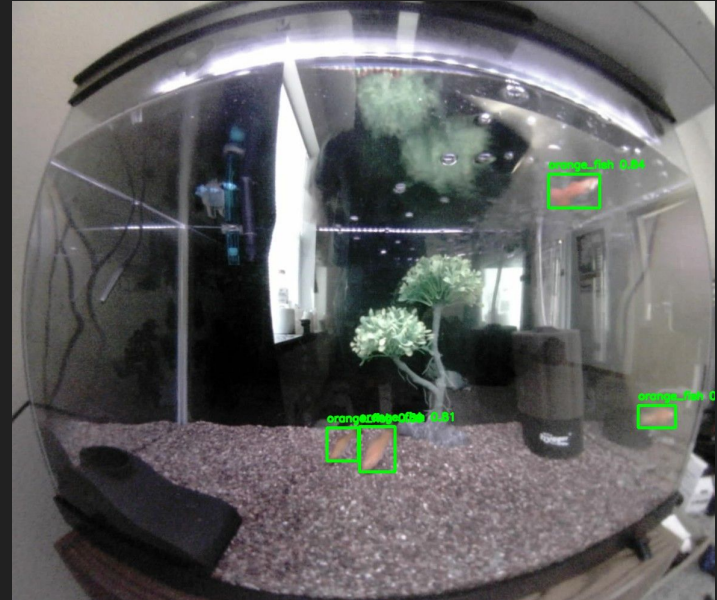
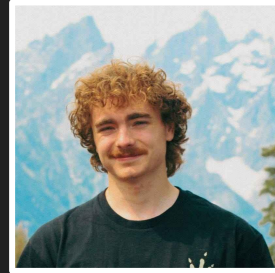


SDMAY26-21 Senior Design

A Smart Aquatic Ecosystem



Meet the team



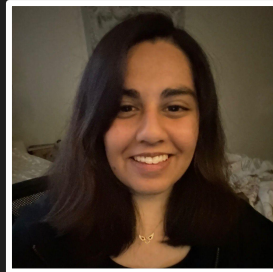
Griffin Urban
Frontend Developer



Aden Koziol
AI Model Programming



Carson Irving
Full Stack Developer



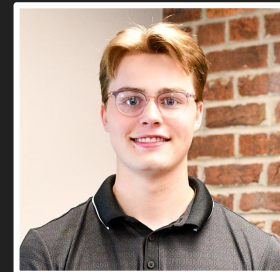
Marina Mannan
Embedded Systems
Engineer



Daniel Pender
Hardware Specialist



Ryan Holden
Backend/Frontend
Developer



Max Purvis
Hardware Specialist

Project Overview

The Smart Aquatic Ecosystem is an AI assisted smart aquarium built to help pet stores track and care for their fish:

- Constant Water Quality Monitoring
- Automatic Feeding
- AI Computer Vision Model Fish Detection

Project Requirements

1. Water Quality Monitoring
2. AI-Powered Fish Detection
3. Temperature Monitoring
4. Automated Feeding
5. Web-Based Dashboard
6. User Alerts

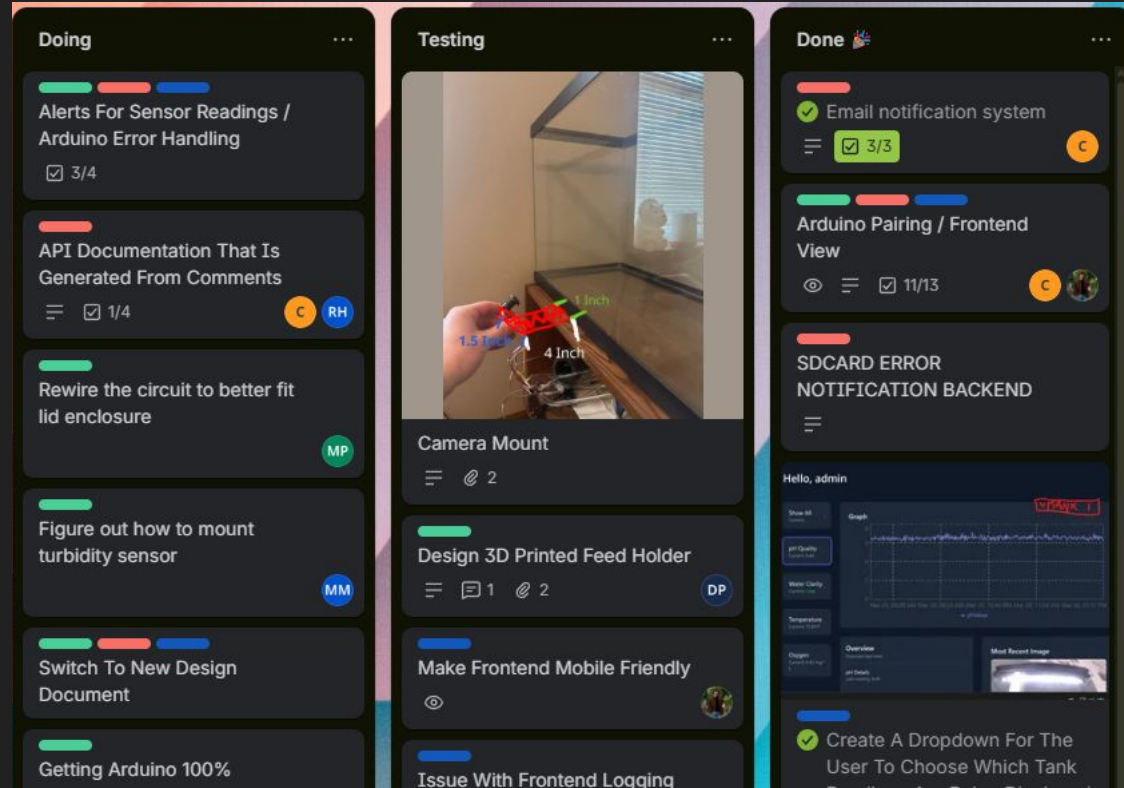
Stakeholders

- Commercial Facilities
- Hobbyists
- The Aquatic Ecosystem

Collaboration

Design Collaboration Process

- Identify need/problem
- Assign ownership
- Create timeline
- Solve problem
- Create tests



Design Method - Agile

Concept Development

- Identify stakeholder needs/desires
- System requirements

Design

- Meets requirements
- Works with other subsystems

Implementation

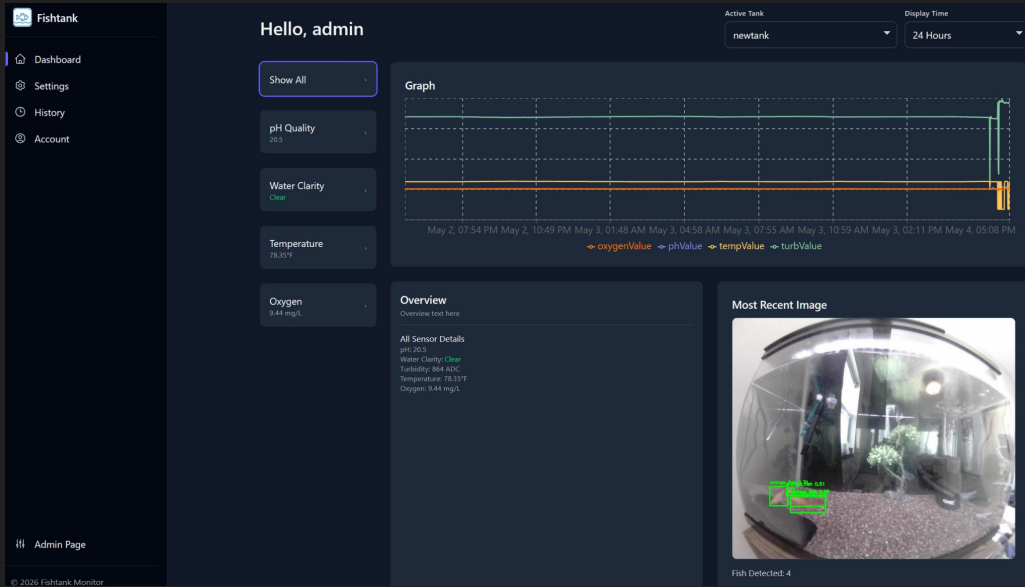
- Develop System

Verify

- Unit testing
- Integration testing

Design

Design - Overview



Design - Physical Components

Fish of choice: Mollys

- Resilience
- Ease of care

Active components:

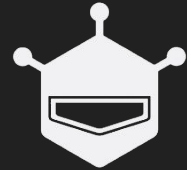
- Water heater
- Water filter
- Air pump



Design - Hardware

Microcontroller (Arduino UNO R4 WiFi)

- Team Familiarity
- WiFi Required For Project
- Powerful Enough To Meet Project Requirements
- Easy Prototyping
- Extensive Documentation and Community Support



DFROBOT
DRIVE THE FUTURE
Spring 2026

Design - Hardware

Important Factors of Fish Health

- Water Clarity
- pH Levels
- Temperature Stability
- Dissolved Oxygen Levels

Sensors:

- Turbidity
- pH
- Temperature
- Dissolved Oxygen



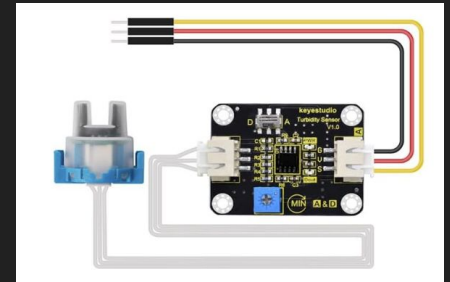
Dissolved Oxygen



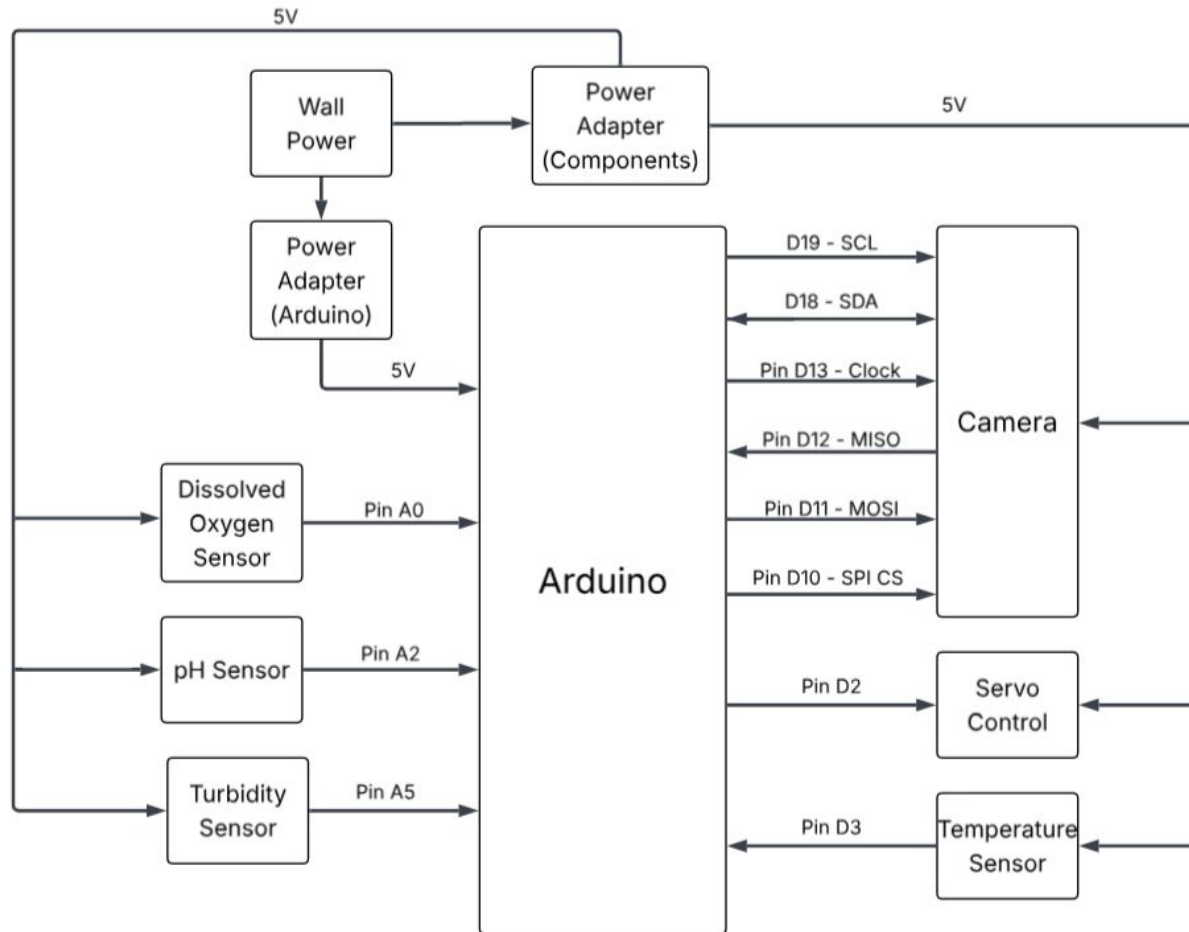
pH



Temperature

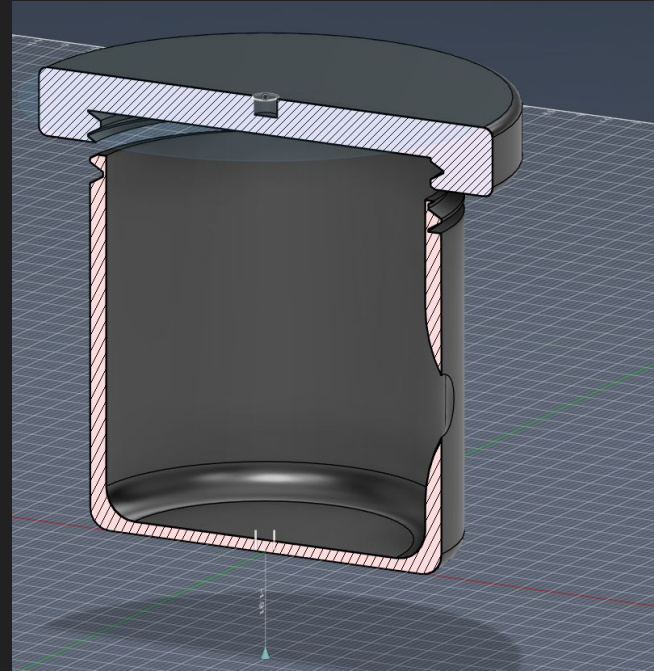


Turbidity



Design - Feeder

- Consistent
- Adjustable
- Refillable
- Stable Long term operation



Design - Frontend

React - dashboard

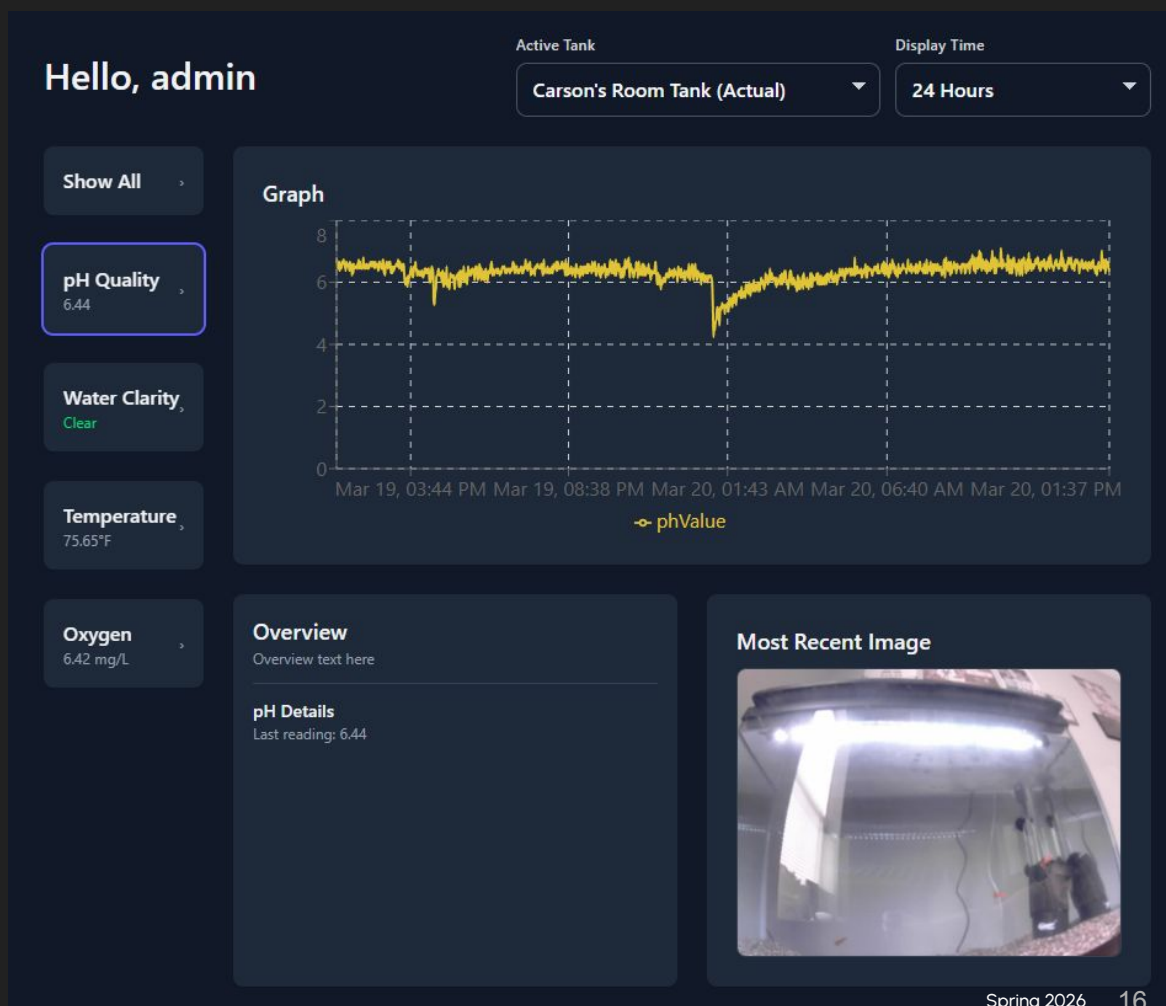
- component based design

Tailwind CSS - design

- lightweight

Recharts - dashboard

- graphs
- Highly customizable



Welcome back

Sign in to view your fishtank dashboard

Username

admin

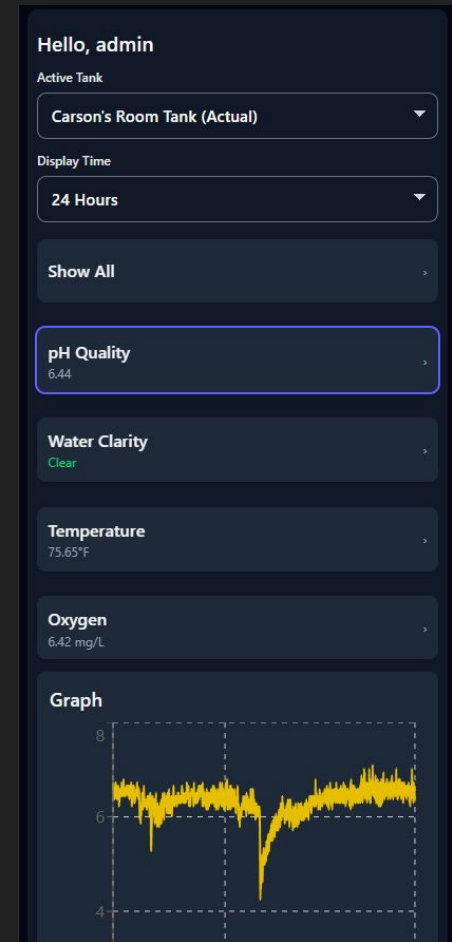
Password

Sign In

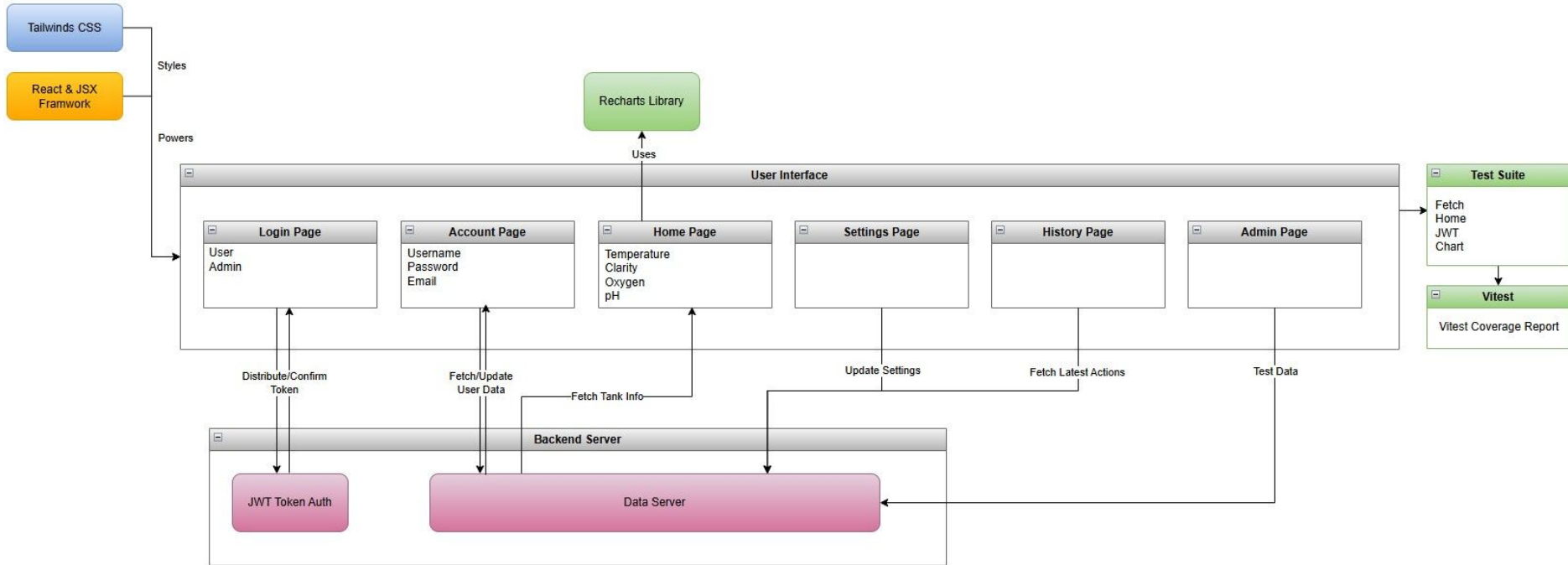
Don't have an account? [Sign Up](#)

Design - Frontend

- Web and mobile friendly
- Tank sharing with other users
- Accessibility features for line colors
- Multiple tank support
- Manual feeding scheduling
- Alerts for sensor values

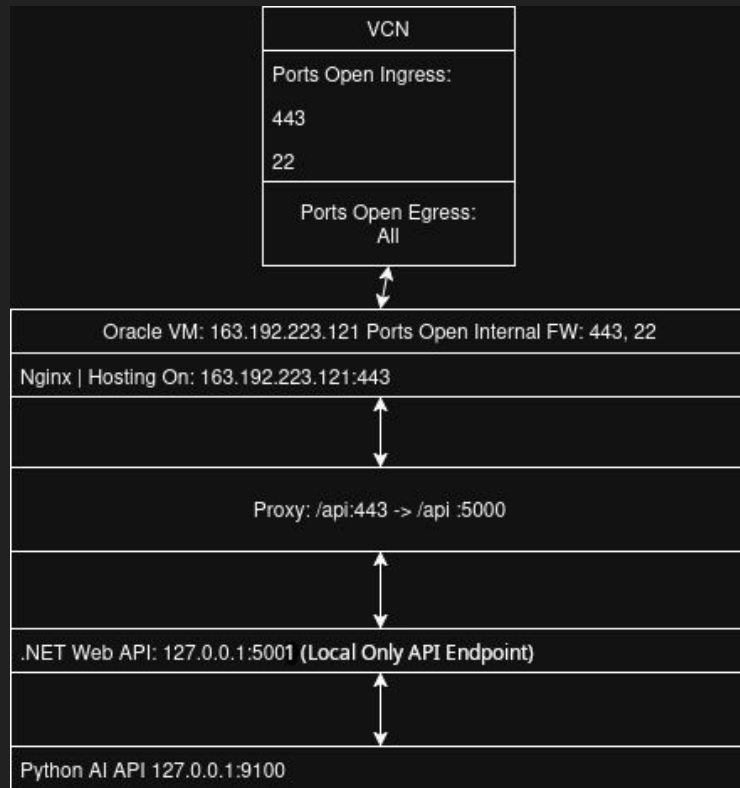


Design - Frontend System Diagram



Design - Backend High Level Overview

- Oracle Free Hosting
- Nginx Web Server
- .NET 9 Web API
- SQLITE Database
- Uvicorn Local Python API



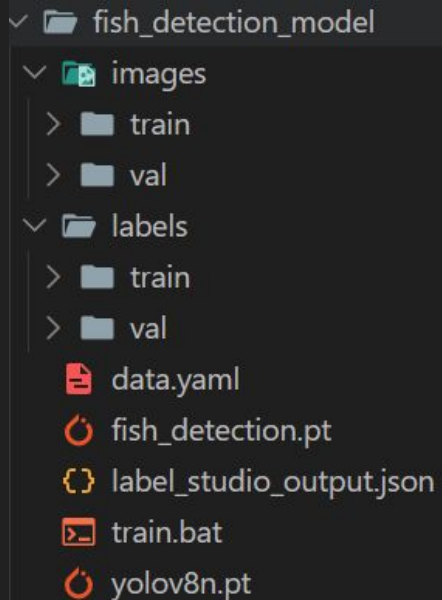
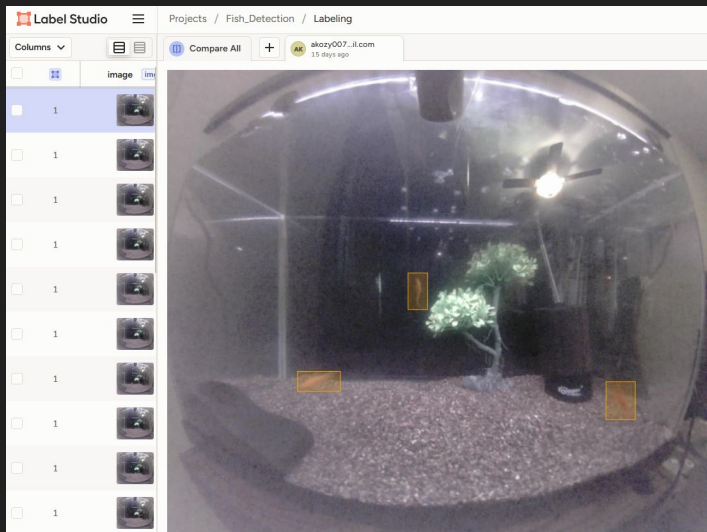
Design - .NET Backend Information

- MVC Design
- EF Core to bridge sqlite
- JWT Authentication for secure endpoints (Tank and User Tokens)

Design - AI Detection

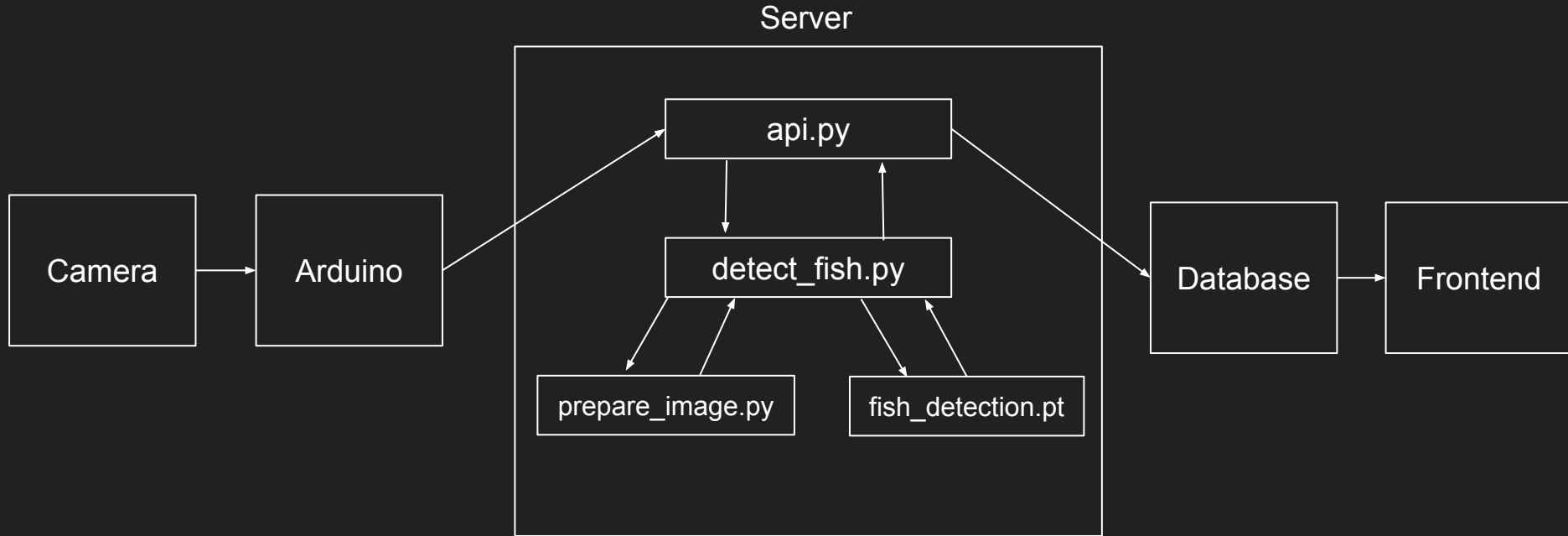


Label Studio
by HumanSignal



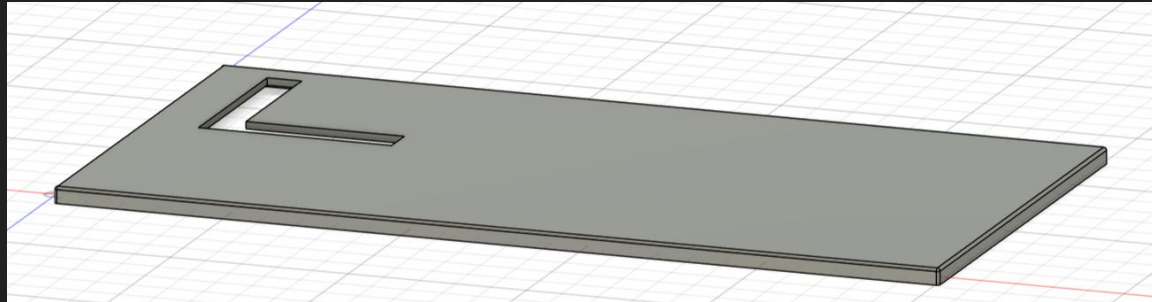
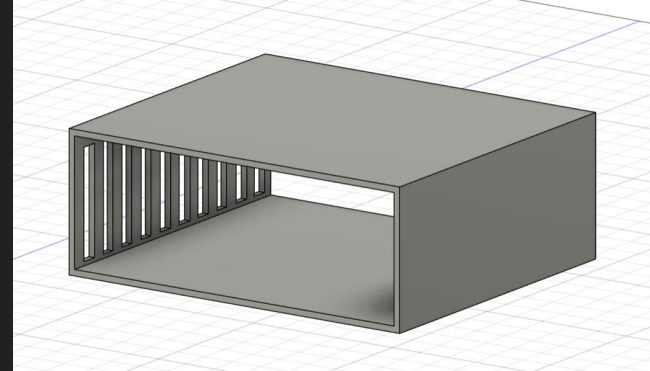
Epoch 1/60	GPU_mem 0G	box_loss 2.021	cls_loss 4.209	dfl_loss 1.43	Instances 17	Size 960: 100% ————— 7/7 5.7s/it 39.9s
	Class all	Images 25	Instances 75	Box(P) 0.0076	R 0.76	mAP50 mAP50-95): 100% ————— 1/1 3.7s/it 3.7s 0.00656 0.00322
Epoch 2/60	GPU_mem 0G	box_loss 1.786	cls_loss 2.821	dfl_loss 1.278	Instances 22	Size 960: 100% ————— 7/7 6.0s/it 42.2s
	Class all	Images 25	Instances 75	Box(P) 0.00893	R 0.893	mAP50 mAP50-95): 100% ————— 1/1 4.1s/it 4.1s 0.243 0.0764
Epoch 3/60	GPU_mem 0G	box_loss 1.766	cls_loss 2.263	dfl_loss 1.268	Instances 11	Size 960: 100% ————— 7/7 6.4s/it 44.8s
	Class all	Images 25	Instances 75	Box(P) 0.0092	R 0.92	mAP50 mAP50-95): 100% ————— 1/1 4.1s/it 4.1s 0.0393 0.0108

Design - AI Detection



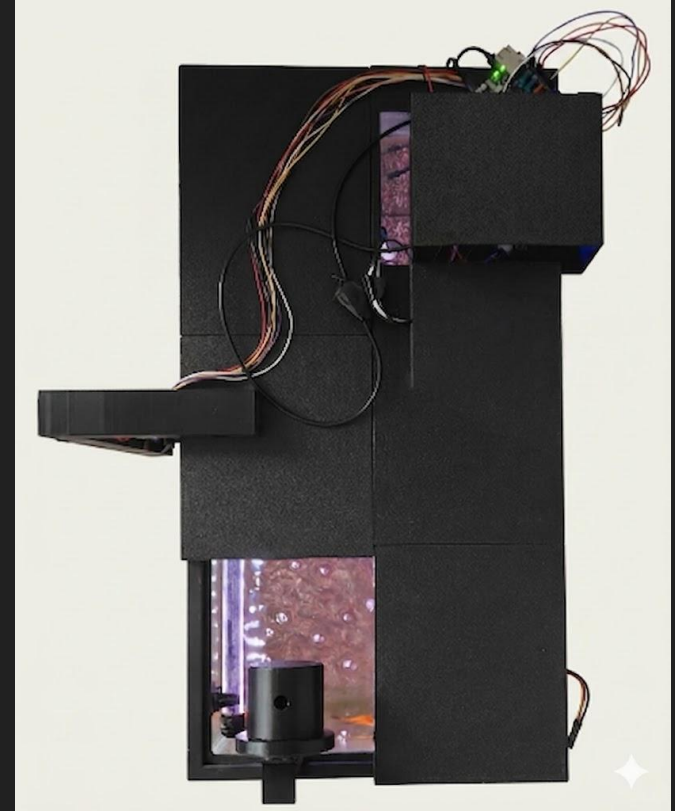
Design - 3D lid application

- Lid designed in CAD with precise dimensions to fit tank geometry
- Cutouts and slots added for component mounting and wire routing
- Adjusted layout for stability and better component spacing



Design - 3D lid application

- Custom 3D-printed lid designed to house electronics and automate tank functions
- Integrated enclosure for Arduino, breadboard, and wiring for a compact setup
- Mounted components include camera and automated feeder mechanism



Testing

Testing - Goals

- Continuous Integration Testing For Frontend, Backend
- Integration Tests Backend Controllers - Mock DB (Ensure Database Integration)
- Test Arduino Over Long Periods with Error Logging
- Test Sensor Accuracy and Longevity
- Test Feeder Precision over Many Cycles
- Accuracy Testing for AI Camera Detection

Testing - Sensors

- Calibration
- Extended Tests
- Storage
- Safety



Testing - Hardware

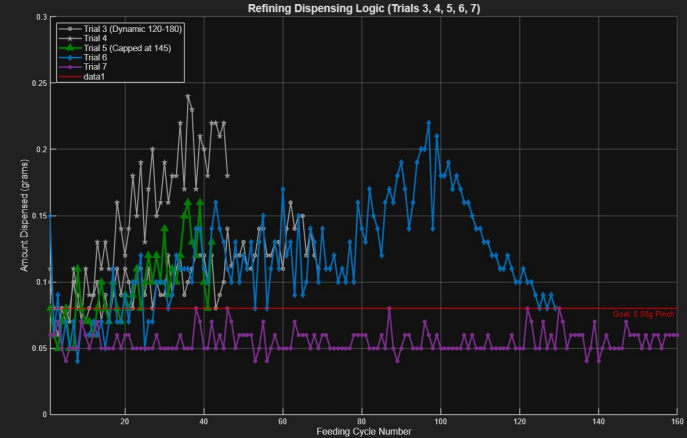
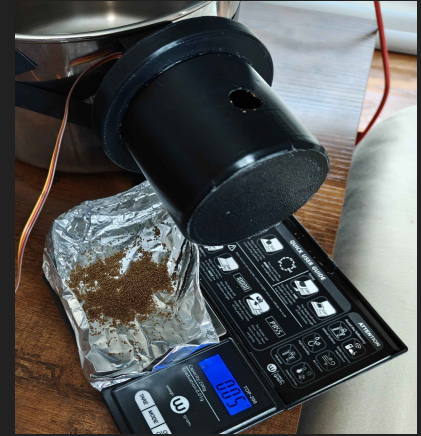
Arduino

```
if (tempDisconnect) Serial.println("TEMP SENSOR DISCONNECTED");  
if (tempError) Serial.println("TEMP SENSOR ERROR");  
if (doError) Serial.println("DO SENSOR ERROR");  
if (phError) Serial.println("pH SENSOR ERROR");  
if (turbError) Serial.println("TURBIDITY SENSOR ERROR");
```

DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 717		TurbV: 3.50		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 716		TurbV: 3.50		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 714		TurbV: 3.49		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 712		TurbV: 3.48		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 712		TurbV: 3.48		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 716		TurbV: 3.50		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 710		TurbV: 3.47		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 713		TurbV: 3.48		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 711		TurbV: 3.48		Turb: LOW
DO (mg/L) : 8.83		pH: 7.00		TempC: 21.12		TempF: 70.03		TurbADC: 715		TurbV: 3.49		Turb: LOW

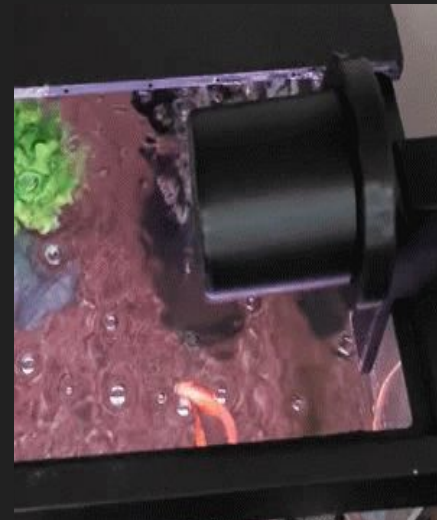
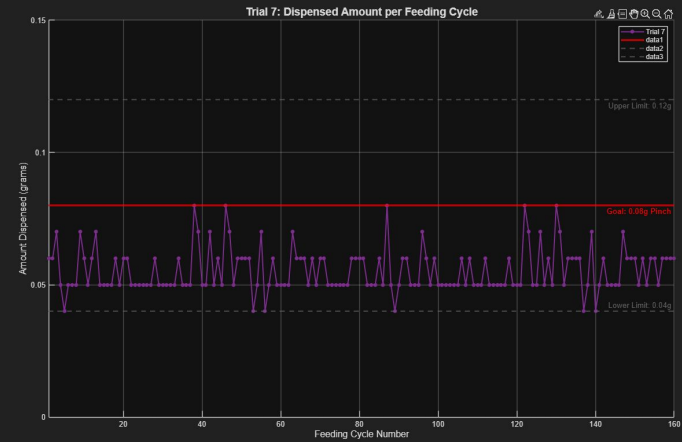
Testing - Feeding Mechanism

- Goal: Consistently drop a pinch of food $\sim 0.08\text{g}$
- Setup: Servo dispensing measured on a high precision scale
- Results: reduced variance, resulting in a stable linear output.

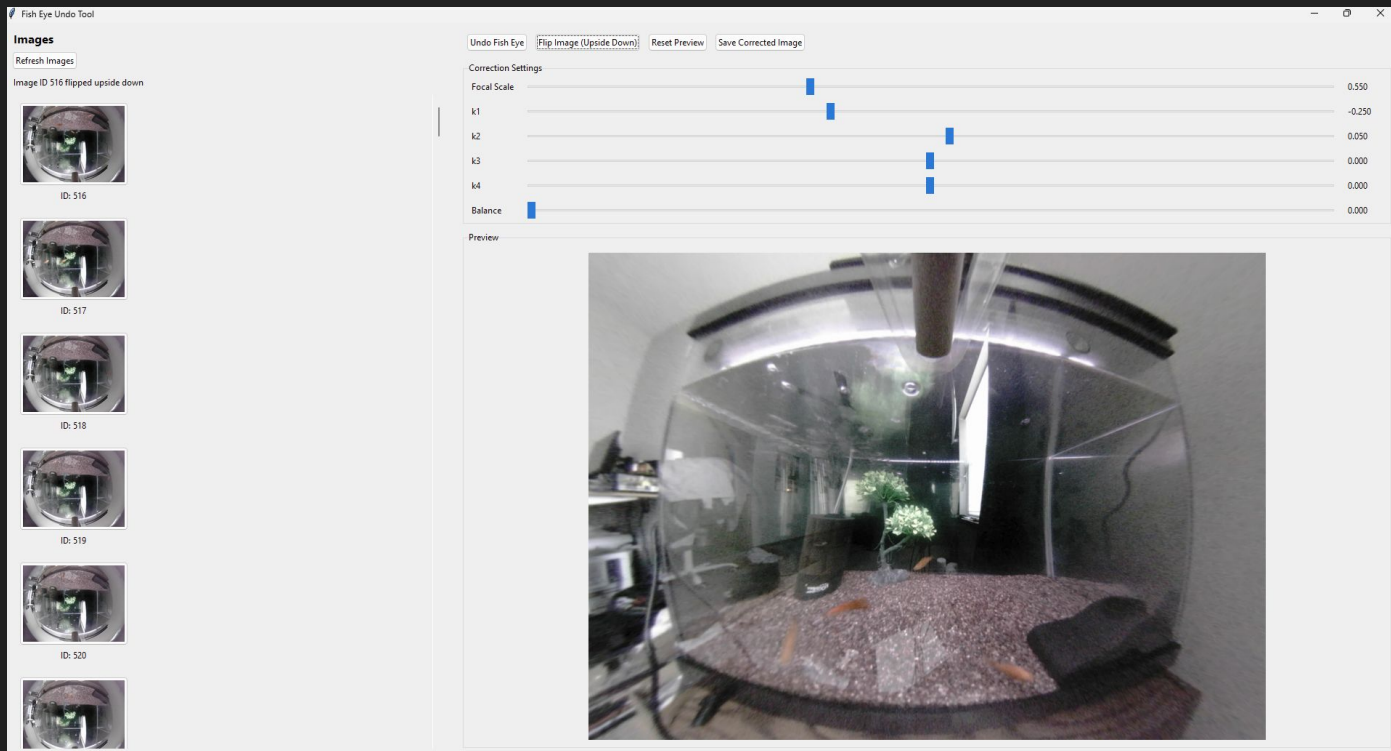


Testing - Feeding Mechanism

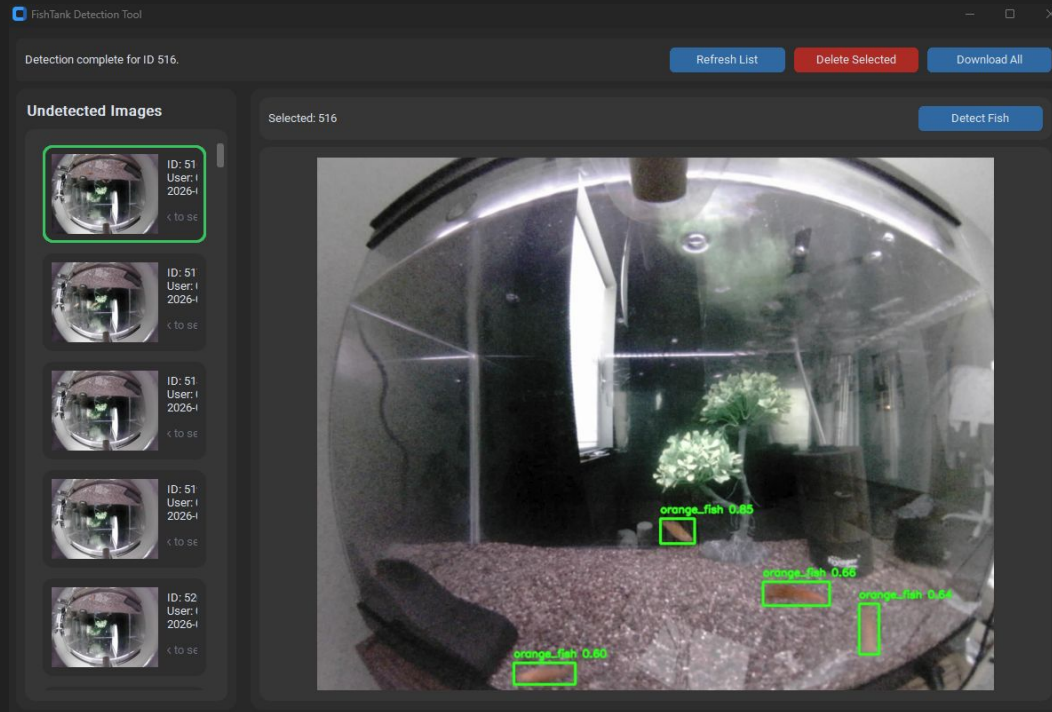
- Consistency: Average 0.055g over 160 cycles
- Safety: Zero output above upper Limit
- Integration: Feeder successfully mounted to the tank



Testing - AI



Testing - AI

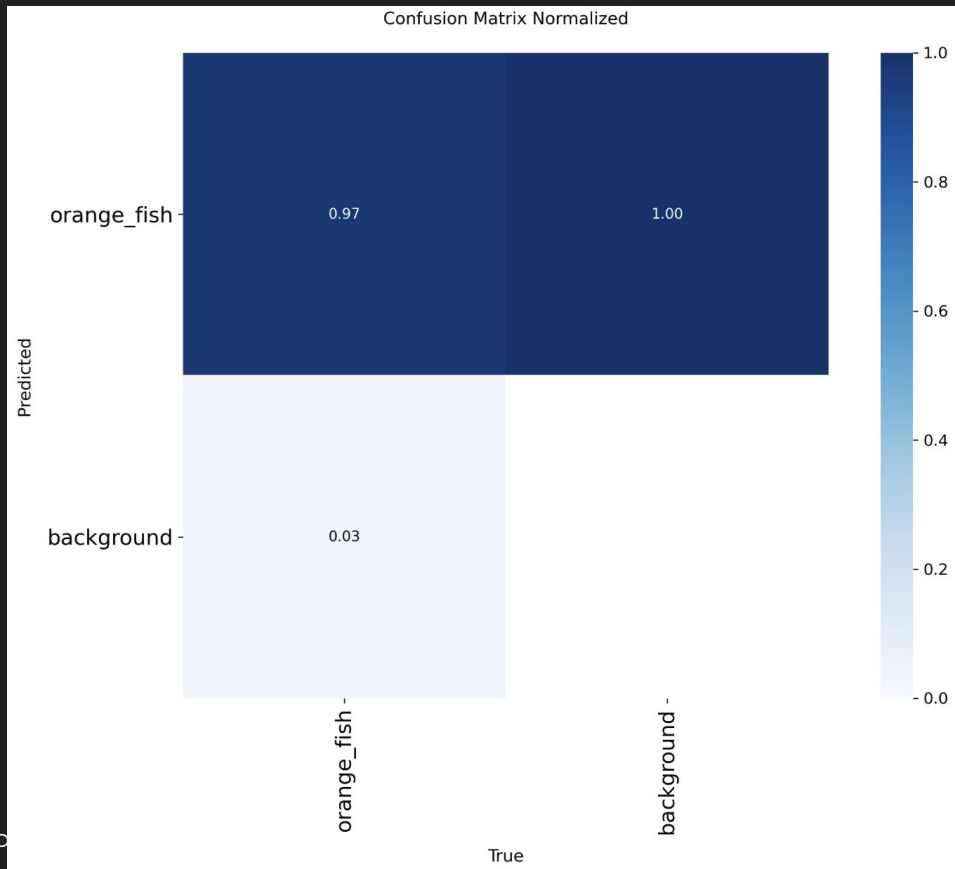


TP

TP + FP + FN

Confidence Levels

Testing - AI



124 Test Images

Precision = $TP / (TP + FP)$

Recall = $TP / (TP + FN)$

$$AP = \sum_{k=1}^m P(k) \times \Delta R(k)$$

mAP50
0.964

Testing - Backend

Framework

- XUnit for Unit and Integration Tests
- Covers controllers, models, and data access

The logo for XUnit.net, featuring a stylized 'X' symbol followed by the text 'Unit.net'.

Test Coverage & Reporting

- Coverlet Collector: Automatically collects code coverage data during tests
- Report Generator: Converts coverage data into human readable format
- Ensures all critical paths are tested
- CI/CD for Continuous Integration Testing To Prevent Regression

The logo for Coverlet, featuring a stylized graphic of overlapping pink and teal lines to the left of the word 'coverlet' in a lowercase sans-serif font.

Testing - Backend

```
45 02:54:06 $ cat TestResults/CoverageReport/Summary.txt
46 02:54:06 Summary
47 02:54:06   Generated on: 5/5/2026 - 2:54:06 AM
48 02:54:06   Coverage date: 5/5/2026 - 2:54:04 AM
49 02:54:06   Parser: Cobertura
50 02:54:06   Assemblies: 1
51 02:54:06   Classes: 34
52 02:54:06   Files: 28
53 02:54:06   Line coverage: 98.4%
54 02:54:06   Covered lines: 1221
55 02:54:06   Uncovered lines: 19
56 02:54:06   Coverable lines: 1240
57 02:54:06   Total lines: 2271
58 02:54:06   Branch coverage: 94.9% (397 of 418)
59 02:54:06   Covered branches: 397
60 02:54:06   Total branches: 418
61 02:54:06   Method coverage: 98.7% (157 of 159)
62 02:54:06   Full method coverage: 89.3% (142 of 159)
63 02:54:06   Covered methods: 157
64 02:54:06   Fully covered methods: 142
65 02:54:06   Total methods: 159
```

Testing - Frontend

React Testing Library

- Great for react components, ensuring content is where it needs to be.

Vitest Library

- Easy and lightweight solution for testing TSX/Vite,
- Mock fetch requests with testing data, and testing utility functions.
- Coverage report generated using Vitest library currently shows coverage above

File	% Stmts	% Branch	% Funcs	% Lines
All files	84.79	79.5	95.45	84.66
src	80.27	74.04	93.63	80.24
App.css	0	0	0	0
home.tsx	83.01	65.94	100	84.21
settings.tsx	79.4	77.66	91.46	79.02
src/components	100	100	100	100
LineChart.tsx	100	100	100	100
src/hooks	98	95	100	98
websocket.tsx	98	95	100	98
src/utils	98.63	93.91	100	98.52
authToken.ts	93.33	92.85	100	93.1
chartColors.ts	100	91.42	100	100
deviceResponse.ts	100	100	100	100
fetch.ts	100	88.46	100	100
jwt.ts	100	100	100	100
testUtils.tsx	100	100	100	100

Testing - Lid and Tank Construction

- Verified the 3D-printed lid's ability to support mounted electronics and mechanical components without deformation
- Confirmed stable performance of the automated fish feeder and camera mount during continuous 24-hour cycles



Key Takeaways

Key Takeaways - Project Progress

Automated Fish Care

- The tank records water quality data and displays it on the user dashboard
- Feeder automatically dispenses food at scheduled intervals

AI Fish Detection

- Exceeded our goal of 95% accuracy

Web-Based Dashboard

- Intuitive, user-friendly design
- User alerts for concerning readings

Key Takeaways - Lessons Learned

Hardware & Integration

- Wiring Complexity: Using individual breakout boards takes up too much space.

AI Fish Detection

- Overfitting: Using too many photos to train the model can result in a worse accuracy.

Key Takeaways - Challenges

Microcontroller

- Sensor Signal Interference
- Keeping Sensors Calibrated
- Managing SPI Interface Across Devices

Camera

- Balancing Lens FOV with Clarity and Fisheye Effects
- No Official Support for Microcontroller

Key Takeaways - Future Prospects

AI Fish Detection

- Upgrade camera setup
- Prevent "hiding" fish

Hardware

- Condense circuitry & Upgrade sensors
- Larger fish tank support

Backend

- Adding a redis cache
- Deleting images after x amount of time

Frontend

- More customizability/modular design

Thank you

Questions?