**ARF (Autonomous Remote Feeder)**

**Team Name**: Cold Fries  
**Team Members**:  
Kenny Do Davin Sonexarth  
Kevin Perez Michael Nguyen  
Khai Vu

**Introduction/Object/Purpose**

The responsibilities of being a pet owner will sometimes keep you from being able to get out and away from the house for prolonged periods of time. The need to repeatedly feed animals daily is a menial task that we hope to relieve pet owners from - giving them extra time in their day and removing the chance of missing a feeding time. Our product, the ARF (autonomous remote feeder), utilizes our skills as electrical and software engineers to build a feeding machine that can repeat daily tasks for the owner in their absence. The coding and software side of our group is using the internet and applications to be able to connect Raspberry Pi to the structure and give the owner options that would allow them to schedule feeding times and connect in different ways with the pet while the owner is away, this ensures the pets are fed at the optimal times by their owners.

**Scope**

This work statement will outline the planning of the project as well as the implementation of the autonomous remote feeder (ARF). Some of the work towards the end item was developed, though highly limited. The scheduled timeline of tasks will be outlined below, as well as both the hardware and software requirements needed of the product.

All dates, deadlines, and descriptions of items below are subject to change as the need arises for the duration of the project planning and implementation.

**Timeline**

**Week 1** : Take all of first semester and combine it into one product

Purchase any unprocurable first semester electrical parts immediately. Any research still needed done here. Speak with Andy about ways to implement the mechanical design. Remind each other where we left off and ensure everyone has tasks to do.

Currently, we bought two separate raspberry pis so that we could have both the hardware and the software side work on the raspberry pi at the same time. Since this happened, we have two functionalities running on two separate pis, thus, we need to take both and combine them into one.

Take a look at designing a circuit board for this specific project

**Week 2** : Start designing the outer shell & clean up code

Any research still needed done here. Work with the team to hone the concepts for implementing the design and mechanical needs of the feeder.

We need to finalize on the dimensions so that we can start creating a mock up of the ARF through CATIA. From there, we need to find the cheapest available option so that we can create the housing cost effectively. This will require a custom design so that it can not only hold all the components, but also incorporate some third party components to aid in the holding and dispensation of the food.

Currently we have some beta code, it does get the job done but since it is rough code, we need to cut it down so that we can be more efficient. We also must continue writing new code so that we can incorporate all the extra additions that we want for the website such as a scheduled feeding

**Week 3** : Find out working mechanics to reduce electrical load / Start creating Website to communicate with internet

Make sure all electrical plans have all parts functioning.

Stated similarly before, we need to compare the output that the raspberry pi can achieve compared to the amount needed for all of the components to function. Currently, our components use more amperage than the raspberry pi can output, so we are looking into implementing mechanical aspects in order to help achieve what is needed while also reducing the amount of electricity needed.

Begin to procure all mechanical parts needed. Decide on exact implementation of end goal product

**Week 4 - 5** : Communicate all of the hardware with software / Create a working link button for internet connection Finish procuring all mechanical parts needed and decide on exact implementation of end goal product

Currently we have the hardware and the software connected to each other and communicated. Since we have not used our official website, we want to migrate all of our code to the website domain and allow for a communication signal. In addition to that, the hardware and the software are able to communicate through the IoT through private networks such as homes. We tested these on the school internet but were quickly blocked by the firewall, so we need to look into a way to get around that and still establish a connection

**Week 6** : Implementation of hardware / Creating functionality on website

Start building the prototype (runs week 6 through 12)

After all of the research done and the prototyping, we want to put all of that together inside of the housing to see if any changes need to be made and to see if it still works. We want to keep all of the electronics compact so that it takes the least amount of space (which is more appealing for consumers).

We also want the website to start coming together, hoping for us to have a layout and user friendly GUI so that it will be more appealing to the user. We also want the website to have buttons working, to show communication between the pi through the IoT and to the user.

**Week 7** : Find a way to dispense the food and handle all of the mechanical aspects in relation to electrical.

Adding any mechanical aspect that may aid in the production of the project.

Expound upon the code for the electrical components. EE and CS will work together to implement the correct functions needed.

**Week 8** : Re-evaluate methods, look for ways to optimize all parts / Continue to implement all mechanical, hardware, software

After getting a working project, we want to find a way so that we as a group can still accomplish the same end goal in a more efficient pattern, whether that is reducing power draw, allowing for faster response time from user -> IoT -> pi.

**Week 9 - 10** - Implementation

Taking all the prototypes from the software and hardware and meshing them together

**Week 11 - 12** : Clean up all hardware and software to make it more presentable and efficient

Creating a more presentable design, cleaning up the design and making it more appealing to the eye so that it catches more attention.

Focusing more on the looks of the website, so that users can use it without hassle

Prototype mechanical finished and electrical fully installed.

**Week 13 - 14** : Final Testing/Debugging & Finishing Product

Wrapping up our project and seeing if there are any problems with it. If there are, we have ample time in order to fix it. No longer improving and/or adding to the project, reducing the chances for an unexpected error.

**Week 15** : Prep for presentation & Finish Product

Giving ourselves a week to prepare for the presentation, setting up whatever is needed to provide us with optimal time so that we can ensure functionality

**Week 16** : Presentation week

Arrive early to set everything and begin the presentation.

**Location of Work**

The main areas of work we did for the first half of our prototyping was done individually at our own homes. The CS side used the second floor library lounge to discuss code integration with the raspberry pi, while the EE/CE side mainly did work at each other's homes since it is more convenient having all their equipment at home, such as wires, breadboards, etc. After the software side wrote up some sample code and the hardware side tested hardware connections, we all met up to integrate our contributions at the JBC. We used various different areas from the study rooms, lounges, and computing labs.

**Acceptance Criteria**

Looking at what needs to be done in order for the ARF project to be considered complete, we know that these critical aspects must be implemented

* Internet connectivity (We cannot do it on a local server)
* Functioning website with a proper GUI
* Special case error checking, in case of things like wifi disconnection, power surges, etc.
* Ability to physically feed incase of things like no power (can be as simple as removing the food from the container).
* Power transformation of some sort (so that the working components do not get overloaded with an excessive amount of current)
* Container in which can hold and then dispense the food that is put into it
* A functioning servo motor and/or different way to dispense a proper amount of food
* Firewall and internet bypass so that we can connect the device to the user’s internet so that they can communicate with the ARF

**Hardware Acceptance**

Functioning parts will include a motor to dispense food, camera to see the animal, sensors to let the user know when the hopper needs to be refilled with food, sensors to show that food was dispensed in the bowl successfully, touch screen with basic functionality connecting to the internet that can allow for touch dispense and be able to see metrics and allow for programming the feeder. A backup battery will also be included in the final product for power outages to sustain the feeder for at least 12 hours.

Functioning product housing will include a hopper, dispensing tube and a well thought out design to house the project. A tight lid must be secured to the top of the hopper to not allow for animals breaking into the food. The housing must be sturdy enough to hold 10 lbs of food and not deteriorate or collapse when run into with some force or weight.

Each part will be subjected to rigorous testing involving at least 100 dispenses, continuously checking for errors and improving the code to hardware to fit the design.

**Software Acceptance**

The acceptance of our software relies on it being able to run self-sufficiently and without the need for the consumer to run any code. Since our product is based on wireless activation via web servers, we do not want our consumer to be forced to run any code or do any extra setup such as port-forwarding. Once our software is able to run either automatically on startup, or with very minimal input that does not require the user to know any extra information, then it will be usable for our product. It needs to also contain the necessary functions and features such as recording feeding logs, scheduling, and near 1:1 response time when wanting to dispense immediately..

Additionally, it will be required that the raspberry pi can successfully link to the internet using WPA configuration. Once it is linked it will have access to the website to do the necessary operations such as turn off/on LED run the motors etc. This will all be connected through Rasp Gpio to send signals to the designated pins and run power through them. At first glance the user should know if it successfully connected or not to the internet if the buttons are as they are intended. The users should also be able to tell if the pins are properly powered if the desired piece of equipment is operating or not. Users can tell through both these means to see if the raspberry pi is receiving any of the needed signals or instructions.

**Work Activities**

Overall, both sides still need to research as to what it is we need in order to refine our product so that it can be optimized. One important aspect that we are looking into is either a way to reduce the amount of amperage that is being used to power the components compared to the capabilities of the raspberry pi headers and their maximum amperage output. Currently we have some ideas that we can try, but much more research is needed in order to actually complete this. For the website based side, we still need to get our website database to form a solid stable connection to the web server hosted on our raspberry pi. At the current state, we have them as separate entities, in other words, they are not able to read or write to each other. We will need to do a bit more research and trial and error code testing to see if we can connect our database directly to the raspberry pi, or host it directly on the pi itself. In order for the website to work we linked it through a server and once the raspberry pi had established a stable connection from the internet it would be able to execute what was programmed on the website such as buttons, switches and other sorts. We used raspi-config to set the boot option to automatically login as pi so this would act as any other wifi device would. Once given the password or necessary information it would store it for later use. In this case we are setting it so that it will always remember to use Pi as the login and skip the boot login process.

**Software**

Looking at the project from a software aspect, we find that the main thing that needs to be done is the website. Since the website is the main way for the users to send in an input, this means that the website not only has to have a user friendly GUI, but also needs a way for each user to create their desired output. With that, we are looking to create a scheduling aspect in which the pet feeder can automatically feed their pet at a certain period of time. This will be accessed from the website, where there will also be a manual feed button. The website should include options to change local time, specify the amounts to feed, and if it should be put on a repeating cycle, so consumers do not have to set it up for every single day. Since this is a website, we will most likely have a main website host where customers have their own specific login. We plan to have a user and password system that can differentiate which customer has what access to which ARF. Since this is a household product, we will need a secure login method, so that unauthorized individuals are unable to access the ARF website.

The web client will be storing the scheduling and history log within a local database connected to the device. The web client will also be hosted on the raspberry pi, meaning that the raspberry will be acting as a server for both the site and the database hosted on it. This will be set up using a LAMP that will need to be set up and configured on the raspberry pi.

Using the wpa config we are added a new file and nothing changes anything of the Rasp and this would allow us to set up the wifi with ease as long as the user had knowledge of the username and password to the given wifi they desire to connect to. This would act as any other modern device as once it detects the wifi it will automatically connect. Once the file has been made, a button can be coded to have this indeed purpose of connection and with one tap it should automatically link to the wifi as the file should store the needed information to establish a connection.

**Hardware**

Looking at the hardware aspect, the main components that need to be working are the microcontroller (in our case, it is the raspberry pi), the food dispensation mechanism (the servo), and many other electrical aspects. The main focus will be the raspberry pi, since this is the way that communicates the software and the hardware, we need a way for the pi to not only communicate to the user, but also output enough power so that all of the given mechanisms can function as intended. The servo will be responsible for dispensing a certain amount of food at any given time.

**End Item Deliverables**

Our end item software components should offer all the features needed to feed a pet, including setting a feeding time, and on-demand manual feeding. The code should be able to function regardless of any combination of manual or randomized input, meaning it should encounter no bugs or errors at any point. The code should also be able to run in a constant indefinite manner, with case checking for whether something such as a wifi disconnection occurs. Our end item will also be able to track and record the user input into a database to set automatic feeding, along with being able to check and notify the system if there is either no food, or low food remaining.

**End Item Test Requirements**

For the end product, all functions will need to be tested. Both the manual and automatic feeding will need to be tested multiple times. Amounts of food dispensed needs to be semi-measurable to amounts the end user would want to see given to their pets. Increments from quarter cup to 1 full cup will need to be done; as well as every quarter amount between them (½, 3/4ths). We will have to show that food is running out and be able to send that information to the end user. We will have to be able to capture pictures of animals depending on feeding times and send them to the end-user. We will have to show that there is food in the bowl to the end user all over the internet connection. The web client will need to be tested on different browsers. The distance between the user and product will need to be taken into consideration, to make sure that the user will be able to use the web client, no matter where they are. The scheduling will also need to be tested to see how accurate the dispensing time will be to the scheduled time. A manual dispense will also need to be tested on this to make sure it happens almost as soon as the button is hit.

The end product of the internet connectivity aspect would be to have the raspberry pi run seamlessly through the website and no missing signals from either end and it is a smooth process for the user. Button functionality is similar to a roku where you click a button and it does what it was programmed for.

By clearly printing our names here, each member of the group has confirmed that they have read the entire report and it meets their personal quality standards

[X] Kenny Do

[X] Davin Sonexarth

[X] Kevin Perez

[X] Michael Nguyen

[X] Khai Vu