Requirements Document

for

Hug the Rail IoT

Version 1.1

Prepared by Tristan Kensigner, Grace Mattern, Zachary Schuh, & Christian Szablewski-Paz

Stevens Institute of Technology

March 1, 2021

Table of Contents

11. Revision History

Name(s)	Date	Reason For Changes	Version
Christian	2/15	Create document/begin section 1: Intro	1.0
Grace	2/16	Revise intro	1.0
Zach, Christian, Tristian	2/16	Organize intro into paragraphs, add notes	1.0
Grace, Christian, Zach, Tristian	2/22	Add section 2: Overview	1.0
Christian	2/23	Rewrite bullet points into paragraphs	1.0
Tristan	2/23	Add section 10: Cost	1.0
Grace	3/1	Revise/organize entire document, add graphics	1.0
Tristan	3/1	Elaborate section 2.4, add graphics	1.0
Grace and Christian	3/2	Extend section 3: Requirements	1.1
Grace	3/4	Fix typos, revise section 3: Requirements	1.1
Grace, Tristan, Christian	3/9	Update section 3: Requirements with CTO of HTR recommended features	1.1

1. Introduction

1.1 Problem

Trains are a very common and popular form of transportation that require analysis of data derived from the weather conditions, fuel consumption, and time prediction, among others, in order to ensure safe and successful travel. Current trains retrieve data from servers to receive updates that are needed to make calculations for the Locomotive Control System (LCS). One major setback to this system is that trains need to be connected to servers to keep receiving updates and consequently provide safe travel. This leaves the trains vulnerable when a cellular wifi or internet connection is lost. No new data can be retrieved without this connection and thus the safety of the passengers and operator are jeopardized.

1.2 Purpose

The Internet of things (IoT) Hug the Rail (HTR) project is a system that will allow the LCS to make decisions locally in absence of cellular and wifi connectivity to back offices. For example, it will provide the operator with information regarding if the train is slipping or if there is an obstruction. Furthermore, the operator will be able to make informed decisions regarding the problems and cautions specified by the IoT thereby maintaining safe operation and avoiding potentially dangerous situations. The overall purpose of this product would be to make HTR safer, less costly, and more efficient.

1.3 Team

Our dedicated team of developers are skilled in areas such as problem solving, programming, and project management and are excited to participate in the project. Despite having little experience with embedded systems, GitLab, and IoT systems, we are driven to design a system that allows trains to efficiently operate without a dependence on a connection. We hope to innovate and reimage the future of the locomotive industry. Our team will develop and implement IoT Hug the Rail from February 9th to April 1st of 2021.

1.4 Evolving Current Operations

Current operations on trains receive data from the departure station and cloud. During the travel, updated data is sent from servers in range and the operator utilizes the information to control the LCS. Once at the destination, the train, station, and cloud exchange data. The cloud communicates this information over the larger network. While the LCS has evolved over the years as it now can supply lots of information such as heat of the engine, fuel consumption, and generally offers a failsafe-like “software” to prevent the engine from operating outside of predefined safe limits.

1.5 Approach

The edge devices will capture data from other HTR locomotives and the environment instead of relying on data from servers. Many sensors will be implemented to provide crucial information that has now become unavailable (due to the lack of connection). Examples include the front sensors and wheel sensors which will pass information to the LCS regarding if there is an obstruction on the railroads and how fast the wheels are spinning (rpm), respectively. The wheel sensor will communicate with the still working GPS (as it provides speed) to decide if the train is slipping. Additionally, a thermometer will measure the temperature and if

weather conditions are correct, then it will inform the operator the train is slipping on ice and that sand should be put on the railroad tracks. An analytic engine on board the train will process information from the IoT software and make decisions that are passed onto the LCS. The product will provide the capability for the operators to enter in commands, receive statuses, and download the latest rules for operation from the Fog/Cloud onto the IoT software. Consequently, a finite state machine will be created, and such features will interact with the LCS controlling the entire train.

1.6 Timeline*

The timeline of this project will closely follow an agile methodology; throughout the development process changes will likely be made to both requirements and capabilities. Within each iteration of development, the team will work to recognize problems early and adapt to any changes while continuing to work efficiently. These include but are not limited to software design, technical requirements, deliverables, and the timeline itself. We find that this model helps provide a high quality project that is attuned to needs and desires of the consumers and stakeholders while also reducing the failure to analyse risks.

Week 1: 2/2 - **Communication phase** (constantly revisited, understand new desires of stakeholders)
Form teams
Decide roles and responsibilities
Week 2: 2/9 - **Planning phase** (revisited to address issues)
Understand the problem
Create documentation template
Start Section 1: Intro
Research existing systems
Week 3: 2/16 - Review/revise Section 1: Intro
Start Section 2: Overview
Research sensors that can be used to provide information to the IoT
Begin Section 10: Cost
Week 4: 2/23 - Familiarize team with GitLab/Git
Upload version 1.0 on GitLab
Research wheel sensors
Research front sensors
Research GPS interaction with wheel sensor
Research thermometer interaction with wheel sensor
Understand how sensors capture data from other HTR trains and surrounding environment
Week 5: 3/2 - **Modeling phase** (revisited if old model fails to fit new requirements)
Start Section 3: Requirements
Implement sensors that enable the IoT to make decisions
Week 6: 3/9 - Start Section 4: Use Cases
Use analytics engine to process info from the sensors to the LCS
Week 7: 3/16 - **Construction phase** (revisited when changes are made)
Start Section #
Enable operator to enter commands to the LCS
Enable operator to receive statuses when connected
Enable operator to download the latest data from Fog/Cloud onto the IoT when connected
Week 8: 3/23 - Revise/Review/Analyze
Week 9: 3/30 - Revise/Review/Analyze
Week 10: 4/1 - **Deployment**
Deliver project

*Subject to change

2. Overview

2.1 Problem Statement

Current IoT technologies on trains are focused on live data received from the Central Operation Center Servers via WiFi/Cellular networks. This heavy dependence on WiFi/Cellular connectivity introduces a massive weakness regarding safety when a stable connection is lost. Therefore it is imperative that a reliable IoT system based on local data be developed to allow for the continuation of a safe trip. For a train to operate properly external data must be provided. In the case that wifi is unavailable to provide the necessary data, a backup solution must be put in place.

2.2 Data Required to Operate

The IoT will need to collect data about the weather conditions (snow, ice, rain, wind speed) as it is essential in recognizing hazardous weather conditions. Additionally this data provides information regarding safe travel speeds and whether to put down sand or halt. Data regarding moving objects (front or back) is especially important if the obstruction is another train or moving object(s). The data will alert the operator to slow down, stop, or take another route.The train will also need information about the crossing gate's status (open or closed). If the gates are closed, the train will either need a way to open them or would need to stop. We assume that the train will still have access to the GPS/satellite. The GPS data can coordinate with the wheel sensor data which will provide data on the wheels' spin rate (rpm). In a sense, these two data will act to verify each other since when they contradict each other it should alert the operator that the train is probably slipping.

2.3 Expected Output

The data will be accessible from the IoT, from which the operator can process and make the appropriate decisions. Specifically, the IoT will interface with sensors to collect data, analyze it, and use a rule-based logic to issue recommendation or action to the operator via the IoT screen. For example, if the front sensor detects an object is too close (will be defined later), it then will detect if it is moving. This information will be available on the IoT and provide a hazard warning to the operator, who can then stop, slow down, etc. the train. Another example is the wheel sensor which will verify its data with the still working GPS. The GPS provides the speed the train is going while the wheel sensor will provide how fast the wheels are rotating (rpm) and thus if one of the data fails to corroborate the other, then a warning will appear on the IoT. The thermometer will provide additional information concerning what the likely cause is (i.e. ice, snow, rain, etc.). With both the knowledge that the train is slipping and the likely cause, say ice, the operator can slow down or pour sand on the railroad tracks. This same system of sensors can also notify the operator when slipping has stopped.

2.4 Available Technologies

There are a multitude of different sensors available. Thermometers, hygrometers, friction/shear force sensors, anemometer, and tachometers are the main sensors we intend to use to pass weather related information to the LCS and consequently to the operator. A hygrometer measures the humidity, an anemometer measures wind speeds, and a tachometer measures the rpm of a wheel. Another important sensor is a radar/sonar/ultrasonic-like device that can map out the environment surrounding the train for the purpose of warning/alerting the operator of obstructions. Popular technologies include LED time of flight sensors and LiDAR sensors, the latter of which are not uncommon on cars and airplanes. A LiDAR sensor continually

fires off beams of laser light and then measures how long it takes for the light to return to the sensor. In effect, a LiDAR sensor returns the distance of objects hit by the laser beam and thereby creating a 3D visualization of the environment. With continuous firing, it can also present the movement of objects surrounding the train just as it would notify a driver that a pedestrian, curb, or other vehicle is near. The GPS is a versatile technology that provides information about the position of the train. Not only is it highly reliable, but it also integrates well with the previous technologies. When used in conjunction with weather based sensors, it can help authenticate that a train is slipping. Additionally when used in conjunction with obstruction detection sensors, it can provide the appropriate speed to slow down the train.

The sensors for the IoT Hug the Rail Project would be mainly located in three spots. The top of the train will house most of the weather related sensors such as the anemometer, thermometer, and hygrometer. The tachometer and friction sensors will be located near the wheels of the train as these sensors require information from the wheel's rotation and friction. This data will provide the necessary information to the IoT about the train's speed, change of velocity, and friction force. The front and back of the train will house sensors concerning 3D mapping for obstacle detection. Our IoT system will still have access to the GPS as it does not require a WiFi/cellular connection. The status (open/closed) of cross gates will be handled locally by solid state circuits and pressure sensors on the railroad tracks. **Figure 1** provides a general layout of the sensors on HTR trains.

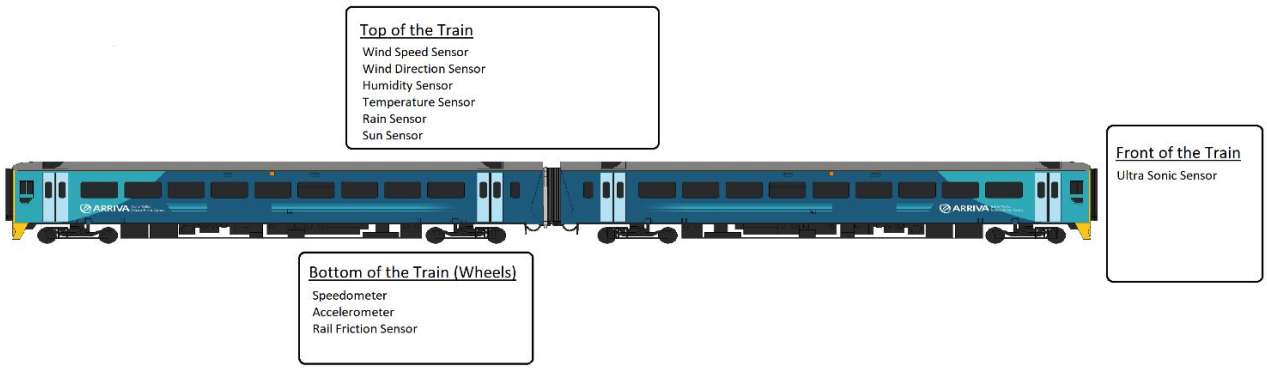

Figure 1

2.5 Systems

The IoT HTR system will collect data retrieved from the various sensors and interpret them accordingly. Results (i.e. weather, speed, obstructions, etc.) will be available via the IoT and suggest operational changes to the operator based on its detections. Below are some conceptual architectures for our IoT. **Figure 2** and **Figure 3** communicate the general problem and solutions. **Figure 4** shows how the IoT systems would be used specifically regarding weather based slipping.

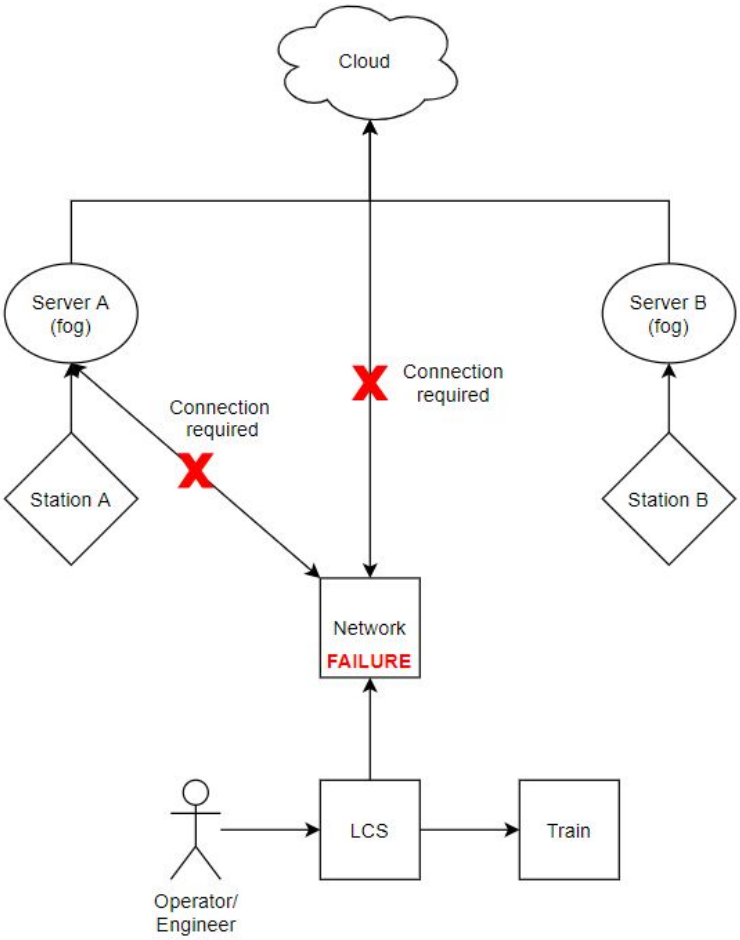

Figure 2: *shows the problems that occur if a connection is lost (represented by the red x's). Supposing the train starts in the range of station A, the LCS is connected to the network which receives data from server A and also the cloud. However when the connection is lost no new data is retrieved from server A or the cloud and the network fails.*

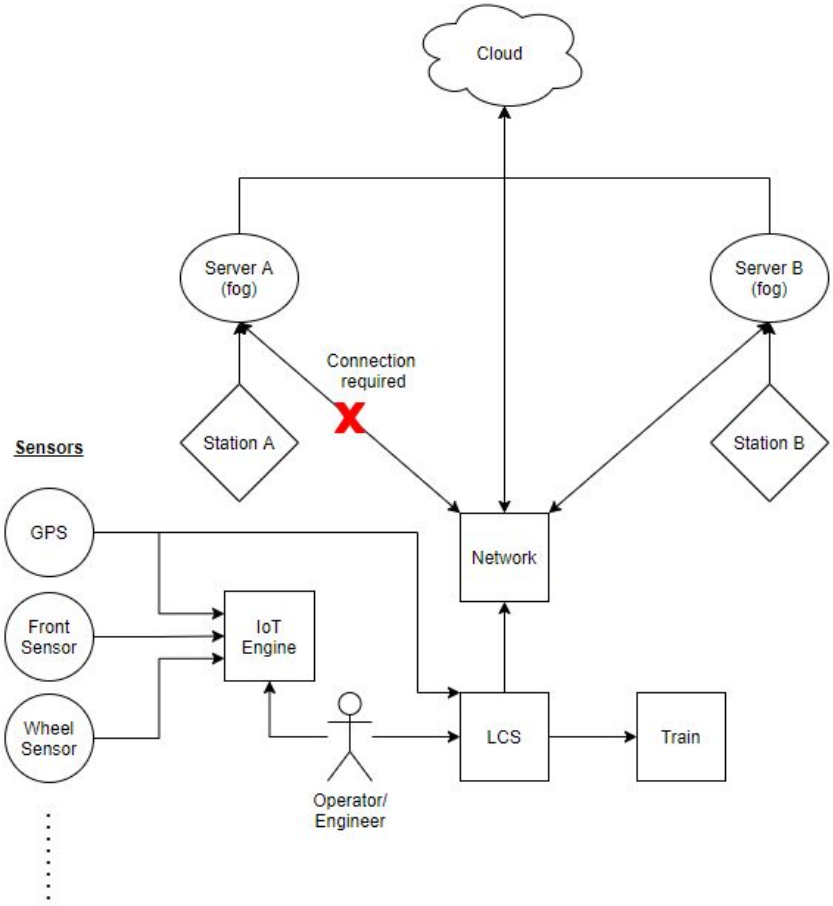

Figure 3: *shows the general solution IoT Hug the Rail Project will provide. Supposing the train is headed to station B without a connection, information from the GPS, front sensor, wheel sensor, and other sensors are processed by the IoT Engine which are read by the operator who can then make the appropriate decisions. Once at station B, a connection is reestablished. The train, server B, and the cloud share data.*

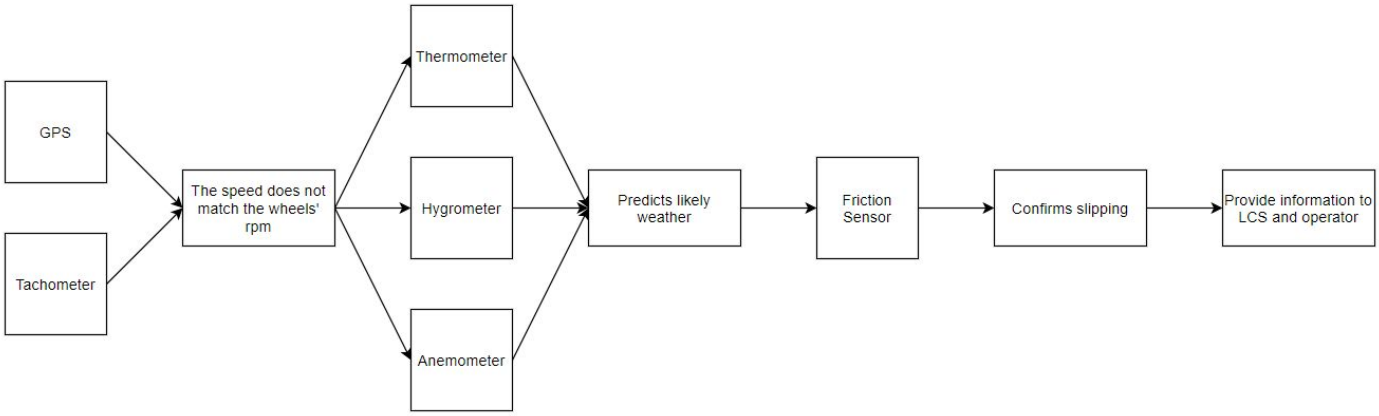

Figure 4: *Weather sensors can provide information about ice, snow, rain, slipping, storms, winds, etc. For example the GPS could report the speed of the train but the tachometer would contradict the data by showing that the wheels are rotating faster than the given speed. The thermometer, hygrometer, and anemometer could predict the weather conditions (i.e. snowing, raining, storming, etc.) and the friction/shear force sensor could verify that the train is slipping. The following data will be available to the operator on the IoT.*

3. Requirements

3.1 Non-Functional Requirements

3.1.1 Security

R-1: IoT HTR shall only be accessed via user ID and password
R-2: IoT HTR Admin shall have secured (Admin ID/Pwd) to all sensors and equipment
R-3: The only thing operators will see on the screen is the operation of the train
R-4: Only admins have access to the raw data, logs, and whole system
R-5: IoT HTR Network shall be secured by a LoRaWan Protocol
R-6: IoT HTR will endode all user-supplied output
R-7: IoT HTR will lock the system after 5 failed attempts to login
R-8: Only operators and admins have access to capabilities listed in 3.2

3.1.2 Performance

R-9: IoT HTR shall process an event within 0.5 seconds
R-10: IoT HTR shall process 1000 events per second without degrading services
R-11: IoT HTR shall alert the operator of objects on the track within a 2 mile radius
R-12: IoT HTR shall alert the operator if an obstruction is a moving object within 1 second
R-13: IoT HTR shall display processed events within 0.1 seconds
R-14: IoT HTR shall display updated data within 0.1 second of retrieval

3.1.3 Reliability

R-15: IoT HTR shall operate with no failures 99.99% of the time
R-16: IoT HTR shall alert the operator how fast obstructions are moving with 95% accuracy
R-17: The precision of distance-based calculations shall be at least 0.000001
R-18: IoT HTR defect rate shall be less than 1 failure per 1000 hours of standard operation
R-19: No more than 1 per 1000000 user interactions shall result in a failure requiring the system to restart
R-20: IoT HTR shall meet or exceed 99.9% uptime
R-21: If the engine fails, it will be identified in 0.01 seconds
R-22: If the engine fails, it will be resolved by an admin

3.2 Functional Requirements

3.2.1 Standing Objects

R-23: Proximity based sensor shall send data to the IoT every 0.05 seconds
R-24: Proximity based sensor shall send its status to the IoT every 0.05 seconds
R-25: IoT will process proximity sensor data
R-26: IoT will analyze for obstructions in a 2 mile radius
R-27: IoT will detect standing objects on the path of the train
R-28: IoT will display the distance of standing objects

R-29: Will display a warning message if obstruction is detected
R-30: IoT will display the speed the train is currently moving
R-31: IoT will suggest to the operator to break/slow down/speed up

3.2.2 Moving Objects

R-32: Proximity based sensor shall send data to the IoT every 0.05 seconds
R-33: Proximity based sensor shall send its status to the IoT every 0.05 seconds
R-34: IoT will process proximity sensor data
R-35: IoT will analyze for obstructions
R-36: IoT will detect moving objects on the path of the train
R-37: IoT will display the distance of moving objects
R-38: IoT will display if the moving object is behind or in front of the train
R-39: IoT will detect the speed of the moving object in a 2 mile radius
R-40: Will display a warning message if obstruction is detected
R-41: IoT will display the speed the train is currently moving
R-42: IoT will suggest to the operator to break/slow down/speed up

3.2.3 Gate Crossings

R-43: IoT will detect gate crossing's status (open/close)
R-44: IoT will display the distance of the gate crossing
R-45: IoT will suggest to the operator to break/slow down/speed up

3.2.4 Wheel Slippage

R-46: IoT will have four sensors on four wheels
R-47: Each sensors will pass the rpm of the wheel it is measuring to IoT
R-48: GPS shall send data to the IoT every 0.05 seconds
R-49: GPS shall send its status to the IoT every 0.05 seconds
R-50: IoT will process GPS data
R-51: IoT will analyze for speed
R-52: IoT will display GPS data
R-53: Will display a warning message if the train is going to fast/slow
R-54: Will display a warning message if the train is not properly braking
R-55: Will suggest to break/slow down/speed up
R-56: IoT verifies slippage using information about the wheels' rpm and the speed via the GPS
R-57: After braking/speed decrease, IoT will check if train is actually slowing down

3.2.5 Other Functional Requirements

R-58: Operators and admins will be able to start, stop, and pause the IoT engine
R-59: Only admins will be able to test IoT
R-60: Only admins will be able to restart IoT
R-61: The train shall respond to commands given by either the IoT engine or the operator

R-62: Weather based sensor shall send data to the IoT every 0.05 seconds
R-63: Weather based sensor shall send its status to the IoT every 0.05 seconds
R-64: IoT will process weather sensor data
R-65: IoT will analyze for weather conditions
R-66: IoT will display weather sensor data
R-67: Will display a warning message if there are hazardous weather conditions
R-68: Will suggest to break/slow down/speed up
R-69: Will suggest to put sand on the rails
R-70: If no issues are presented by all the sensors, then a continue normal operation message will be displayed

3.3 Hardware & Operating System

R-71: IoT hardware shall be able to support 1,000 sensors
R-72: IoT shall support 5TB of data everyday
R-73: IoT HTR shall be able to import and export information obtained from operating
R-74: IoT HTR will be updated yearly with patches in between

3.3.1 Operating System

R-75: Microsoft Windows 10 IoT Enterprise
R-76: Microsoft Windows 10, 64-bit

3.3.2 Processor

R-77 ARM processor

3.3.3 RAM

R-78: 2GB

3.3.4 Graphics Card

R-79: GPU that supports DirectX 9

3.3.5 Storage

R-80: 32GB of storage

3.3.6 Sensors

R-81: Hygrometer - measures humidity
R-82: Tachometer - measures the rpm of a wheel
R-83: Thermometer - measures temperature
R-84: Anemometer - measures wind speed
R-85: LiDAR - maps surrounding environment
R-86: Cameras - for gate status and LiDAR

R-87: GPS - give global positioning

4. Use Cases

5. [TBD]

6. [TBD]

7. [TBD]

8. [TBD]

9. [TBD]

10. Cost

Costs for any project can be daunting and often difficult to calculate because the project expands and evolves over time. We can give a rough cost estimation of this project based on a sample of devices one would buy. These costs are subject to change and there are multiple companies with varying prices that sell these sensors. It also should be noted that a large portion of the project will be software which does not have any external costs.

Sensor	**Purpose**	**Unit Price**
Anemometer	Wind speed	$50-$800
Wind Vane	Wind direction	$400
Hygrometer	Humidity	$200
Rain Gauge	Rain	$300
Thermometer	Temperature	$200
Tachometer	Wheel's RPM	$50-$900
LiDAR	3D mapping	$500-$1,000
Friction/Shear Force	Friction	$6,000-$8,000
System of solid state circuits and pressure sensors	Gate	$1000+
IoT screen and software	UI/UX	
GPS	Position/speed	$100-$1000
Assemble		