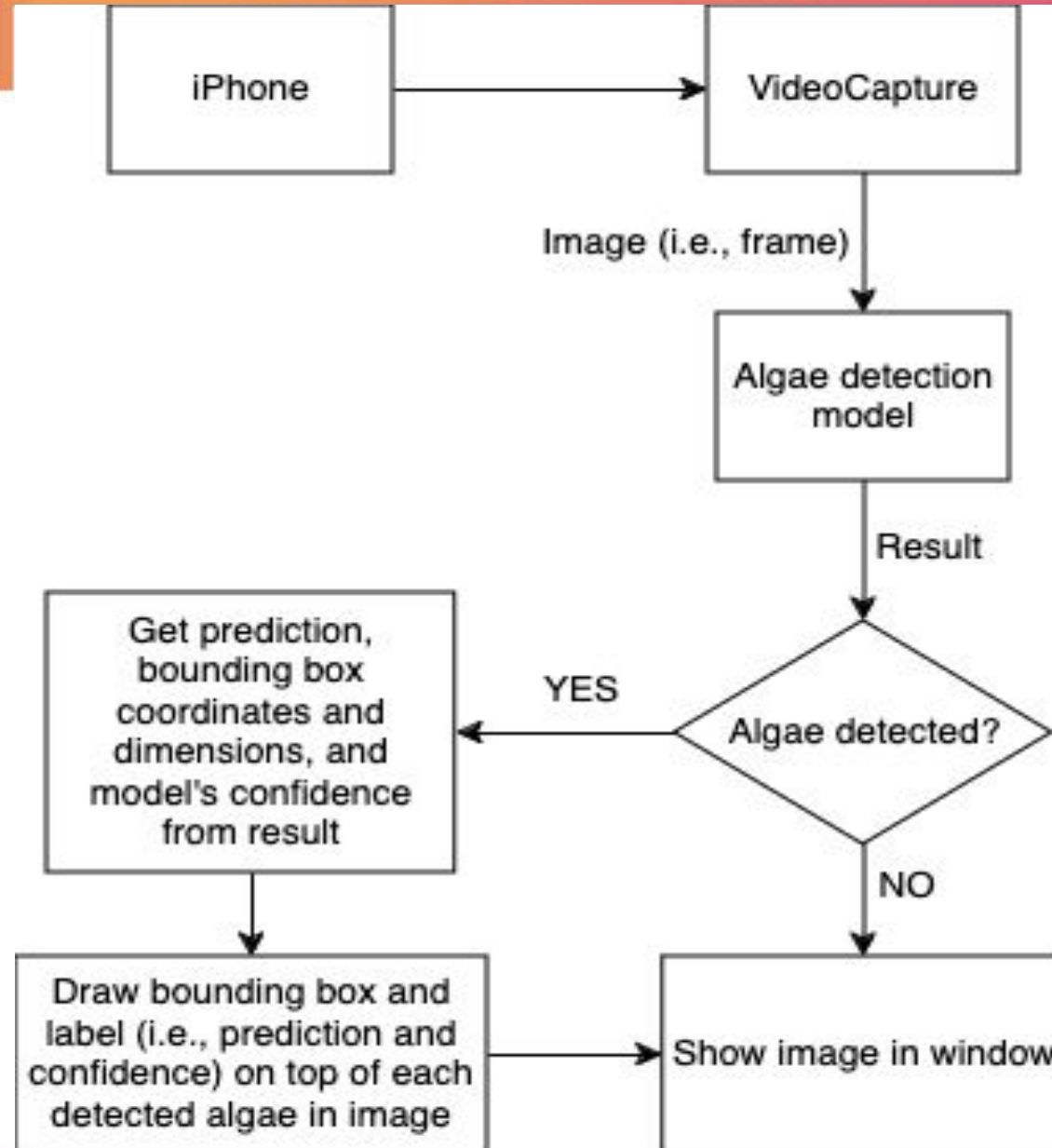
- Harmful algae blooms can be detrimental to the environment when undergoing uncontrolled population growth
- These blooms deplete environments of oxygen, killing fish, harming humans, and disrupting local economies
- With an easier means to locate and detect some of these classes of algae, it would alleviate the burden of classifying algae when trying to search for the appropriate solution

## Solution

- Using the inexpensive and accessible ESP32-CAM, a microscope, and our product, different types of algal blooms can be detected rapidly and dealt with accordingly
- The AI models were trained using the YOLOv8 and PyTorch framework, and implemented using Python

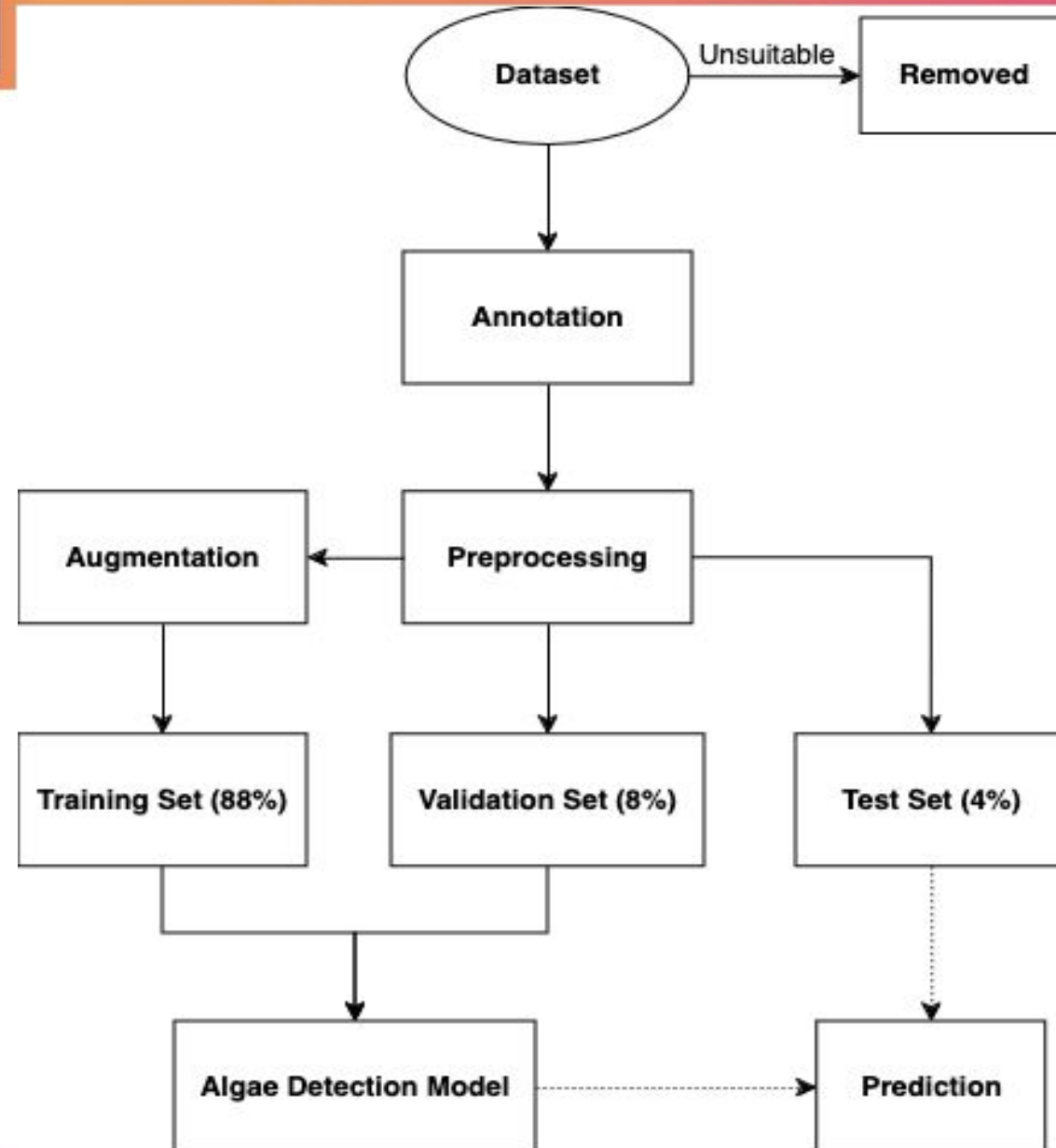




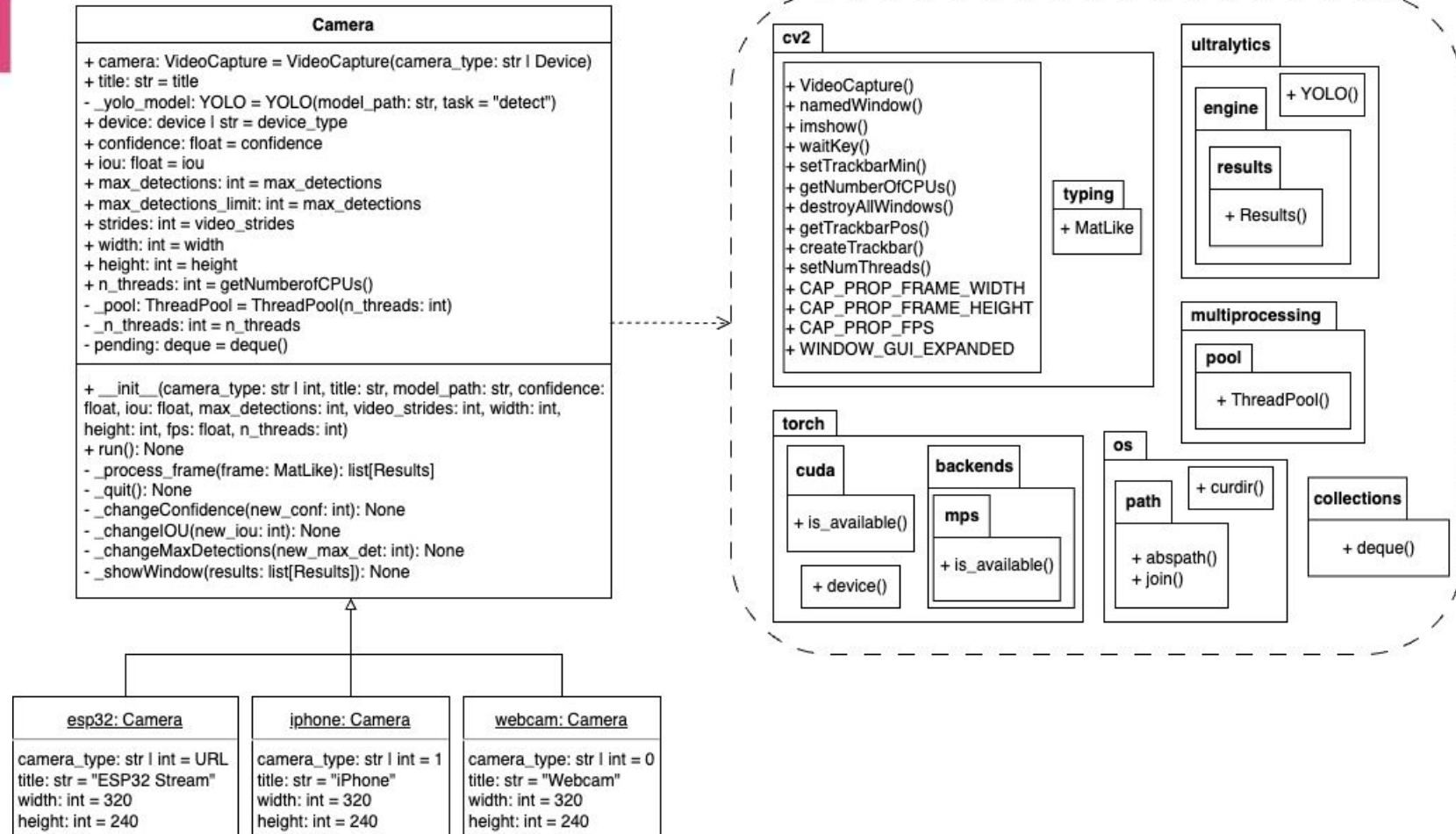




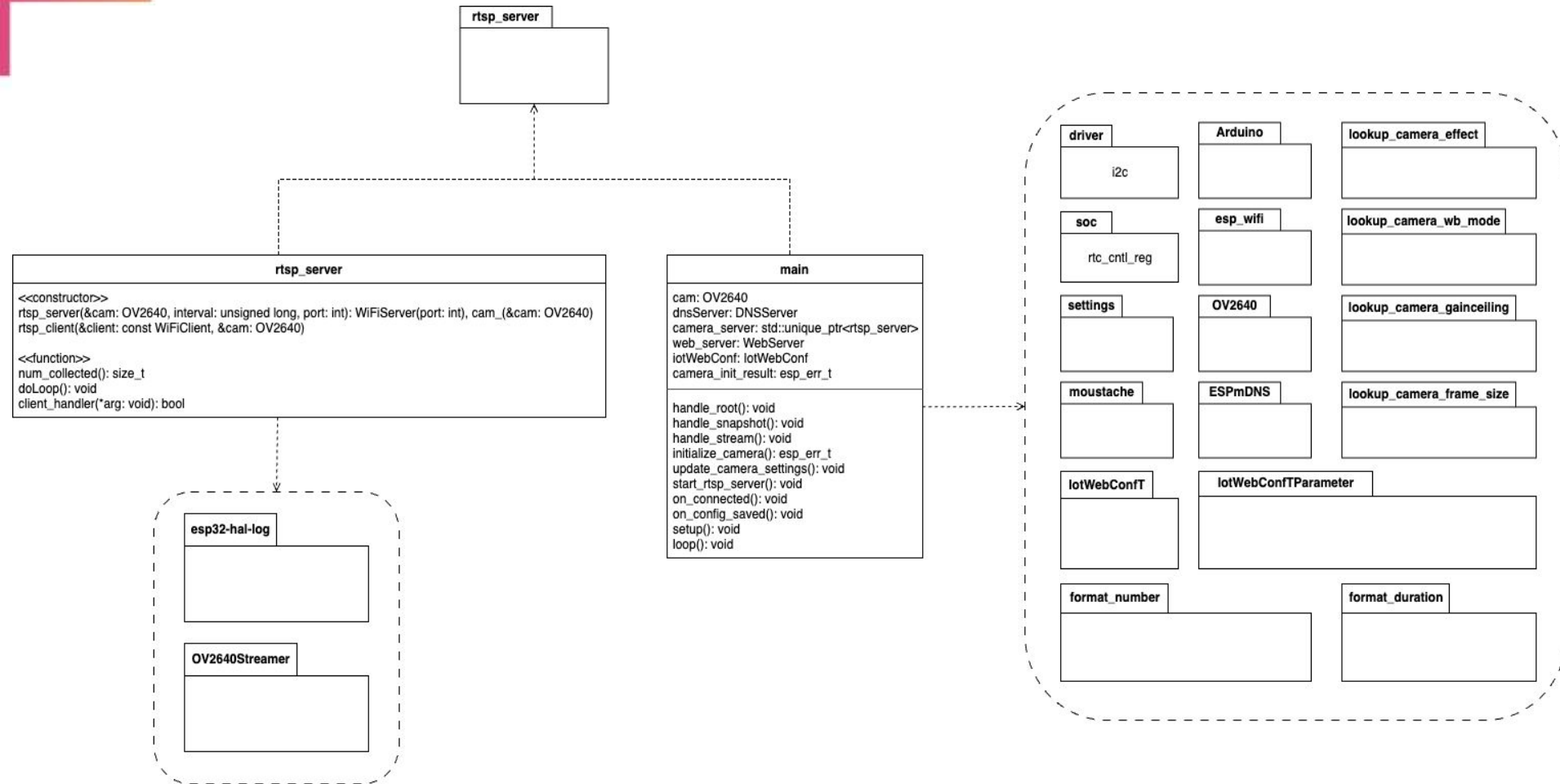
## Solution Flowchart: Dataset



## Solution UML: src/detection



## Solution UML: src/streaming

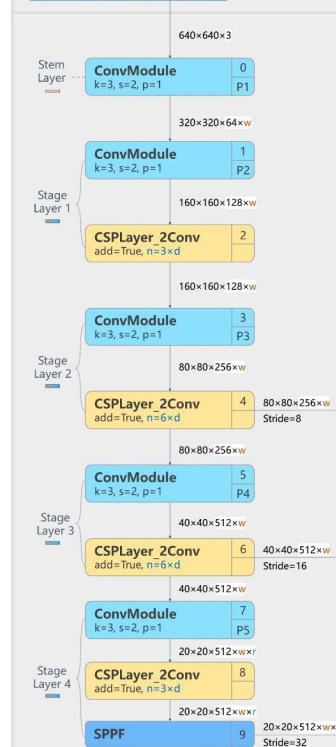
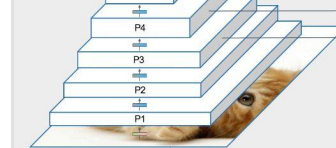




# Solution Architecture: YOLOv8

## YOLOv8

**Backbone**  
YOLOv8  
CSPDarknet  
(P5)

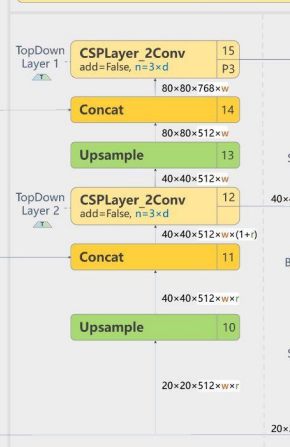
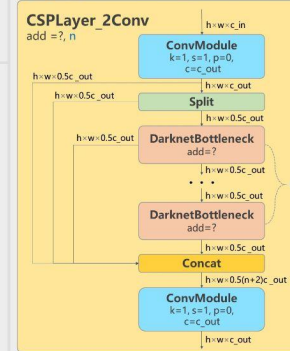


Backbone

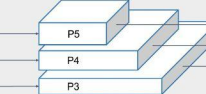
**Neck**  
YOLOv8PAFPN



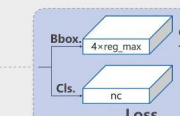
### Details



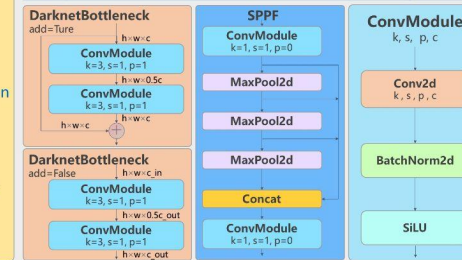
**Head** YOLOv8HeadModule



Decoupled Head  
Decoupled Head  
Decoupled Head



model	d (deepen_factor)	w (widen_factor)	r (ratio)
n	0.33	0.25	2.0
s	0.33	0.50	2.0
m	0.67	0.75	1.5
l	1.00	1.00	1.0
x	1.00	1.25	1.0



**Note:**  
1. The numbers on the connecting line stand for height\*width\*channel  
2. 'CSPLayer 2Conv' stands for 'CSPLayerWithTwoConv' in MMYOLO repo.  
3. Since the number of output channels in the last stage of different sizes of models is different, (ratio) is used in this figure for convenience. In MMYOLO repo, 'last stage\_out\_channels' is used to control the number.

Neck

Head

# Dataset

- Images
  - Around 2000
  - Downscaled to  $240 \times 320$
  - Subset used for training
- Classes
  - Closterium
  - Nitzschia
  - Microcystis
  - Oscillatoria
  - Non Algae



Figure 1. Image of Closterium

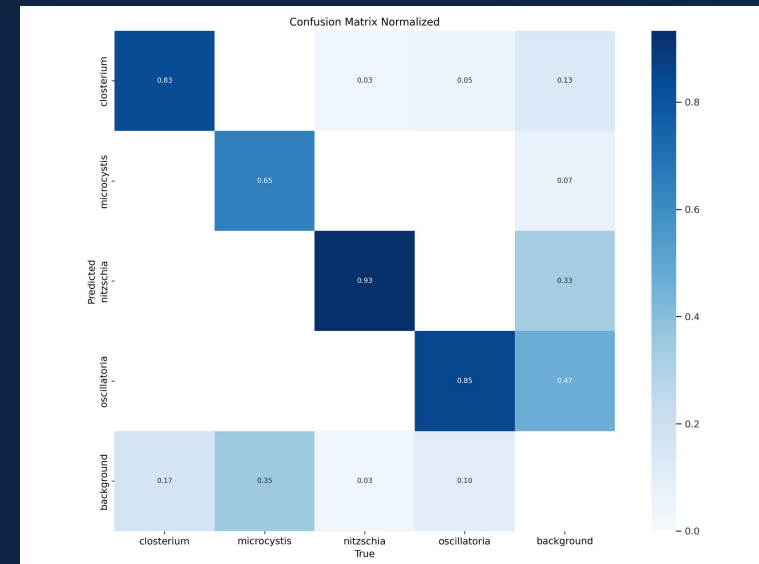
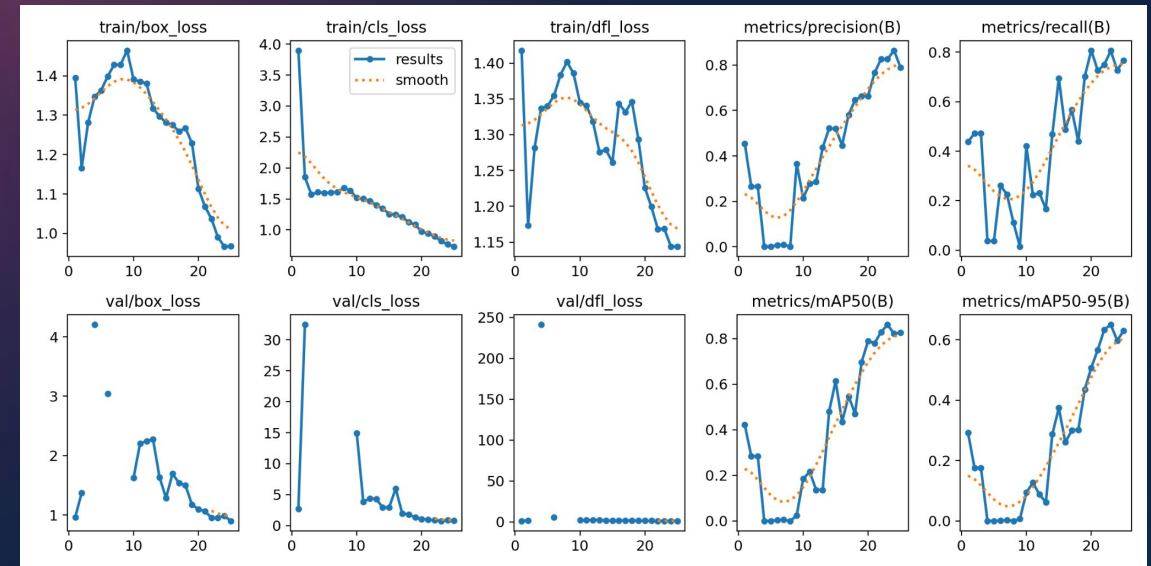


Figure 2. Image of Microcystis

# Model Development: YOLOv8x

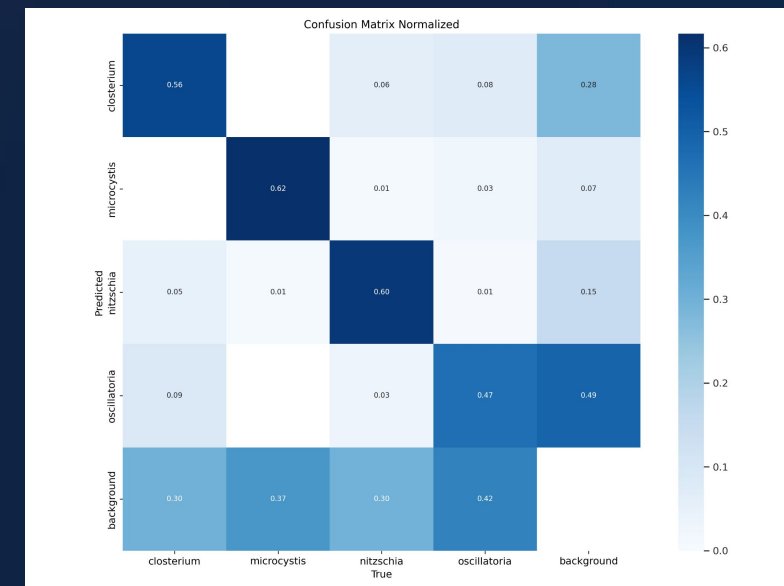
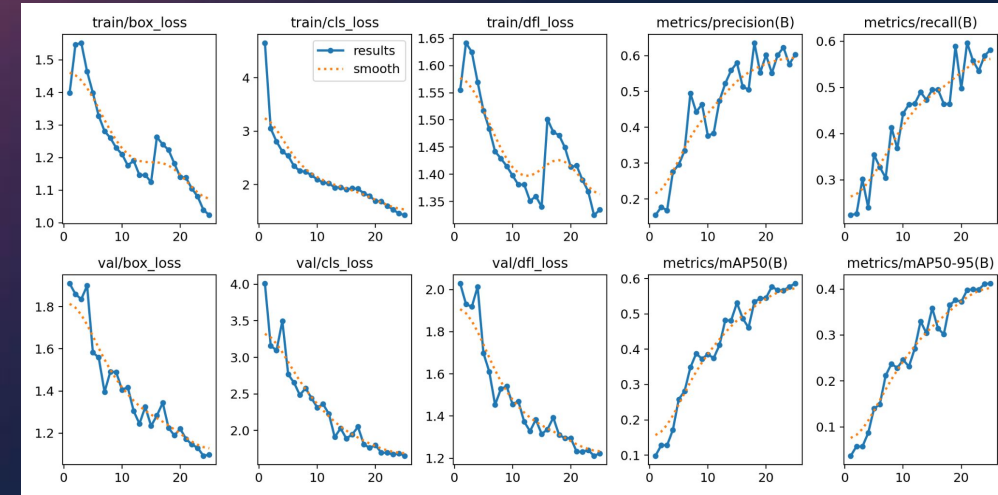
- Split
  - Training: 0.88
  - Validation: 0.08
  - Testing: 0.04
- Epochs: 25
- Optimizer: Adam

Note: This is the most recently trained version (i.e., v2.0)



# Model Development: YOLOv8n + SAHI

- Split
  - Training: 0.88
  - Validation: 0.08
  - Testing: 0.04
- Epochs: 25
- Optimizer: Adam



# Sample Output





# User Interface



Press on the button below to change the settings

[Change settings](#)**ESP32**

Board type: esp32cam\_ai\_thinker  
SDK Version: v4.4.6-dirty  
CPU model: ESP32-D0WDQ6-V3 rev. 3  
CPU speed: 240 Mhz  
CPU cores: 2  
RAM size: 267.08 KB  
PSRAM size: 4 MB  
Flash size: 4 MB

**Diagnostics**

Uptime: 00:18:40  
RTSP sessions: 0  
Free heap: 149.35 KB  
Max free block: 107.99 KB

**Network**

Host name: esp32-08f9e0d374b0.local  
Mac address: 08:F9:E0:D3:74:B0  
Wifi mode: STA  
Access point: Jbrahma  
Signal strength: -67 dbm  
IPv4 address: 10.0.0.114  
IPv6 address: 0000:0000:0000:0000:0000:0000:0000:0000

Connected to the access point

**Camera**

Frame rate: 200 ms (5.0 f/s)  
Frame size: QVGA (320x240)  
JPEG quality: 1 [1-100]  
Brightness: 0 [-2,2]  
Contrast: 0 [-2,2]  
Saturation: 0 [-2,2]  
Special effect: Normal  
White balance: Auto  
AWB gain: Auto  
WB mode: Auto  
Exposure control: Auto  
Auto exposure control (dsp): Enabled  
Auto exposure level: 0  
Manual exposure value: 300  
Gain control: Auto  
AGC gain: 0  
Gain ceiling: 2X  
Black pixel correct: Manual  
White pixel correct: Auto  
Gamma correct: Enabled  
Lens correction: Enabled  
Horizontal mirror: Normal  
Vertical flip: Normal  
Downsize enable: Enabled  
Color bar: Camera

Camera was initialized successfully!

**Special URLs / API**

RTSP camera stream: <rtsp://10.0.0.114:554/mjpeg/1>  
JPEG Motion stream: <http://10.0.0.114/stream>  
Snapshot of the camera: <http://10.0.0.114/snapshot>

# User Interface: ESP32-CAM

- Streaming server's home page
- Includes hardware, diagnostics, network, and camera information
- Includes links to HTTP server, RTSP server, and snapshot

System configuration

Thing name  
ESP32-CAM AI Thinker

AP password

WiFi SSID

WiFi password

Startup delay (seconds)  
15

Camera settings

Frame duration (ms)  
200

Frame size  
QVGA (320x240)

JPG quality  
1

Brightness  
0

Contrast  
0

Saturation  
0

Effect  
Normal

White balance ☒

Automatic white balance gain ☒

White balance mode  
Auto

Exposure control ☒

Auto exposure (dsp) ☒

Auto Exposure level  
0

Manual exposure value  
300

Gain control ☒

AGC gain  
0

Auto Gain ceiling  
2X

Black pixel correct ☐

White pixel correct ☒

Gamma correct ☒

Lens correction ☒

Horizontal mirror ☐

Vertical mirror ☐

Downsize enable ☒

Colorbar ☐

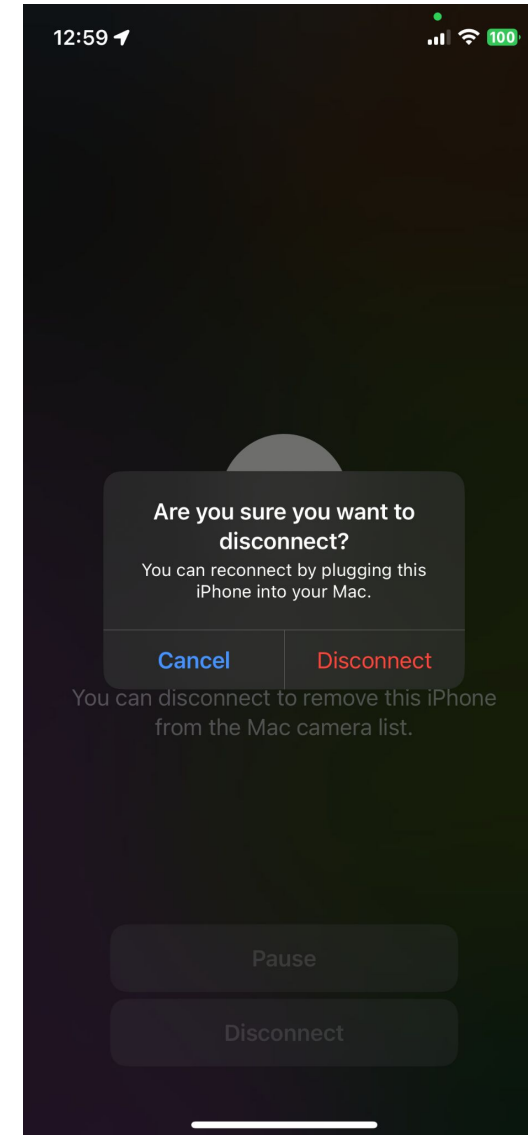
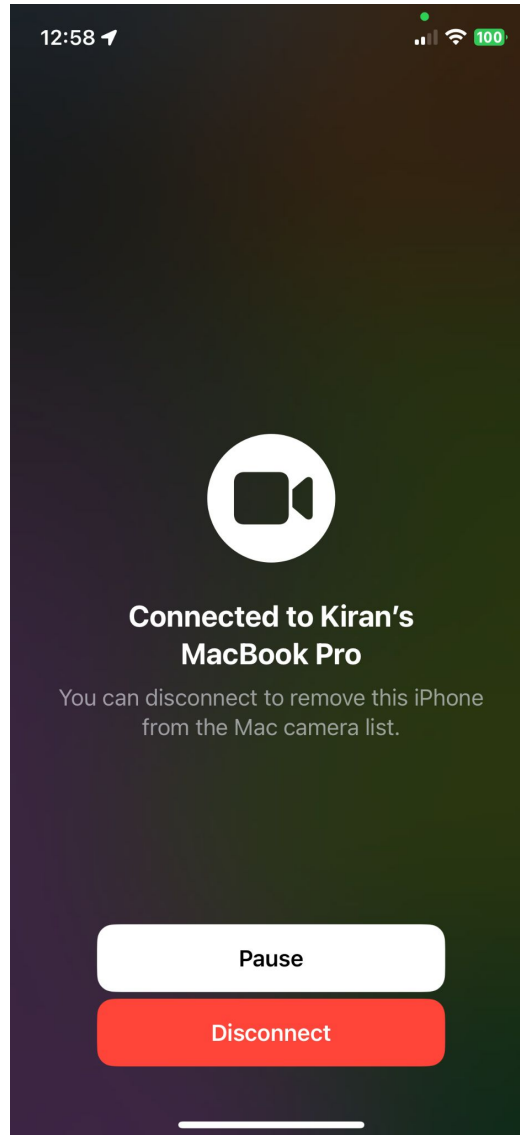
Apply

Firmware config version '1.6'

# User Interface: ESP32-CAM

- ESP32-CAM's configuration options
  - Brightness
  - Contrast
  - Saturation
  - Stream quality
  - Exposure
  - Frame rate
  - Frame size
  - White balance mode
  - And more

# User Interface: iPhone



## Conclusion

- A convolutional neural network was trained on a large dataset of closterium, nitzschia, microcystis, oscillatoria, and non-algae
- This algae detection model allows users to quickly and easily classify algae with the ESP32-CAM, or even their smartphone
- Flexibility has greatly increased due to the model switching from a classification model to an object detection model
- The best focus for future work would be to increase the dataset drastically and to engineer a heatsink for the ESP32-CAM to mitigate overheating, thus improving performance