

A Smart Aquatic Ecosystem

DESIGN DOCUMENT

sdmay26-21

Md Maruf Ahamed (Advisor/Client)

Carson Irving (Frontend/Backend/AI)

Max Purvis (Hardware)

Daniel Pender (Hardware)

Aden Koziol (AI)

Marina Mannan (Embedded Systems)

Griffin Urban (Frontend)

Ryan Holden (Backend)

sdmay26-21@iastate.edu

sdmay26-21.sd.ece.iastate.edu

Revised: Sept. 29th, 2025

Executive Summary

Traditional aquarium maintenance requires frequent manual monitoring of water quality and fish health, which can lead to inconsistent care and increased stress for aquatic life. Our smart aquatic ecosystem aims to use AI and automation to eliminate the need for constant manual adjustment and create a more stable environment for fish or other aquatic life.

The two most important design requirements are automated monitoring and controlling of the environment inside the fish tank. Ideal water conditions can be created by controlling measurable variables of the water quality, including pH, temperature, dissolved oxygen, and turbidity.

Our implementation will use smart sensors to track this data in real time and visualize it on a web-based application for the user's benefit. Automated controls will regulate temperature, feeding, and water exchange/flow. Additionally, the system will track the fish population with a camera using an AI-driven computer vision model. This data will be used along with water quality patterns to model population trends and predict fish health issues. The web app will show records of every adjustment made to the environment, and will notify the user to recommend any necessary action to ensure the continued health of the ecosystem.

Our progress so far has involved researching and ordering the equipment for setting up the fish tank. We ordered the tank itself, pumps, filters, water quality sensors, and an Arduino. We have also researched quality cameras and AI camera detection libraries.

Our next step is to build the tank and begin testing. After we receive the parts, we can begin water treatment to create a stable environment. Then we can buy the fish and begin training the AI model to track population growth.

Put together, this system addresses all the needs of the user by eliminating the need for constant manual care. It monitors water quality and fish health, performs automatic feedings and water replacement, and alerts the user to any urgent changes in the environment.

Learning Summary

Development Standards & Practices Used

List all standard circuits, hardware, and software practices used in this project.
List all the Engineering standards that apply to this project that were considered.

Software

- camelCase and 4-space tabs for programming language
- ASP.NET Core WEB API for backend.
- Descriptive Git commits and work on individual branches before merge approval

Hardware

- Sensor testing and calibration
-

Summary of Requirements

List all requirements as bullet points in brief.

- AI computer vision fish population and growth analysis
- Automatic feeding mechanism using actuators
- Basic Aquarium Equipment (Tank, Filter, Air Pump)
- Advanced Aquarium Sensors (pH, turbidity, water quality, dissolved oxygen, temperature)
- Web-based application to monitor insights, control systems, and real-time visualization of data

Applicable Courses from Iowa State University Curriculum

List all Iowa State University courses whose contents were applicable to your project.

SE

- 2280 - *Introduction to Data Structures*
- 3090 - *Software Development Practices*
- 3170 - *Introduction to Software Testing*
- 3190 - *Construction of User Interfaces*
- 3630 - *Introduction to Database Management*

CS

- 3110 - *Introduction to the Design and Analysis of Algorithms*

CPRE

- 2880 - *Embedded Systems I*

EE

- 1850 - *Introduction to Electrical Engineering and Problem Solving I*
- 2010 - *Electric Circuits*
- 2300 - *Electronic Circuits and Systems*
- 2850 - *Problem-Solving Methods and Tools for Electrical Engineering*

New Skills/Knowledge acquired that were not taught in courses

List all new skills/knowledge that your team acquired which was not part of your Iowa State curriculum in order to complete this project.

Table of Contents

1.	Introduction	5
1.1.	PROBLEM STATEMENT	5
1.2.	INTENDED USERS	5
2.	Requirements, Constraints, And Standards	5
2.1.	REQUIREMENTS & CONSTRAINTS	5
2.2.	ENGINEERING STANDARDS	5
3	Project Plan	6
3.1	Project Management/Tracking Procedures	6
3.2	Task Decomposition	6
3.3	Project Proposed Milestones, Metrics, and Evaluation Criteria	6
3.4	Project Timeline/Schedule	6
3.5	Risks And Risk Management/Mitigation	7
3.6	Personnel Effort Requirements	7
3.7	Other Resource Requirements	7
4	Design	7
4.1	Design Context	7
4.1.1	Broader Context	7
4.1.2	Prior Work/Solutions	8
4.1.3	Technical Complexity	8
4.2	Design Exploration	9
4.2.1	Design Decisions	9
4.2.2	Ideation	9
4.2.3	Decision-Making and Trade-Off	9
4.3	Proposed Design	9
4.3.1	Overview	9
4.3.2	Detailed Design and Visual(s)	9
4.3.3	Functionality	10
4.3.4	Areas of Concern and Development	10
4.4	Technology Considerations	10
4.5	Design Analysis	10

5	Testing	10
5.1	Unit Testing	11
5.2	Interface Testing	11
5.3	Integration Testing	11
5.4	System Testing	11
5.5	Regression Testing	11
5.6	Acceptance Testing	11
5.7	Security Testing (if applicable)	11
5.8	Results	11
6	Implementation	12
7	Professional Responsibility	12
7.1	Areas of Responsibility	12
7.2	Project Specific Professional Responsibility Areas	12
7.3	Most Applicable Professional Responsibility Area	12
8	Closing Material	12
8.1	Discussion	12
8.2	Conclusion	12
8.3	References	13
8.4	Appendices	13
9	Team	13
9.1	TEAM MEMBERS	13
9.2	REQUIRED SKILL SETS FOR YOUR PROJECT	13
	(if feasible – tie them to the requirements)	13
9.3	SKILL SETS COVERED BY THE TEAM	13
	(for each skill, state which team member(s) cover it)	13
9.4	PROJECT MANAGEMENT STYLE ADOPTED BY THE TEAM	13
	Typically Waterfall or Agile for project management.	13
9.5	INITIAL PROJECT MANAGEMENT ROLES	13
9.6	Team Contract	13

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1. Introduction

1.1. PROBLEM STATEMENT

Keeping an aquarium may sound simple, but it's actually a delicate process where small changes in water quality can have big effects on the fish's health. Most aquarium owners stick to a maintenance routine, but what happens when life gets in the way? Our project aims to solve this problem with a smart, AI-powered aquarium management system that monitors water conditions in real time and responds automatically. By using sensors and intelligent controls, the system can regulate temperature, feeding, water flow, and even perform water changes when needed. There will also be artificial intelligence tracking the growth and population of the fish to provide more insights for the owners. This creates a healthier aquatic environment for the fish while reducing the workload for busy owners. The insights gathered will be easily accessible remotely from either a web interface or application, providing peace of mind for the owners who might be away from the tank.

1.2. INTENDED USERS

Who will use the product you create? Who benefits from or will be affected by the results of your project? List as many users or user groups as are relevant to your project (at least three). For each user or user group, (1) describe the user and their key characteristics (e.g., a persona), (2) identify their need(s) related to the project (e.g., a needs statement), and (3) discuss how they might benefit or derive value from the product you create. Justify how these benefits/this value connects to your overarching problem statement.

User Group 1: Casual Aquarium Owners

Description: Individuals or families who have a fish tank at home. They typically have limited knowledge of aquarium care and little time for regular maintenance.

Needs Statement: They need an easy and low-effort way to keep their fish alive and healthy without having to constantly monitor things such as water quality, feeding, and cleaning.

Benefits: The AI tank provides automation for feeding, monitors water quality, and tracks the health of the fish. This makes for a more accessible and less stressful experience for casual fish owners.

User Group 2: Commercial Users

Description: Commercial users, such as pet stores, need to manage several tanks with different species of fish. The staff at these stores may not always be experts, making consistent care across many tanks challenging.

Needs Statement: They need a scalable solution that ensures consistent water quality, reduces labor costs, and keeps fish alive longer.

Benefits: AI automation will reduce manual monitoring and feeding, and maintain standardized care across tanks. This will save costs and increase reliability.

User Group 3: Enthusiasts and Hobbyists

Description: Experienced aquarium owners who might have more expensive fish and setups.

Needs Statement: They need accurate, detailed data and precise environmental controls to optimize their tank conditions, monitor fish health, and experiment with advanced setups.

Benefits: The AI system provides real-time feedback, such as temperature, water quality, and pH. This will allow enthusiasts to achieve higher success rates with their many difference fish, making advanced care more manageable.

Please include any user research documentation, empathy maps, or other artifacts as appendices.

2. Requirements, Constraints, And Standards

2.1. REQUIREMENTS & CONSTRAINTS

List all requirements for your project. Separate your requirements by type, which may include functional requirements (specification), resource requirements, physical requirements, aesthetic requirements, user experiential requirements, economic/market requirements, environmental requirements, UI requirements, and any others relevant to your project. When a requirement is also a quantitative constraint, either separate it into a list of constraints, or annotate at the end of requirement as “(**constraint**).” Ensure your individual requirements are realistic, specific, and reflective or in support of user needs, and ensure your collective requirements are comprehensive.

2.2. ENGINEERING STANDARDS

What Engineering standards are likely to apply to your project? Some standards might be built into your requirements (e.g., many projects uses 802.11 ac wifi standard) and many others might fall out of design.

Q1) Browse the videos available on the IEEE Standards University site (<https://www.standardsuniversity.org/videos/> (Links to an external site.)). Select and watch the following video: <https://www.standardsuniversity.org/video/standards-education-young-professional/> (Links to an external site.), as well as 2 other videos of your own choosing. Note that these videos are generally only 2-3 minutes long.

Briefly describe, in your own words, the importance of engineering standards.

Q2) Browse the IEEE standards website: <http://standards.ieee.org/findstds/index.html> (Links to an external site.). Select a sub-category as appropriate for your project (possibilities include "Computer Technology", "Software and Systems Engineering", and "Communications"). Filter by 'Active Standards'.

Select at least 3 standards that appear, based on their descriptions, to have relevance to your project. Most of these standards can be found on the IEEE Xplore digital library (<http://ieeexplore.ieee.org/Xplore/home.jsp> (Links to an external site.)), which you have access to while using a computer on-campus. If off-campus, you can use the ISU VPN and library portal to get access (<http://instr.iastate.libguides.com/ecpe> (Links to an external site.))

If your project requires *software standards*, please review this source:

<https://www.iso.org/standards.html>

Review each of the selected standards. These can be quite lengthy documents, and you are not expected to read through them in their entirety.

Describe, entirely in your own words, what each standard is about and what it is intended to accomplish.

Q3) After reviewing some of the technical details of the three published standards, do you believe it to have relevance to your project? Why or why not? Be specific.

Q4) Review with your team the standards that each of you have selected. What other standards did some of your team members choose that are different?

Q5) What modifications do you intend to make to your project design to incorporate these standards?

3 Project Plan

3.1 PROJECT MANAGEMENT/TRACKING PROCEDURES

We are using agile for our project management style. While we have a general idea right now of the project goals, there is a lot of room for those goals to change. It would be a mistake for us to lock ourselves into a design when we aren't 100% sure of all of the possibilities with the project. Being able to quickly pivot to add a new feature or change our focus is required in our project, and agile is the best project management style to support these rapid changes.

We are tracking progress and assigning roles in Plaky, which is an issue-tracking board like Trello. The entire team can see what each person is currently working on and get a general idea of their progress. If more detailed information is needed, we can also look at git commits in GitLab to see exactly what changes each person is making. We chose to do this because not all aspects of our project can be tracked through git commits, as there are some non-programming problems that we will have to solve. These include automated fish feeding, water cleaning, and water temperature control. While there will most likely be some code associated with solving these problems, there is more work that will need to be accomplished that cannot be tracked through git.

3.2 TASK DECOMPOSITION

With the help of Plaky mentioned above, we are able to split the project into smaller tasks and subtasks. Currently, our team has organized a file-tree-like structure to split our tasks:

1. Hardware
 - a. Arduino HTTP requests
 - b. Sensor setup
 - c. Feeding system design

- d. Setup camera
 - e. Lid construction
- 2. Software
 - a. Frontend
 - i. landing page
 - ii. HTTP receive
 - b. Backend
 - i. Receiving HTTP
 - ii. Database integration
 - iii. Server hosting
 - c. AI
 - i. AI population Tracking
 - ii. Utilize a camera with an AI model

Each of these tasks may be broken down further into smaller subtasks in order to solve individual problems. The linear timeline of completion of tasks is described in section 3.4

3.3 PROJECT PROPOSED MILESTONES, METRICS, AND EVALUATION CRITERIA

To organize our progress, we have divided our tasks into three main categories: Hardware, Software, and AI. Each task within these categories has a certain milestone and metric/evaluation criteria we use to measure our project's progress.

Hardware

1. Task 1: Arduino HTTP requests
 - a. Milestone: Arduino successfully sends GET/POST requests to the server
 - b. Evaluation Criteria: > 95% requests received successfully
2. Task 2: Sensor setup
 - a. Milestone: Temperature, turbidity, pH, and oxygen sensors calibrated and stable
 - b. Evaluation Criteria: Sensor readings within +/- 5% of reference values
3. Task 3: Feeding system design
 - a. Milestone: Servo motor dispenses food reliably on schedule
 - b. Evaluation Criteria: < 5% deviation from scheduled feed times over 1 week
4. Task 4: Camera setup
 - a. Milestone: Camera captures clear images at defined intervals
 - b. Evaluation Criteria: > 90% usable image rate over 100 captures
5. Task 5: Lid construction
 - a. Milestone: Lid securely houses sensors and camera with waterproofing
 - b. Evaluation Criteria: Passes leak and sensor stability tests

Software

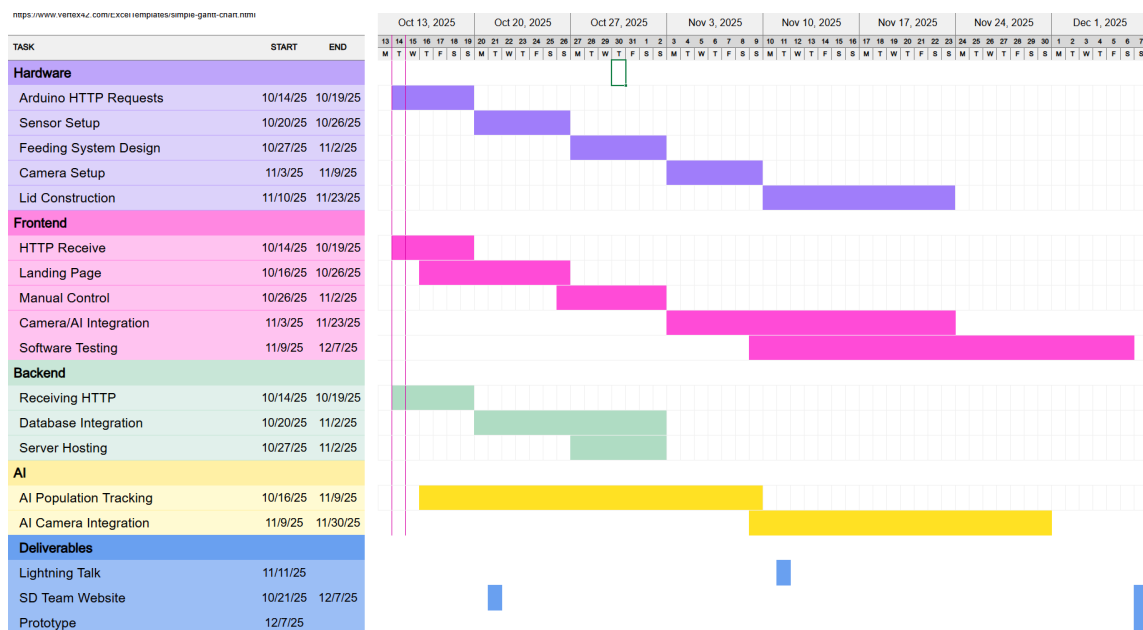
1. Task 1: Frontend landing page
 - a. Milestone: Responsive UI with tank status and manual controls
 - b. Evaluation Criteria: Loads in < 2s, compatible on mobile and desktop
2. Task 2: Frontend HTTP requests received
 - a. Milestone: Frontend receives and displays real-time sensor data
 - b. Evaluation Criteria: > 95% data sync accuracy

3. Task 3: Backend HTTP handling
 - a. Milestone: Server receives and parses Arduino data
 - b. Evaluation Criteria: > 95% parse success rate over 100 transmissions
4. Task 4: Backend database integration
 - a. Milestone: Sensor data stored and retrievable from the database
 - b. Evaluation Criteria: < 1s query latency, > 99% data integrity
5. Task 5: Backend server hosting
 - a. Milestone: Server hosted and accessible via local network
 - b. Evaluation Criteria: > 99% uptime

AI

1. Task 1: AI population tracking
 - a. Milestone: AI model detects and counts fish in tank images
 - b. Evaluation Criteria: > 80% accuracy on annotated test set
2. Task 2: AI camera integration
 - a. Milestone: AI processes live camera feed for fish tracking
 - b. Evaluation Criteria: < 2s latency per frame, > 80% frame classification rate

3.4 PROJECT TIMELINE/SCHEDULE



Prototyping and testing will be ongoing throughout the project. Our project management style is agile, so this schedule will be updated as it is necessary to address the requirements of our project.

3.5 RISKS AND RISK MANAGEMENT/MITIGATION

For each task, identify all salient risks (certain performance target may not be met; certain tool may not work as expected) and assign an educated guess of probability for that risk. For any risk factor with a probability exceeding 0.5 and each high-severity risk, develop a risk mitigation plan. Can you eliminate that task and add another task or set of tasks that might cost more? Can you buy

something off-the-shelf from the market to achieve that functionality? Can you try an alternative tool, technology, algorithm, or board?

An agile project can associate risks and risk mitigation with each sprint.

1. Arduino HTTP requests
 - a. Risks
 - i. Poor signal strength could cause the backend requests to the Arduino to fail (P = .6)
 - ii. Arduino may have limited memory, and the pictures taken by the camera might be too large (P = .6)
 - b. Mitigation Plan
 - i. Use retries so that it will try multiple times and make sure the WiFi connectivity is strong.
 - ii. We could switch out the Arduino for a more expensive Raspberry Pi that has more memory.
2. Sensor setup
 - a. Risks
 - i. Sensors may need calibration after a certain period of time (P = .6)
 - ii. Electrical noise could cause readings to not be accurate (P = .6)
 - b. Mitigation Plan
 - i. Implement a warning email that is sent to the user once every certain period of time to recalibrate sensors
 - ii. We could replace the cheap sensors that we bought with higher-grade sensors
3. Feeding system design
 - a. Risks
 - i. The servo motor may misalign (P = .6)
 - ii. Food could run out without the user noticing. (P = .8)
 - b. Mitigation Plan
 - i. Upgrade the servo motor to a higher-grade servo
 - ii. Add some sort of weight sensor, or calculate how many times the fish can be fed with the container of food. Notify user when out
4. Setup camera
 - a. Risks
 - i. The camera could capture blurry or dark images (P = .7)
 - ii. The camera may not be the most compatible with Arduino (P = .2)
 - b. Mitigation Plan
 - i. Test with multiple lighting setups
5. Lid construction
 - a. Risks
 - i. Failure to waterproof could lead to electrical damage (P = .3)
 - ii. Parts and wires could come loose (P = .6)
 - b. Mitigation Plan
 - i. Make 3D printed mounts to secure equipment properly
6. Landing page

- a. Risks
 - i. UI may not look correct on mobile (P = .9)
 - ii. Load time might be longer than 2 seconds (P = .6)
 - b. Mitigation Plan
 - i. Use a responsive framework like React
 - ii. Optimize scripts or lessen picture quality
- 7. HTTP receive
 - a. Risks
 - i. Websocket implementation errors (P = .2)
- 8. Receiving HTTP
 - a. Risks
 - i. Overload if too many requests are being made (P = .6)
 - b. Mitigation Plan
 - i. Grab data from a database instead of sending a request to the Arduino. Arduino updates the database every x seconds.
- 9. Database integration
 - a. Risks
 - i. Slow queries as data piles up (P = .6)
 - b. Mitigation Plan
 - i. Use a lighter-weight database like SQLite
- 10. Server hosting
 - a. Risks
 - i. Server overheating (P = .1)
- 11. AI population Tracking
 - a. Risks
 - i. The model does not perform well due to a small dataset (P = .7)
 - ii. False positives like reflections or bubbles (P = .7)
 - b. Mitigation Plan
 - i. Use object detection libraries that exist and have been extensively trained
 - ii. Use background subtraction and average out the fish count over multiple images
- 12. Utilize a camera with an AI model
 - a. Risks
 - i. Latency > 2 seconds because of AI response time (P = .9)
 - b. Mitigation Plan
 - i. Use a machine learning model instead, which is much faster

3.6 PERSONNEL EFFORT REQUIREMENTS

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be the projected effort in total number of person-hours required to perform the task.

Task	Explanation	Estimated Hours
Hardware Design & Assembly	Select and connect sensors (pH, turbidity, temperature, DO), actuators (heaters, pumps, feeders), and power supply circuits	30
AI & Computer Vision Development	Develop and train models for fish counting, growth tracking, and predictive analysis of water quality trends	20
Web Dashboard Development	Design and implement a web-based interface for monitoring and alerts	40
System Integration & Testing	Combine hardware, firmware, and software modules, test communication, reliability, and automation features.	40
Calibration & Optimization	Fine-tune sensors, control thresholds, and AI model accuracy under real conditions.	20

3.7 OTHER RESOURCE REQUIREMENTS

Identify the other resources aside from financial (such as parts and materials) that will be required to complete the project.

Hardware Components:

- Arduino Uno
- Turbidity Sensor
- Temperature Sensor
- Dissolved Oxygen Sensor
- pH Sensor
- Servo Motor
- Feeding Mechanism
- Power Supplies

Software Components:

- Arduino IDE
- .NET Core Web API (Backend)

- React JS/ Tailwind (Frontend)
- SQLite (Database)

Fish Tank Components:

- Tank
- Air Pump
- Filter
- Water Heater

4 Design

4.1 DESIGN CONTEXT

4.1.1 Broader Context

Area	Description	Examples
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., the solution is implemented in their communities)	Increasing/reducing exposure to pollutants and other harmful substances, increasing/reducing safety risks, increasing/reducing job opportunities
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	Development or operation of the solution would violate a profession's code of ethics, and implementation of the solution would require an undesired change in community practices
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	Increasing/decreasing energy usage from nonrenewable sources, increasing/decreasing usage/production of non-recyclable materials
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	Product needs to remain affordable for target users, product creates or diminishes opportunities for economic advancement, and high development cost creates risk for the organization

4.1.2 Prior Work/Solutions

The Dangbei Smart Fish Tank 1 Ultra is a product with a similar idea that has not yet hit the market. It was announced at IFA 2025 in August, but it is currently unavailable for consumer purchase. Dangbei's announcement for the Smart Fish Tank advertises AI-powered feeding and

water quality monitoring [1].

The main distinction that sets our design apart is our usage of an AI camera vision model. Both our system and the Dangbei system automate feeding and water quality monitoring. Only our system uses an AI camera vision model to track the fish population and chart growth trends.

Pros of our design:

- AI-powered feeding schedules
- Automatic water quality monitoring
- An AI camera vision model to monitor fish population and health
- 20-gallon tank, easier for a wider market of consumers to fit in their homes

Pros of Dangbei Smart Fish Tank 1 Ultra:

- Full-Color Spectrum Ecological Light
- Dual-Circulation filtration system

There does not appear to be any other similar products available yet. Older smart aquarium systems offer live camera feeds for the user to view their tank at any time, but none implement AI vision models to monitor the tank automatically [2].

There are other University students in the US looking at similar ideas. An engineering student at Brown University made a post to a discussion forum where he proposed the idea of a “smart camera” to track fish activity and provide insight into caring for the aquarium [3]. However, this person does not appear to be pursuing the idea any further or working on a prototype. Overall, our design would be a unique introduction to the market that offers new features that are not available anywhere else.

References

Internet Newsgroup

- [1] T. Ricker, “AI fish tank,” The Verge, Aug. 29, 2025.

<https://www.theverge.com/news/767931/ai-fish-tank> (accessed Oct. 28, 2025).

Internet Newsgroup

- [2] “Smart Tank Technology is the Future of the Aquarium Hobby,” Rate My Fish

Tank.<https://www.ratemyfishtank.com/blog/smart-tank-technology-is-the-future-of-the-aquarium-hobby> (accessed Oct. 28, 2025).

General Internet Site

- [3] “Ideas about a ‘smart’ (AI-driven) aquarium monitoring camera,” Reef2Reef, Jul. 07, 2025.

<https://www.reef2reef.com/threads/ideas-about-a-smart-ai-driven-aquarium-monitoring-c>

amera.1119015/ (accessed Oct. 28, 2025).

4.1.3 Technical Complexity

1. The AI fish tank is multi-faceted and has many subsystems that need to work together in order for it to function. The most complex system within this project is the AI. The AI is responsible for monitoring the fish, their growth, and population. In order to have the AI function as intended, it takes work from both hardware and software. We needed to pair the AI with a server in which the camera, through the Arduino, sends data to the server so the AI can track what we need. In addition to the AI, we have many sensors. We are measuring the temperature, pH, dissolved oxygen, and turbidity of the fish tank. On the hardware side, we have calculated the current draw and orchestrated a way for all of the sensors to feed information to a website. This requires a completely different skill set from the software side. In order for all this to function, we have a server set up in order to store the information gathered.
2. Our system uses computer vision to continuously monitor the fish, which is not commonly used in any available product commercially. Our hardware collects data across multiple parameters. There are many products that offer some data that gets displayed, but few allow for the amount of information our system will be able to display. We also maintain a data pipeline that's fed to a remote server and website for users to view. This allows for more consistent timestamps so users can make more in-depth decisions. Most available systems only offer numbers locally or through a simple app. This design offers automated long-term analysis.

4.2 DESIGN EXPLORATION

4.2.1 Design Decisions

1. AI / Computer Vision Design - We have decided to use a YOLOv8n model, which has already been trained on fish. We chose this because it is lightweight and will be able to be run on a server. We also decided to hook up a camera to our Arduino, which will take pictures for a certain period of time and send them to a database.
2. Sensor & Hardware Design - We chose to use an Arduino for our main hardware component. We chose this because we can easily connect lots of sensors to this and write a code loop that will grab data for a certain period of time. The Arduino has wifi capabilities, so we could easily send it to a server and insert the data into a database.
3. Backend API - We chose to use an API for this project so that we can easily and cleanly grab data from a database and send it to the frontend. The API can be run on the cloud, so it can always receive requests and update data.

4.2.2 Ideation

We looked at different models as well as different ways to run them and do the object detection.

1. YOLOv8n model on cloud (chosen) - Deploy the script onto the cloud so that the object detection is done there. The Arduino just sends the image to the cloud.

2. YOLOv8n model on Arduino - The Arduino isn't that powerful, and constantly using the object detection script, as well as the automatic feeder and sensor reading, would be too much for the Arduino.
3. Smaller YOLOv3 model on Arduino - Even with a smaller model, the Arduino is doing too much work, and the Arduino script wouldn't be as clean, and the whole system wouldn't be very modular.
4. EfficientDet-Lite0 on cloud - Deploy the script on the cloud. Use the EfficientDet-Lite0 model.
5. Train a new machine learning model and deploy on the cloud - Deploy the script on the cloud. Train our own machine learning model on pictures of our own fish tank.

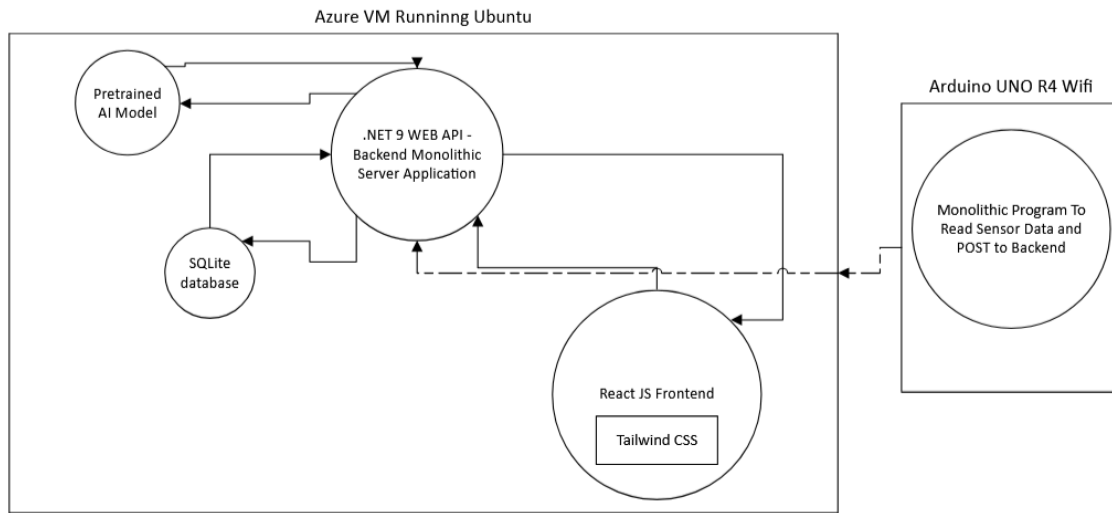
4.2.3 Decision-Making and Trade-Off

1. YOLOv8n model on cloud (chosen)
 - a. Pros: Clean, Best machine learning model
 - b. Cons: Must deploy on the cloud
2. YOLOv8n model on Arduino
 - a. Pros: One less cloud deployment
 - b. Cons: Too much happening on Arduino, Arduino is not very powerful, and less clean Arduino Script
3. YOLOv3 model on Arduino
 - a. Pros: Arduino can run it, one less cloud deployment
 - b. Cons: Less accurate model, too much happening on Arduino
4. EfficientDet-Lite0 on cloud
 - a. Pros: Clean
 - b. Cons: Must be deployed on the cloud, Slower and less accurate model
5. Train a new machine learning model and deploy it on the cloud
 - a. Pros: Clean, works better in theory
 - b. Cons: Spend lots of time training a new model, probably less accurate, must be deployed on the cloud

4.3 PROPOSED DESIGN

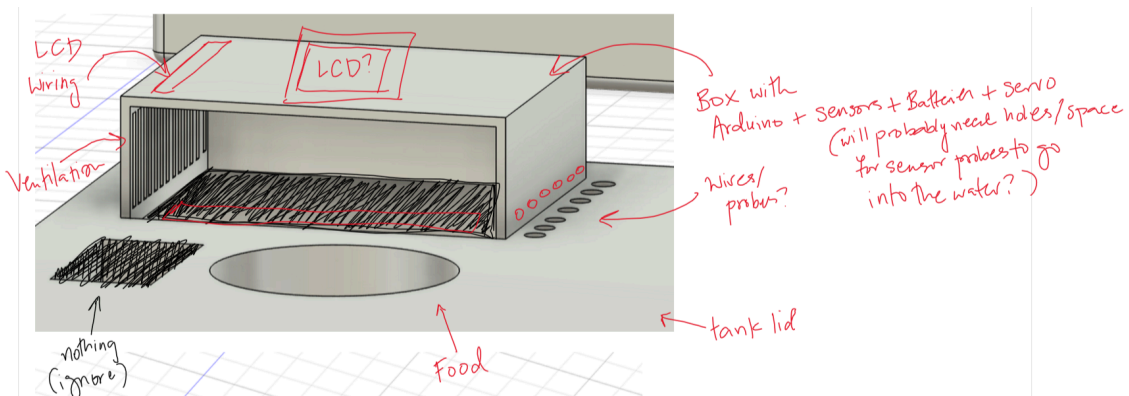
4.3.1 Overview

Technical Design



Currently, we are hosting a .NET 9 WEB API monolithic backend application on a Microsoft Azure VM running Ubuntu. We are using SQLite as our database, as it is a very lightweight solution, and we are using a free, less powerful virtual machine currently. React JS with Tailwind CSS is used for the frontend and is also planned to be hosted on the Azure VM. The pretrained model will ideally also be running on the server. The Arduino is communicating with the virtual machine using HTTP and is posting Images and Sensor Data back to be stored in the database.

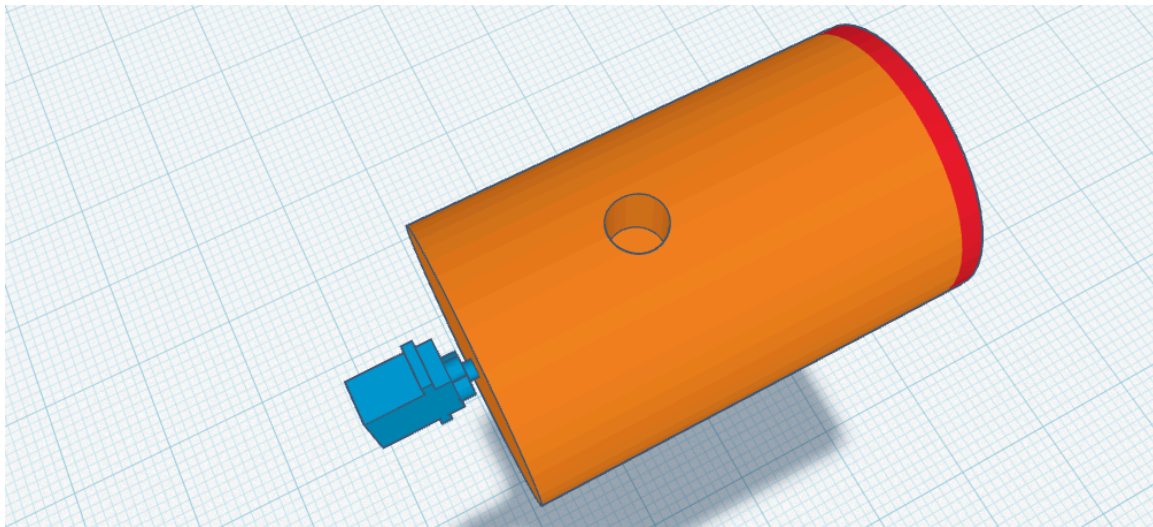
Physical Design



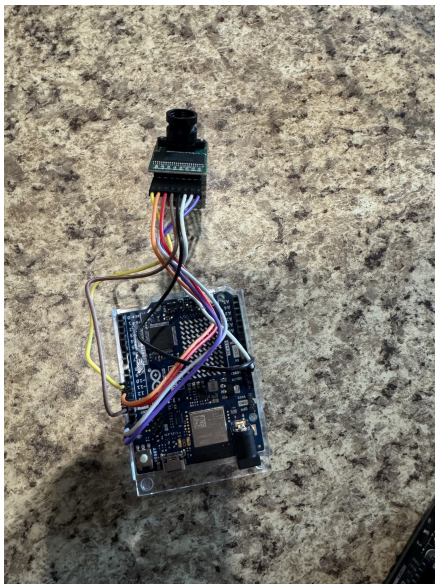
We are planning on designing a lid for the fish tank that will integrate with the Arduino and all of our sensors. We want the design to be practical, so keeping the form factor as close to a normal fish tank as possible is one of our main focuses with the physical design.

4.3.2 Detailed Design and Visual(s)

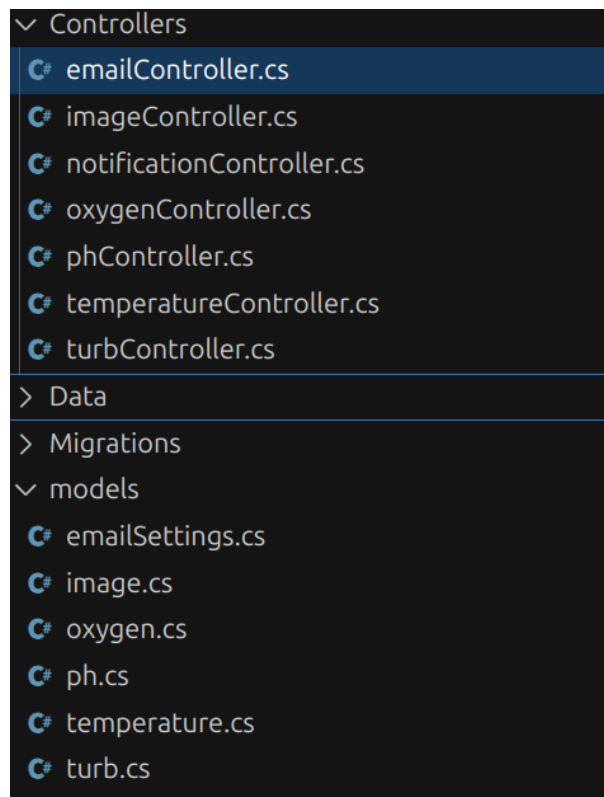
Design For Fish Feeder



The automatic fish feeder is a fairly simple design and will be controlled via a servo motor from the Arduino. In the image above, the blue part is the servo motor that is attached to the end of the cylinder. The orange is the hollow cylinder that holds the fish food, and the red is the cap that you will take off to refill the food. The way the feeder works is by spinning the entire cylinder, which will allow the food to fall for a short period before clogging the smaller hole due to gravity. You will be able to test the functionality to see how much of your chosen food will come out, and then adjust how many rotations you would like to schedule at feeding time.



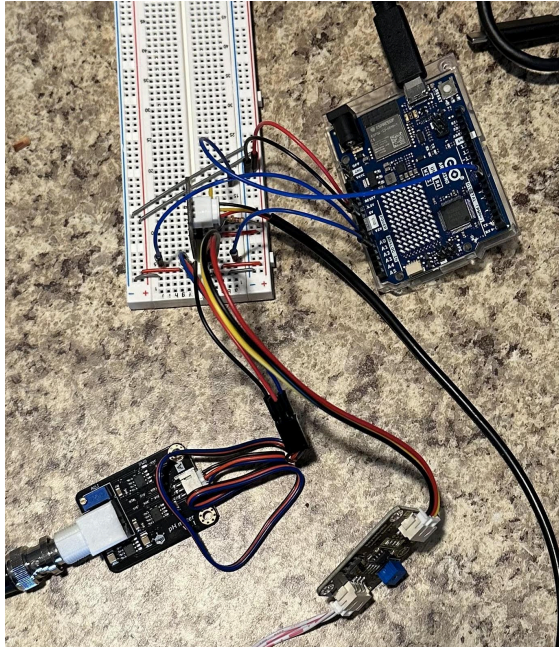
The camera will ideally be placed against the side of the tank, though we are still experimenting with different lenses to get as wide a field of view as possible and still get usable images. The Arduino handles taking the pictures and streaming the data to the backend for storage in the database. The images are stored in base64 strings for easy storage and use with the frontend.



Our backend is organized with models and controllers. Each model, when defined, can have its own table in our database, and if there are any changes made to the model in the future auto-generated migrations can be applied to the SQLite table. For the database and API request handling, I am using all first-party Microsoft packages. For email notifications, I decided to use the MailKit package as it made it very simple to implement.

I am using environment variables to store credentials on the server where needed; these are automatically grabbed by .NET when named properly and used in the server program where needed.

Sensors and Arduino:



All of the sensors will plug into the digital or analog pins on the Arduino UNO R4. Each sensor has a power, ground, and data wire. The pH, dissolved oxygen, and turbidity sensors each use an analog pin, and the temperature sensor uses a digital pin. Each sensor is powered by a 5-volt source. The temperature sensor requires a 4.7 kilo-ohm pull-up resistor on the data pin. Each sensor first breaks out into a separate board, included with the sensor, that contains circuit protection and some of the signal processing elements.

4.3.3 Functionality

Our design is intended to assist aquarium owners by automating routine tank maintenance and providing real-time visibility into the tank's health. The system integrates several sensors (temperature, turbidity, dissolved oxygen, and pH) connected to an Arduino for environmental monitoring. As the aquarium operates in a home setting, the user interacts with it primarily through a web-based dashboard. This dashboard lets the user view current sensor readings, track trends over time, and receive notifications if conditions move outside a recommended range.

In its intended final form, the system will not only measure water conditions but also automatically react to them. For example, if the water temperature falls below a defined safe range, the controller will activate a heater to restore it to normal levels. Similarly, an automated feeding mechanism will dispense food according to a user-defined schedule. The goal is to reduce the user's manual involvement while ensuring stable, healthy conditions for the fish.

A typical user experience might look like this:

1. The user installs the aquarium system and connects it to their home network.
2. The aquarium continuously reads water parameters through its sensors.
3. The user checks the dashboard from a phone or computer to view conditions.

4. If any parameter is abnormal, the system sends an alert and (when applicable) automatically adjusts the environment.
5. The user can override settings or change schedules directly from the dashboard.

This functionality forms the basis of a smart aquarium that both informs the user and takes corrective action to maintain a healthy aquatic environment.

4.3.4 Areas of Concern and Development

The current design meets early requirements for data collection and user visibility, as our sensors are being calibrated and we have established successful communication between the Arduino and the backend. The dashboard displays incoming data, and we are in the process of integrating WebSocket-based notifications. These features begin to satisfy user needs for real-time monitoring and transparency.

However, the system is not yet able to take autonomous corrective actions. Our main challenge moving forward is transitioning from data reporting to full environmental control, while ensuring safety and reliability when actuators such as heaters or pumps operate without direct human intervention.

Our primary concerns moving forward include:

- Completing hardware calibration and validating stable sensor readings.
- Prototyping the automated feeding mechanism and enabling microcontroller-driven actuation.
- Clarifying the level of autonomy expected

Questions for clients, TAs, and faculty advisers include:

- How autonomous is the system expected to be? Should it fully self-correct environmental conditions, or only notify the user to take action?
- Are there recommended safety guidelines or constraints for automated feeding or temperature-regulation systems?

4.4 TECHNOLOGY CONSIDERATIONS

Technology/ Component	Purpose	Strengths	Weaknesses	Trade-off/ Alternative
Arduino UNO R4 WiFi	Central microcontroller for sensor reading, control logic, and communication	-Widely supported, easy to program, large community -Built-in WiFi -Multiple analog and digital I/O	-Limited processing power -Highly sensitive to current.	An alternative would be a Raspberry Pi for heavier data processing, but with higher cost and complexity,

		pins		and only has digital pins.
pH Sensor	Measures water acidity/alkalinity	-Direct, real-time measurement -Analog output is easily read and interpreted by the Arduino	-Requires regular calibration -Fragile	There are digital pH probes with internal compensation for more stable readings, but they would be harder to connect with the Arduino and cost more.
Dissolved Oxygen Sensor	Measures oxygen saturation in water	-Essential for fish health monitoring -Provides real-time readings	-Requires an electrolyte solution in the probe to complete the circuit. -Fragile	There are optical dissolved oxygen sensors that are more accurate but far more expensive
Turbidity Sensor	Monitors water clarity	-Low cost and easy to integrate with Arduino	-Affected by ambient light -Little information available from the manufacturer.	There are optical backscatter probes that are more accurate but costlier and harder to calibrate.
Temperature Sensor	Measures water temperature	-Accurate, digital output reduces noise -Can use multiple sensors on a single bus	-Slower response in rapidly changing temperatures.	Thermistors can offer faster response but require more circuitry and would not be as safe in water.
SG90 Servo Motor	Rotates a container to drop food into the tank	-Simple design -Easy for the user to replace food when needed	-Can't support very much horizontal weight -Draws a lot of current on startup	Stronger servo motors exist to support the weight of the food container; an upgrade to one of these is a solution.
ArduCAM-Mini-5MP-Plus-OV5642 Camera	Takes pictures of the tank	-Compatible with Arduino -Supports still	-The ArduCAM library needs to be updated to	Several alternatives exist, but are less

Module		image capture	support the Arduino R4	compatible with Arduino, if at all.
.NET 9 WEB API	Handles Backend HTTP Requests / Database Management	-Familiarity -Ease of Use -Cross Platform -Open Source -Library Support	-Less performant than something like C or C++	An alternative would be Java, but our backend workers are more familiar with C# and .NET Core
SQLite	A database to store sensor data and images gathered from the Arduino	-Very performant -Great Support in .NET 9 -Easy to set up	-Requires Mitigations to refactor tables -Struggles with high concurrency	An alternative would be MYSQL, but since we do not have much computing power with our VM currently, I think SQLite is the best choice, and has all the features we need. It would need to be reconsidered with massive amounts of traffic.
React JS	JavaScript library for building our user interface	-The component-based design is modular and allows for easy scaling of our interface.	-View rendering can lead to performance issues if the code is not optimized	We could switch to Solid.js if we end up needing better performance, but for now, React fits our needs better
Tailwind CSS	CSS framework for designing the user interface	-Very customizable -Live changes to the user interface make styling faster	-Inline styling can create cluttered HTML files.	Bootstrap is a popular alternative that offers easy design templates, but we want the more detailed customization of Tailwind.

4.5 DESIGN ANALYSIS

Backend:

So far, the backend consists of a .NET 9 Web API application with models and controllers for each required object. This is paired with a SQLite database, and SQLite was chosen for performance reasons. And so far, it has accomplished everything we need a database to do. .NET 9 has also allowed us to accomplish everything we needed an API to do. I, Carson, had prior experience creating .NET applications, so I was confident it would work, and so far it is proving successful. I am currently using XUNIT.NET for unit testing, and it is working great to make sure we do not break anything with future updates. For the future, I hope to have CICD set up with GitLab to be able to automatically test and deploy my application when I push changes. I also foresee the need to make more specialized methods based on what we need to do with each vector of data, but I do not foresee any issues with creating these. Overall, I think our choices for the backend are successful so far, and I do not currently see any major changes that will need to take place.

Hardware/ Electrical:

So far, we have tested the temperature, turbidity, and pH sensors. They all function correctly and as expected, though the pH and turbidity sensors require further calibration. We have also built the fish feeder with the servo motor. It functions as expected, but the weight of the food container is a concern because the servo motor cannot hold up very much weight. We are looking into either reducing the size of the container significantly or getting a stronger servo, both of which would be great solutions. We have set the camera up to capture images, but are still searching for ways to capture as much of the fish tank as possible, such as using different lenses. We have not been able to test the dissolved oxygen sensor yet because it requires the prepping of the probe with an electrolyte solution to complete the circuit, and we do not yet have that. Running this sensor without the electrolyte solution can damage the electrodes within the probe, and the probe requires regular maintenance when in use, so we are waiting to set it up.