PipeWan

Project Team: PipeWan

Team Members:

Alex Rossillon

Aron Wiley

Edgar Chavez

Mohammad Daoud

# Introduction, Purpose, and Objective

Our pipe monitoring system involves two parts. First, a device attaches to water pipes and monitors the temperature and flowrate of water. This device then sends its data remotely to our application. Together, our system will serve to ensure the safety of pipes from bursting by warning building owners before their pipes burst when the temperature or flow rate indicate a possible failure.

# Scope

This work statement contains plans for the second version of our pipe monitoring system. The updated hardware and software requirements are stated in the following sections.

Dates and features stated below are subject to change throughout the following time during our research, building, and testing of the final product. The timeline will shrink and grow in response to variation in completion time of each part. Additional time has been allotted to milestone in order to account for potential issues we might encounter.

# Location of Work

Most of our physical design and testing will take place in the John Bardo Center, namely in the GoCreate Lab. The lab is well equipped with tools and devices we will need especially for the fabrication of parts in our nodes such as CATIA software, multimeters, soldering irons, etc. The lab also houses LoRaWan gateways which are the essence of our project. These gateways are essential for integrating our node into the LoRaWan network through OTAA activation. Software programming will take place in various locations such as team members’ homes, in JBC labs, or common areas. Once we have a minimal viable product, testing will take place throughout the JBC.

# Work Schedule

**Week 1:**

* Research API methods to connect computer programs to ChirpStack
* Connect antenna to SparkFun and send data to application wirelessly

**Week 2:**

* Choose a suitable battery, O-ring or silicone, and method of affixing the temperature probe to the outside of a metal pipe.
* Implement a battery relay to allow for multiple batteries.

**Week 3 – 9:**

* Perform regular power draw tests to verify changes are actually reducing the electricity consumption of the Arduino. Research LoRaWan best practices for data packaging and message sending
* Improve the programs visual aesthetic as determined by a majority vote of group members, in case of a tie we will collect tie breaking votes from outside the group.
* Add the following features to the UI
  + Show battery percentage
  + Show connectivity strength
  + Update graphs
  + Add different themes (light mode, dark mode)
  + Log in page
  + Water pressure
  + Total amount of water flow
* Add the following feature to the program backend
  + Notification system: Email and SMS
  + Verify login
  + Connect to database
  + Code to implement front-end features (battery, connectivity strength, etc.)
* Create unit tests to verify that the code functions properly.
* Create database

**Week 10 – 12:**

* Allow the Arduino to be put asleep remotely and to be woken up remotely.
* Control the rate of data from the node to the application to save battery.

**Week 13:**

* Determine a satisfactory water resistance rating, design a node to meet that rating, and test the water resistance of the node.
  + For example, IP67 implies the node could survive under 1 meter of water for 30 minutes.
* Print the final version of the node capable of meeting the selected rating and improved visual aesthetic as determined by a majority vote of group members, in case of a tie we will collect tie breaking votes from outside the group.

**Week 14 – 16:**

* + Testing completes when all features work continuously for one week.

# Tasks and Deliverables

|  |  |  |  |
| --- | --- | --- | --- |
|  | Requirement | Description | Schedule |
| 1. | Weekly meetings and individual journals | This will serve as a weekly progress checkpoint where team members will share their progress, concerns, and make sure our schedule is fulfilled.  These meetings will occur in person or virtually when needed. All members will be expected to attend and participate. | At least once a week |
| 2. | Integrate with LoraWan | We will attach the antennae to our node then use existing gateway to make our node visible on the ChirpStack. |  |
| 3. | Meet with Sponsor | Based on our progress, we will try to set up regular meetings with our sponsor to capture the more specific project requirements. | Bi-Monthly |
| 4. | Hardware design/testing | The CE members will focus on making sure the existing hardware meets compliance standards. Then, design new hardware that fits the scale of implementation and perform testing. |  |
| 5. | Software Development Requirements | The CS members will focus on finding documentation and requirements for:   * Libraries * Connecting to hardware * Coding environment |  |
| 6. | Software testing | The CS members will verify that the code they have programmed works correctly with all devices, and will conduct a code review with at least one (1) other CS member. A unit test of the code is also required for each piece of code written. | Bi-Monthly |
| 7. | Environment Testing | The CE members will conduct site surveys and generate heatmaps to determine optimal placement of antennae.  The team will also determine mitigations for possible environmental impacts on our product like water, obstacles, and RF interference. |  |
| 8. | Electrical design/testing | We will include different electrical components that will provide power and continuity for the product:   1. A suitable battery 2. Omni Antennae 3. Electronic Relay 4. RF filters   Once the product is complete it is the CE members’ job to ensure these components are compliant with industry standards and serve their functionality properly. |  |
| 9. | Software implementation | All group members will verify that the software runs correctly with all the hardware. |  |
| 10. | Project research | Research will take place throughout the semester. It will help us improve initial data, gather feedback, and ensure compliance. |  |
| 11. | Product Implementation | After all the testing has been complete, we will move to implementing our product in a real-life situation. |  |

# Work Performance

Our work schedule is listed in the previous section. We as a team will meet once a week to discuss further plans and progress for the current week and decide our agenda for the following week.

# Acceptance Criteria

Testing our system will consist of software and hardware validation. For our software we will make sure all features function correctly through different test cases. For the hardware itself we will ensure that the device can withstand certain conditions while being able to function properly.

Software Testing

* Validate all features work properly and have individual methods for validation
  + Receiving the flow rate: Ensure we are receiving the flow rate and then validate the flow rate for different test cases to distinguish between a normal and an abnormal flow rate.
  + Temperature: Ensure that the temperature being received is valid and within an acceptable margin of error, we can also verify the temperature using a known good temperature probe.
  + Battery Percentage: Send the command to find battery percentage then verify by using a voltmeter to verify the accuracy of the command.
  + Database Connected: Run the AT command to check signal strength, a strength enough to send data will be considered passing. This feature should be tested in multiple locations to verify the node will work across campus.
  + Water Pressure: Calculate water pressure by taking the current flowrate and the known diameter of the pipe. This can be verified by utilizing pipes with a known water pressure.
  + Notification System: Verify the notification system works by testing communication on a multitude of different devices and devices with different cell carriers. We should also test different types of emails to make sure emails work.
  + Sleep mode: Verify that putting the Arduino can be put into sleep mode remotely. Also verify that the Arduino can be woken up remotely. When the Arduino is in sleep mode it should take less power than it usually does, this can be verified by seeing how much power the Arduino consumes while awake for a duration of one hour and making sure the Arduino consumes less power while asleep for the same amount of time.

Hardware testing:

* Ensure all hardware components work properly and integrated with the software
  + Design Omni-directional antennae with US902-928 standards, then conduct a site survey to ensure the area where they are installed is not saturated with other signals operating on the same frequency. Then conduct tests for the range of the signal.
  + Backup battery: This will be done through a relay with a secondary battery to provide power redundancy and prevent single points of failure. To test this, we will run one battery low and then test the relay to make sure the battery switches over.
  + Water resistance: The case that we use for the electronics will need to be water resistant, to make sure that the case is water resistant we will test it by putting it in a shower with different settings and letting it run for approximately 30 minutes. This can be used to simulate moderate to heavy rainfall in case the building leaks.
  + Flow sensor: We need to make sure that the flow sensor accurately works, to do this we can use a known amount of water (measured out by weight beforehand) and run it through the system to calculate how much water has gone through. This can give us a close estimate to the accuracy of the flow sensor.
  + Temperature probe: The temperature probe needs to be accurate so that in case of a freeze or colder temperatures our system will correctly warn the building owner. To test the accuracy of the temperature probe we can use another known good temperature sensor and compare the values.
  + Case fitting: To make sure the case is able to fit on a variety of different pipes we will find different pipes that we can test our fitting on. Our design will need to be able to change and fit on multiple sizes to help reduce overall cost.