

SCHOOL *of* BUSINESS AND TECHNOLOGY

Department of Engineering and Aviation Sciences

**Clam Activity Detection System**

**Shanice Nurse**

**Ashley Afueh**

Advisor: Dr. Zhang

April 23, 2020

Tank Activity Detection

By

Shanice Nurse & Ashley Afueh

Submitted to the Department of Engineering and Aviation Sciences in partial fulfillment of the requirements for the degree of Bachelor of Science in Engineering at the

UNIVERSITY OF MARYLAND EASTERN SHORE

Date

The author hereby grants the University of Maryland Eastern Shore permission to reproduce and distribute publicly, paper and or electronic copies of this document in whole or in parts.

Authors Shanice Nurse, Ashley Afueh

Signature

Date

Department of Engineering and Aviation Sciences

Certified by

**List of Contents**

[**List of Contents 2**](#_30j0zll)

[**List of Figures 4**](#_1fob9te)

[**List of Tables 5**](#_3znysh7)

[**Abstract 6**](#_2et92p0)

[**1.**](#_tyjcwt) **Introduction 7**

[**1.1**](#_3dy6vkm) **Backgound/Motivation 7**

[**1.2**](#_4d34og8) **Objective 7**

[**1.3**](#_2s8eyo1) **Design Requirements 7**

[**1.4**](#_17dp8vu) **Design Constraints 7**

[**1.5**](#_3rdcrjn) **Design Methods 7**

[**2.**](#_26in1rg) **Project Description 9**

[**2.1**](#_lnxbz9) **System Description 9**

[**2.2**](#_35nkun2) **System Diagram 9**

[**2.3**](#_44sinio) **System Functions 9**

[**3.**](#_2jxsxqh) **Implementation Plan 11**

[**3.1**](#_z337ya) **Tasks 11**

[**3.2**](#_3j2qqm3) **Team Organization 12**

[3.2.1.](#_1y810tw) Responsibility of Team Member 1. 12

[3.2.2.](#_4i7ojhp) Responsibility of Team Member 2. 12

[**3.3**](#_2xcytpi) **Timeline/Milestones/Delivery Plan 12**

[**4.**](#_3whwml4) **Implementation 13**

[**4.1**](#_2bn6wsx) **Implementation of Task 1. 13**

[4.1.1.](#_qsh70q) Implementation of Subtask 1.1 13

[**4.2**](#_3as4poj) **Implementation of Task 1. 13**

[**5.**](#_1pxezwc) **Conclusion (Discussion and Future Plans) 14**

[**Acknowledgment 15**](#_2p2csry)

[**Appendix 16**](#_147n2zr)

[**A.**](#_3o7alnk) **Component Specs 16**

[1.](#_23ckvvd) Specs of Arduino Due 16

[2.](#_ihv636) Specs of Raspberry Pi 16

[**B.**](#_32hioqz) **Source Code. 16**

[1.](#_1hmsyys) Source Code of Graphic User Interface 16

[2.](#_41mghml) Source Code of Robotic Arm 16

[**REFERENCES 17**](#_2grqrue)

**List of Figures**

[Figure. 1.](#_1t3h5sf) Mussel Active Filtering (A), Voltage Output (B), and Mussel Resting (C). 6

[Figure. 2.](#_1ksv4uv) Design Methods. 7

**List of Tables**

[Table 1.](#_1ci93xb) Project Timeline and Delivery Plan 10

**Abstract**

By the end of the project, summarize the project into short text and put here (can be waived for Senior Design I). The abstract needs to be more than half page.

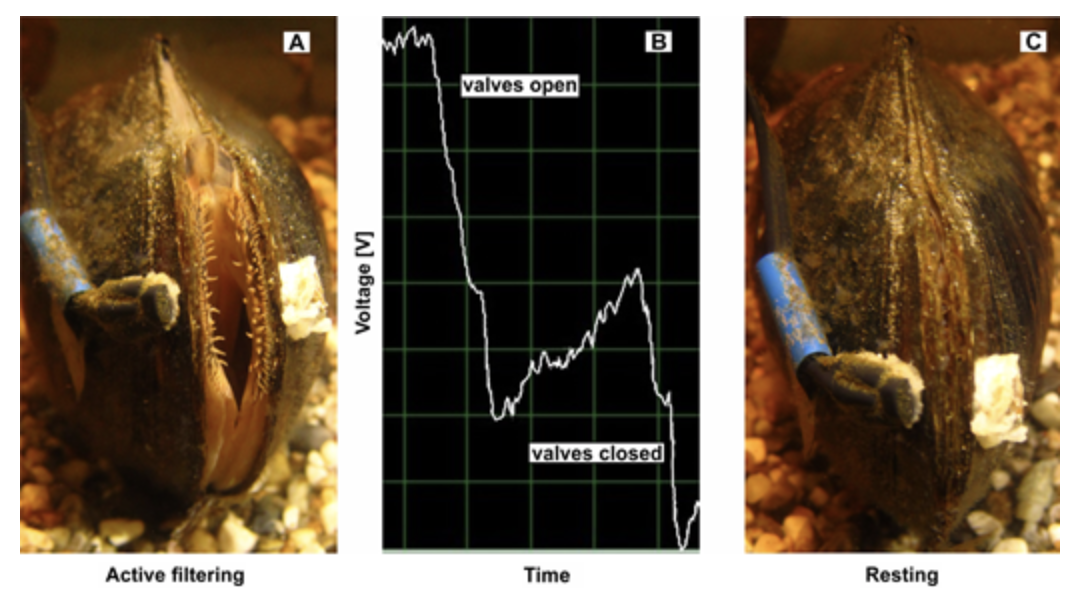
1. **Introduction**

## 1.1 Background/Motivation

The capability of remotely monitoring the behavior of animals and their relationship with their habitat has restructured how ecologists conduct research. Freshwater mussels (Unionoida) are target species in aquatic conservation and belong to one of the fastest diminishing taxa globally (Bogan, 1993; Geist, 2010, 2015). Most recently, there has been a critical effort to put resources into improving the conception of the reasons for the decrease of Unionoida. Biologists have urged the discovery of a solution to observe the behavior, physiology, and ecological conditions experienced by creatures as they progress and communicate with their habitat. On account of the mechanical advances made in the course of the most recent 40 years, a wide range of approaches have been created to precisely comprehend the lives of creatures that go overlooked by the observation of an ecologist. Comprehensively, the strategies used to monitor clams are directly related to the "biotelemetry", which can be characterized as the remote detection of physiological, or behavioral activity information. It is effective to include biotelemetry into the study of clams in order to have a precise understanding of its preservation to the ecosystem. The standard amount of biotelemetry incorporated in research, differs enormously, from the utilization of data recorded, to transmitters that send data to either land-based receivers, or to a satellite circumnavigating the earth. For this reason, researchers and different articles suggest it useful to attach sensors to the mussels’ shell for ecological investigations and preservation ventures.

Various resesearch projects range from, but are not limited to, mark-recapture studies to translocation and restocking programs, and mussel behavior research (Smith, Villella & Lemarié, 2003; Villella, Smith & Lemarié, 2004; Kurth *et al.*, 2007; Wilson *et al.*, 2011; Hartmann *et al.*, 2015). For instance, the controlled release of captive-reared freshwater pearl mussels (*Margaritifera margaritifera*) requires effective tools to monitor the success of such conservation projects (Gum, Lange & Geist, 2011). Adjoining sensors to a mussel may bring about strain for the mussel and increase endangerment to their vulnerability to harmful mixes from the glue. To connect an item includes the expulsion of the mussel from its condition, cleaning and drying the shell, applying the paste, squeezing the remote article against the shell and, for some paste', the procedure additionally includes keeping the mussel out of the water until the paste is dry. Currently, there are experimental studies of filtration-behavior and biological rhythm for freshwater mussels. Biologists perform their research by attaching objects to bivalve shells for conservation projects, mark-recapture studies, and behavioral analysis. The “marking” process is performed by using a magnet and a rubber-coated Hall sensor glued with cyanoacrylate adhesive attached to a mussel shell known as the “Anodontia Anatina.” These attachments were built to last for nine months after attachment. This system was designed for a filtration-behavior experiment, which identified a circadian rhythm. This experiment was performed by placing twenty-six mussels that were randomly selected were placed into two different closed recirculating aquarium systems. After allowing the mussels to adapt to their laboratory habitat for about 12 days, a Honeywell SS495A linear position Hall sensor (Honeywell, USA) and magnet were then attached for monitoring behavior. Additionally, this system allowed ecologists to observe the water quality parameters, dissolved oxygen (DO), pH and EC daily. Figure 1 depicts the measurement of a mussel’s filtration behavior using the change in proximity of a magnet (in right valve) to a Hall sensor (on left valve). In Figure 1, photograph B depicts a graph showing a measurement of the voltage output transduced by the Hall sensor when the mussel is active and resting.

Figure 1.



1. Mussel Active Filtering (A), Voltage Output(B), Mussel Resting(C).

This project is intended to assist in research for the Food & Science Department at the University of Maryland Eastern Shore. By developing a Clam Activity Detection System, a dataset from the activity of the clams will be derived from the activity observed without putting addition strain on the clams. This would be extremely beneficial to ecologist and researchers as they can remotely monitor precise data from their experiments. The dataset will be the basis for the development, evaluation, and use of the neural network developed. During this project, there will be models developed for photograph classification, and object detection to further investigate the activity of clams. By creating this development, the department will have a better insight of the clams’ behavioral activity with its environment. The Deep learning Convolutional Neural Network will be trained with an image classification model to observe each clam individually. This neural network is intended to monitor the clams’ amount of interaction, length of interaction, growth, and its overall adaptivity to its surroundings. This deep learning model will have the ability to integrate the feature extraction and classification process into a whole to record when the clam is most comfortable.

The purpose of this project is to develop a system that focus on the adaptive ability to accurately process images used to observe the activity and interaction of clams without the tedious strain imposed on their shell. By using deep learning approach to monitor the clam’s activity the data will be recorded with increased accuracy. As previously stated, there methods on the market that will accomplish similar tasks as our system. However, the methods do not offer the precision and comfortability our system can guarantee.

## Objective

The objective of this project is to create a system that can precisely monitor and record activity of clams with 95% accuracy.

## Design Requirements

1. The system will individually track each clam.
2. A time-lapse recording log will be implemented for visual data recording.
3. Access to a 24-hour data log will be offered with hourly updates.
4. Controlled parameters of the water quality along with the clam’s interaction to the environment will be accessed through a local network.

## Design Constraints

1. Determining whether the raspberry pi camera will record data most precisely underwater or mounted amongst tank walls.

## Design Method

1. Mussel Active Filtering (A), Voltage Output(B), Mussel Resting(C).

## Standards

List in this section all (industry) standards the project complied with.

1. **Project Description**

## System Description

This system intended to remotely monitor activity is controlled by a Deep Learning Convolutional Neural Network. For the project, the neural network is trained to identify various activities. A dataset will need to be prioritized with photographs of active clams and resting clams provided as a subset of photographs from a larger dataset. Using Keras the photos will be loaded to pre-process into standard directories. Python will then be used to create directories and subdirectories for both the “train” and “test” directories. After testing the dataset, a baseline Convolutional Neural Network will need to be developed. Different convolutional layers within the baseline model will be tested for accuracy. About three different VGG-based architecture models will be implemented to improve the performance by increasing the model capacity. This technique will alter the performance by allowing the model to learn features that will maximize the training dataset. After the process of model improvements, a final model configuration is selected and adopted. A final model normally includes all available data; the combination of train and test datasets. Through a web application, researchers will be able to view recorded datasets.

## System Diagram (or Flow Chart)

A screenshot of a cell phone

Description automatically generated

1. System Flow Chart

## System Functions

Please clearly define every functional module and their relationships (I/O) in the system here. Make sure the state graph is complete. For example:

"

1. Unit one is a XXX device. Its function is XXX and XXX. It receives signal from XXX and transmit XXX to XXX. When the caller press a button, first the message will show on the display of the device with the caller, also the message will be coded together with its ID and send to the callee.
2. …

"

1. **Implementation Plan**

## Tasks

* Task 1. Clam identification image processing code
* Subtask 1.1: Photo dataset preparation
* Subtask 1.2: Develop a baseline CNN (convolutional neural network) model
* Subtask 1.3: Improvement on CNN model
* Subtask 1.4 Develop Transfer learning
* Subtask 1.5: Finalize CNN model

* Task 2: Clam activity level image processing code
* Subtask 2.1: Photo dataset preparation
* Subtask 2.2: Develop a baseline CNN (convolutional neural network) model
* Subtask 2.3: Improvement on CNN model
* Subtask 2.4 Develop Transfer learning
* Subtask 2.5: Finalize CNN model
* Task 3: Network capturing code (sends data from Camera to Pi)
* Subtask 3.1 - Create code to send pics from Camera to raspberry pi
* Task 4: Code to Send Sensor Data to Raspberry Pi
* Subtask 4.1: Create code to send sensor data to raspberry pi
* Task 5: Code for Data Organization in Raspberry Pi
* Subtask 5.1: Sorting data files
* Subtask 5.2: Writing data to a file
* Subtask 5.3: Modeling Data for database
* Subtask 5.4: Work with data constructed
* Subtask 5.5: Ensuring all cameras are synchronizing with little to no error
* Task 6: Send Raspberry Pi data to Cloud
* Subtask 6.1: Create channel for data
* Subtask 6.2: Create API key
* Subtask 6.3: Create python code for data
* Task 7: Create Website Database
* Subtask 7.1: Organize layout for website
* Subtask 7.2: Sort data for general clam activity and activity level
* Subtask 7.3: Sort data for individual clam activity and activity level
* Subtask 7.4: Create figures for data
* Subtask 7.5: Run website, fix errors
* Task 8: Create App Database
* Subtask 8.1: Organize layout for website
* Subtask 8.2: Sort data for general clam activity and activity level
* Subtask 8.3: Sort data for individual clam activity and activity level
* Subtask 8.4: Create figures for data
* Subtask 8.5: Run website, fix errors
* Task 9 - Lab Setup
* Subtask 9.1: Mound Cameras to tank
* Subtask 9.2: Mound Sensors to tank
* Subtask 9.3. Setup Raspberry pi for experiment
* Subtask 9.4: Setup any more other configurations
* Task 10: System evaluation
* Subtask 10.1: Determine accuracy levels of 4 cameras and of each sensor.
* Subtask 10.2: Test entire system
* Subtask 10.3: Fix any errors
* Subtask 10.4: Evaluate system
* Subtask 10.5: Refine system (if necessary)

**3.2: Team Organization**

Team Member 1: Shanice Nurse

Team Member 2: Ashley Afueh

***3.2.1: Responsibility of Team Member 1:***

Task 1: Clam identification processing code

Task 3: Network Capturing Code

Task 6: Code to Send Raspberry Pi data to cloud

Task 8: Create App Database

Task 9: Lab Setup (half)

Task 10: System Evaluation (half)

***3.2.2: Responsibility of Team Member 2:***

Task 2: Clam activity level image processing code

Task 4: Code to send sensor data to Raspberry Pi

Task 5: Code for data organization in Raspberry Pi

Task 7: Create Website Database

Task 9: Lab Setup (half)

Task 10: System Evaluation (half)

## Timeline/Milestones/Delivery Plan

1. **Project Timeline and Delivery Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| Time | Task | Comment | Responsible Personnel |
| Week 1 | Begin Subtask 1.1, 1.2 | Task 1: Clam identification processing code | Shanice Nurse |
| Week 2 | Finish Subtask 1.1, 1.2 | Shanice Nurse |
| Week 3 | Begin Subtask 1.3, 1.4 | Shanice Nurse |
| Week 4 | Finish Subtask 1.3, 1.4 | Shanice Nurse |
| Week 5 | Begin and Finish Subtask 1.5 | Shanice Nurse |
| Week 6 | Begin Subtask 2.1, 2.2 | Task 2: Clam activity level image processing code | Ashley Afueh |
| Week 7 | Finish Subtask 2.1, 2.2 | Ashley Afueh |
| Week 8 | Begin Subtask 2.3, 2.4 | Ashley Afueh |
| Week 9 | Finish Subtask 2.3, 2.4 | Ashley Afueh |
| Week 10 | Begin Subtask 2.5 | Ashley Afueh |
| Week 11 | Finish Subtask 2.5 | Ashley Afueh |
| Week 12 | Begin and Finish Task 3 | Task 3: Network Capturing Code | Shanice Nurse |
| Week 13 | Begin and Finish Task 4 | Task 4: Code to send sensor data to Raspberry Pi | Ashley Afueh |
| Week 14 | Finish subtask 5.1-5.3 | Task 5: Code for data organization in Raspberry Pi | Ashley Afueh |
| Week 15 | Finish Subtask 5.4-5.5 | Ashley Afueh |
| Week 16 | Finish Subtask 6.1 – 6.3 | Task 6: Code to send Raspberry Pi data to cloud | Shanice Nurse |
| Week 17 | Finish Subtask 7.1 – 7.3 | Task 7: Create Website Database | Ashley Afueh |
| Week 18 | Finish Subtask 7.4 – 7.5 | Ashley Afueh |
| Week 19 | Finish Subtask 8.1 – 8.3 | Task 8: Create App Database | Shanice Nurse |
| Week 20 | Finish Subtask 8.4 – 8.5 | Shanice Nurse |
| Week 21 | Finish Subtask 9.1 – 9.4 | Task 9: Lab Setup | Ashley Afueh, Shanice Nurse |
| Week 22 | Finish Subtask 10.1 – 10.5 | Task 10: System Evaluation | Ashley Afueh, Shanice Nurse |

1. **Implementation**

4.1: Implementation of Task 1:

The first step to start this project is to make sure the four cameras mounted on each side of the tank are able to identify the image of clams. To be able to do this, a convolutional neural network will be created that will be able to identify clams through image processing. By using python, a code will be developed such that, when a dataset of pictures is given to the system, it will be able to determine which pictures are clams or not.

4.1.1 *Implementation of Subtask 1.1*

To prepare the convolutional neural network model to be able to read images of clams, there must be images of clams available. A dataset will be developed including a vast library of images. Half of this library will be of pictures of clams, and the other half will not be of pictures of clams. These pictures will be uploaded to a folder to be later used in the python programming to introduce a dataset for the image processing.

4.1.2 *Implementation of Subtask 1.2*

A baseline cnn model must be made to set the standard for the minimal requirements needed for image recognition. This baseline will serve as a model architecture that can be used to study the images and improve the image identification as well. The model architecture includes 3x3 convolutional layers and a mac pooling layers. These layers form blocks, and the number of filters in the block increase as the network’s depth increases. Each layer then goes through an activation function and weight initialization. Then a function that defines a model is created. This function can be edited to define various baseline models. The data is then prepared and iterators are prepared for the dataset.

4.1.3 *Implementation of Subtask 1.3*

Using the baseline model created from the previous task, we will improve on this model to improve the performance of the image processing. First, we will introduce three VGG (Visual Geometry Group) blocks into the baseline model. These blocks will help us to explore improvements to the model. When the model displays signs of overfitting, this can be fixed by dropout regularization and data augmentation. Dropout regularization removes inputs into a later in order to regularize the neural network. Data augmentation expands the size of the dataset by modifying the images in the dataset. When the images are modified, it adds “noise” to the data, making the model learn the same features. This results in more accurate results for image processing.

4.1.4 *Implementation of Subtask 1.4*

To implement transfer learning, a various amount of pre-trained models will be used to be trained on a related task. The model includes a feature extractor and the classifier part of the model. The feature extractor is made of VGG blocks. Classifiers are added to the feature extraction also. When a new model is created, it is trained using the dataset. The model is tested using various examples, and reports of the model performance is recorded on datasheets.

4.1.5 *Implementation of Subtask 1.5*

Model improvement can be continued as long as desired as long as there are datasets to test them on. For starters, a final dataset that is fit on all available data must be prepared. This dataset is then fit on a final model. This final model will be saved, and used to make predictions on new images.

4.2: Implementation of Task 2:

The next step to start this project is to make sure the four cameras mounted on each side of the tank are able to identify the level of clam activity – meaning whether it is a large or small clam opening. To be able to do this, a convolutional neural network will be created that will be able to identify the type of clam opening through image processing. By using python, a code will be developed such that, when a dataset of pictures is given to the system, it will be able to determine which pictures of clam openings are big or small.

4.2.1 *Implementation of Subtask 2.1*

To prepare the convolutional neural network model to be able to read images of clam activity levels, there must be images of clams available. A dataset will be developed including a vast library of images. The library will consist of big clam openings and small clam openings. These pictures will be uploaded to a folder to be later used in the python programming to introduce a dataset for the image processing.

4.2.2 *Implementation of Subtask 2.2*

A baseline cnn model must be made to set the standard for the minimal requirements needed for image recognition. This baseline will serve as a model architecture that can be used to study the images and improve the image identification as well. The model architecture includes 3x3 convolutional layers and a mac pooling layers. These layers form blocks, and the number of filters in the block increase as the network’s depth increases. Each layer then goes through an activation function and weight initialization. Then a function that defines a model is created. This function can be edited to define various baseline models. The data is then prepared and iterators are prepared for the dataset.

4.2.3 *Implementation of Subtask 2.3*

Using the baseline model created from the previous task, we will improve on this model to improve the performance of the image processing. First, we will introduce three VGG (Visual Geometry Group) blocks into the baseline model. These blocks will help us to explore improvements to the model. When the model displays signs of overfitting, this can be fixed by dropout regularization and data augmentation. Dropout regularization removes inputs into a later in order to regularize the neural network. Data augmentation expands the size of the dataset by modifying the images in the dataset. When the images are modified, it adds “noise” to the data, making the model learn the same features. This results in more accurate results for image processing.

4.2.4 *Implementation of Subtask 2.4*

To implement transfer learning, a various amount of pre-trained models will be used to be trained on a related task. The model includes a feature extractor and the classifier part of the model. The feature extractor is made of VGG blocks. Classifiers are added to the feature extraction also. When a new model is created, it is trained using the dataset. The model is tested using various examples, and reports of the model performance is recorded on datasheets.

4.2.5 *Implementation of Subtask 2.5*

Model improvement can be continued as long as desired as long as there are datasets to test them on. For starters, a final dataset that is fit on all available data must be prepared. This dataset is then fit on a final model. This final model will be saved, and used to make predictions on new images.

4.3: Implementation of Task 3:

This task will send the pictures taken from the cameras to the raspberry pi.

*4.3.1: Implementation of Subtask 1:*

A network capturing code will be created using python. This code will send the pictures depicting the clam activity to the raspberry pi.

4.4: Implementation of Task 4:

There will be sensors in the tank that determine the temperature levels, pH levels, oxygen levels, and light levels of the tank. There is going to be a standard setting for all these parameters, and any deviation from the standard setting for these parameters will alert the database.

*4.4.1: Implementation of Subtask 4.1:*

Using python, a code will be created to determine if there is any deviation from the standard settings for the sensors. If there is, an alert will be sent to the website and app.

4.5: Implementation of Task 5:

Once all the data gathered from the camera and sensors is sent to the raspberry pi, this data must be stored in an organized manner. This task is to create a structural form of data that makes data collection easier to sort through.

*4.5.1: Implementation of Subtask 5.1:*

Using python, folders will be created in the raspberry pi. Data will be stored in these folders. Different folders will include different information, including sensor information, general clam activity, general clam activity level, individual clam activity, and individual clam activity level.

*4.5.2: Implementation of Subtask 5.2:*

The data is written into a CVS file. A CVS file is a file with simple plain text. The CVS file will be written in python.

*4.5.3: Implementation of Subtask 5.3:*

Next, the data will be modeled with a miniature database. The miniature database that will be used is SQLite. This database is very popular amongst mobile and handheld devices. Even though CVS files make data logging simple, it is still difficult to search through data with a CVS file. Because of this, there will be database engine used alongside the raspberry pi. Using python, a database will be setup for this data.

*4.5.4: Implementation of Subtask 5.4:*

The python code will run and generate and generate a “.db” file. SQLite will generate a file that allows us to play with the data.

*4.5.5: Implementation of Subtask 5.5:*

The clam activity data collected by each individual camera will be compared to one another in order to assure all four cameras are getting equal and accurate results. The amount of clam openings per hour, and the type of clam activity that each opening had, will be compared. The expectation is for there to be little to no error between the data the cameras obtained, with an error percentage expectancy of less than 1%.

4.6: Implementation of Task 6:

The data collection obtained from the raspberry pi will be sent to a public cloud-based online machine called “Google Cloud”. This data will be sorted into organized folders in order to have the clam activity easily classified.

*4.6.1: Implementation of Subtask 6.1:*

A channel for the data is created in this step. This channel is essentially a folder that will hold all of the data from the raspberry pi in an organized fashion. This folder is created using python.

*4.6.2: Implementation of Subtask 6.2:*

In this step, an API key is created. This key is what is used in the python code to upload things to the website and app database. The API key is copied onto the clipboard for the next step.

*4.6.3: Implementation of Subtask 6.3:*

A python code for the raspberry pi is created, and this code makes it such that the data is sent to “Google Cloud”.

4.7: Implementation of Task 7:

Here, one out of the two databases are formed – the website database. This is done using a website creation software called *pantheon.*

*4.7.1: Implementation of Subtask 7.1*

The layout for the website is created in this step. How the website is going to look is determined here. A site code is created on pantheon to create the various tabs and links that can be utilized to view the clam activity. The user will have options of how they want the data depicted.

*4.7.2: Implementation of Subtask 7.2*

The general clam activity is the clam activity of all clams as a whole. The general clam activity and activity level will be sorted into its own separate groups accordingly. There will be two categories for this: general clam activity, and general clam activity level. The general clam activity will depict the number of openings per each hour, while the general clam activity level will depict the *type* of clam activity for each of these hours, explaining whether they’re were big clam openings or small clam openings. The user will have options of how they want this data depicted. The amount of big and small clam openings per hour will sum up to the total amount of general clam openings per hour. A python code will run to depict the data in this format.

*4.7.3: Implementation of Subtask 7.3:*

The individual clam activity is the clam activity of each clam separately. Each separate clam’s activity level is going to be sorted into its own separate groups accordingly. There will be two categories for this: individual clam’s general activity, and the individual clam’s activity *level.* The individual clam activity will depict how many openings each individual clam had in total, and the individual clam activity level will depict the *type* of opening the clams had, explaining whether they are big or small. The amount of big and small clam openings per hour will sum up the total amount of general clam openings per hour for each individual clam. A python code will run to depict the data in this format.

*4.7.4: Implementation of Subtask 7.4:*

This step will create figures for graphs on the website that depict clam activity data. Using a feature on pantheon called “New Relic APM Pro”, graphs are created when data is entered. These graphs have many different graphical design settings, which in turn helps to distinctively visually depict the clam activity easily.

*4.7.5: Implementation of Subtask 7.5:*

The website will be tested. Over the course of 24 hours, the website will collect enough data for a daily update. This daily update on the website will be viewed, checked for errors, and corrected.

4.8: Implementation of Task 8:

Here, the other one out of the two databases are formed – the app database. This is done using an app creation software called *MIT app inventor.*

*4.8.1: Implementation of Subtask 8.1*

The layout for the app is created in this step. How the app is going to look is determined here. A site code is created on MIT app inventor to create the various tabs and links that can be utilized to view the clam activity. The user will have options of how they want the data depicted.

*4.8.2: Implementation of Subtask 8.2*

The general clam activity is the clam activity of all clams as a whole. The general clam activity and activity level will be sorted into its own separate groups accordingly. There will be two categories for this: general clam activity, and general clam activity level. The general clam activity will depict the number of openings per each hour, while the general clam activity level will depict the *type* of clam activity for each of these hours, explaining whether they’re were big clam openings or small clam openings. The user will have options of how they want this data depicted. The amount of big and small clam openings per hour will sum up to the total amount of general clam openings per hour. A source code will run to depict the data in this format.

*4.8.3: Implementation of Subtask 8.3:*

The individual clam activity is the clam activity of each clam separately. Each separate clam’s activity level is going to be sorted into its own separate groups accordingly. There will be two categories for this: individual clam’s general activity, and the individual clam’s activity *level.* The individual clam activity will depict how many openings each individual clam had in total, and the individual clam activity level will depict the *type* of opening the clams had, explaining whether they are big or small. The amount of big and small clam openings per hour will sum up the total amount of general clam openings per hour for each individual clam. A source code will run to depict the data in this format.

*4.8.4: Implementation of Subtask 8.4:*

This step will create figures for graphs on the website that depict clam activity data. Using a feature on pantheon called “New Relic APM Pro”, graphs are created when data is entered. These graphs have many different graphical design settings, which in turn helps to distinctively visually depict the clam activity easily.

*4.8.5: Implementation of Subtask 8.5:*

The app will be tested. Over the course of 24 hours, the website will collect enough data for a daily update. This daily update on the app will be viewed, checked for errors, and corrected.

4.9: Implementation of Task 9:

In this task, the lab setup for the experiment will take place.

*4.9.1: Implementation of Subtask 9.1*

The cameras will be mounted to the sides of the tank using a raspberry pi camera mount.

*4.9.2: Implementation of Subtask 9.2*

The light sensor, temperature sensor, pH level sensor, oxygen level sensor, and light sensor will be mounted to the tank. The temperature sensor will consist of a thermometer in the tank that communicates with the sensor outside of the tank to determine temperature. The light sensor will be mounted outside of the tank to determine the amount of light the tank is receiving. The pH sensor and oxygen sensor will have sensors in the tank that communicate with parts outside of the tank to determine pH level and oxygen.

*4.9.3: Implementation of Subtask 9.3*

The raspberry pi will be set up, and the various codes created throughout the experiment for the raspberry pi will be installed in the raspberry pi via flash drive

4.9.4: Implementation of Subtask 9.4

Any other task in the setup of the experiment will be set up (wire configuration, wire placement, etc.)

4.10: Implementation of Task 10:

In this part of the project, the system will be evaluated and given accuracy percentages.

*4.10.1: Implementation of Subtask 10.1:*

The data for the individual clam data and general clam data will be collected during various different hours per day. Each camera will have its own data. Because of this, there will be four different types of data for each camera. The averages of this data will be compared to determine the accuracy of the cameras. The expected error rate is under 1%.

*4.10.2: Implementation of Subtask 10.2:*

The entire system is tested. Over a 24-hour period, the cameras will be monitoring the clam activity of the clams. This data will be sent from the cameras to the raspberry pi, then from the raspberry pi to the cloud to the website. The website will give all results of the data collected in the past 24 hours. The results detected will be reviewed to assure realistic data was obtained.

*4.10.3: Implementation of Subtask 10.3:*

Any errors found during the testing of the system will be fixed.

*4.10.4: Implementation of Subtask 10.4:*

The system will then be evaluated and its total accuracy level will be determined.

*4.10.5: Implementation of Subtask 10.5:*

The system will be refined if necessary.

1. **Project evaluation**

In this chapter, please evaluate the performance of the solution completed in the project with the considerations of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors. And please consider improvement suggestion for the future on the aspect.

1. Public health
2. Safety
3. Welfare
4. Global factor
5. Cultural factor
6. Social factor
7. Environmental factor
8. Economic factor
9. **Conclusion**

By the end of the project, conclude the project and your learning experience.

Please make sure you include the overall evaluation of the project and the **future plan** about how the project can be improved.

* Project summary

…

* Learning and practice experience

…

* Future plan (how to improve)

…

**Acknowledgment**

If you get help or support from someone else (besides the team member and the advisor) and want to show your appreciation, put here (**do not include the advisor**).

**Appendix**

You can put reference info here, including i) specs of components used in the system, ii) source code (must be here but not in the body text), iii) CAD figures, etc.

1. **Component Specs**
2. ***Specs of Arduino Due***

...

1. ***Specs of Raspberry Pi***

…

1. **Source Code.**
2. ***Source Code of Graphic User Interface***

…

1. ***Source Code of Robotic Arm***

…

**REFERENCES**

[1] D. Vantrease, R. Schreiber, M. Monchiero, M. McLaren, N. P. Jouppi, M. Fiorentino*, et al.*, "Corona: System Implications of Emerging Nanophotonic Technology," in *Computer Architecture, 2008. ISCA '08. 35th International Symposium on*, 2008, pp. 153-164.

[2] X. Zhang and A. Louri, "A multilayer nanophotonic interconnection network for on-chip many-core communications," in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*, 2010, pp. 156-161.

[3] C. Batten, A. Joshi, J. Orcutt, A. Khilo, B. Moss, C. Holzwarth*, et al.*, "Building manycore processor-to-DRAM networks with monolithic silicon photonics," in *High Performance Interconnects, 2008. HOTI '08. 16th IEEE Symposium on*, 2008, pp. 21-30.

[4] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A. Choudhary, "Firefly: illuminating future network-on-chip with nanophotonics," in *IEEE/ACM Intl. Symp. on Computer Architecture (ISCA)*, 2009, pp. 429-440.

[5] N. Kirman, M. Kirman, R. K. Dokania, J. F. Martinez, A. B. Apsel, M. A. Watkins*, et al.*, "Leveraging Optical Technology in Future Bus-based Chip Multiprocessors," in *Microarchitecture, 2006. MICRO-39. 39th Annual IEEE/ACM International Symposium on*, 2006, pp. 492-503.

[6] J. M. Cianchetti, C. J. Kerekes, and H. D. Albonesi, "Phastlane: a rapid transit optical routing network," in *Proceeding of: 36th International Symposium on Computer Architecture (ISCA)*, 2009, pp. 441-450.

[7] A. Shacham, K. Bergman, and L. P. Carloni, "Photonic Networks-on-Chip for Future Generations of Chip Multiprocessors," *Computers, IEEE Transactions on,* vol. 57, pp. 1246-1260, 2008.

[8] A. Shacham, K. Bergman, and L. P. Carloni, "On the Design of a Photonic Network-on-Chip," in *First International Symposium on Networks-on-Chip, 2007. NOCS 2007*, 2007, pp. 53-64.

[9] M. Kwai Hung, Y. Yaoyao, W. Xiaowen, Z. Wei, L. Weichen, and X. Jiang, "A Hierarchical Hybrid Optical-Electronic Network-on-Chip," in *VLSI (ISVLSI), 2010 IEEE Computer Society Annual Symposium on*, 2010, pp. 327-332.

[10] D. Ding and D. Z. Pan, "OIL: a nano-photonics optical interconnect library for a new photonic networks-on-chip architecture," presented at the Proceedings of the 11th international workshop on System level interconnect prediction, San Francisco, CA, USA, 2009.

[11] A. Joshi, C. Batten, K. Yong-Jin, S. Beamer, I. Shamim, K. Asanovic*, et al.*, "Silicon-photonic clos networks for global on-chip communication," in *Networks-on-Chip, 2009. NoCS 2009. 3rd ACM/IEEE International Symposium on*, 2009, pp. 124-133.

[12] D. Vantrease, R. Schreiber, M. Monchiero, M. McLaren, N. P. Jouppi, M. Fiorentino*, et al.*, "Corona: system implications of emerging nanophotonic technology," in *Proc. 35th IEEE/ACM Int'l Symp. Computer Architecture (ISCA)*, 2008, pp. 153-164.

[13] L. Zhang, E. Regentova, and X. Tan, "A 2D-Torus Based Packet Switching Optical Network-on-Chip Architecture," presented at the *IEEE International Symposium on Photonics and Optoelectronics* (SOPO 2011), Wuhan, China, 2011.

[14] L. Zhang, E. E. Regentova, and X. Tan, "Packet switching optical network-on-chip architectures," *Comput. Electr. Eng.,* vol. 39, pp. 697-714, 2013.

[15] G. Huaxi, X. Jiang, and W. Zheng, "A novel optical mesh network-on-chip for gigascale systems-on-chip," in *Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on*, 2008, pp. 1728-1731.

[16] G. Huaxi, X. Jiang, and Z. Wei, "A low-power fat tree-based optical Network-On-Chip for multiprocessor system-on-chip," in *Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE '09.*, 2009, pp. 3-8.

[17] Y. Yaoyao, X. Jiang, H. Baihan, W. Xiaowen, Z. Wei, W. Xuan*, et al.*, "3-D Mesh-Based Optical Network-on-Chip for Multiprocessor System-on-Chip," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 32, pp. 584-596, 2013.

[18] A. Shacham, K. Bergman, and L. P. Carloni, "Photonic networks-on-chip for future generations of chip multiprocessors," *IEEE Trans. Computers,* vol. 57, pp. 1246-1260, 2008.

[19] A. W. Poon, F. X. Xu, and X. Luo, "Cascaded active silicon microresonator array cross-connect circuits for WDM networks-on-chip," in *Proc. SPIE*, 2008, p. 689812.

[20] M. Lipson, "Compact Electro-Optic Modulators on a Silicon Chip," *IEEE Journal of Selected Topics in Quantum Electronics,* vol. 12, pp. 1520-1526, 2006.

[21] M. Lipson, "Guiding, modulating, and emitting light on Silicon-challenges and opportunities," *Lightwave Technology, Journal of,* vol. 23, pp. 4222-4238, 2005.