CSCE 4925: Project Aero
Detailed Design

By: Alyssa Thurston, Breuna Riggins, James Sabetti, Travis Goral
Project Manager: Alyssa Thurston
Instructor: Professor Keathly
Client: Dan Minshew and Kyle Tayle, Denton Techmill
Date: 1 December 2017
Status: Final Draft

Table of Contents

Revisions

Below is a list of any revisions made to this document.

Date	Description of Change Made	Person Making Change
11/7/2018	Draft Created	Breuna, Alyssa, James. Travis
11/23/2018	Draft Compiled	Breuna
11/24/2018	Document Edited	James
12/1/2018	Final Draft Complete	Breuna, Alyssa, James, Travis
2/13/2018	Final Draft Revisions- Reformatting, Editing of the functions	Alyssa
5/1/2018	Final Draft Revisions	Alyssa, Travis

1 Project Summary

The air quality in the City of Denton is notoriously bad. It's not a surprising fact given that there are two universities and multiple major highways and roadways that run through the city's limits. Worse still, there is only a single air quality sensor within city limits, and the next closest one is almost 15 miles away from the city.

The goal of this project is to create a network of air quality sensors to measure the air quality in the city of Denton so that citizens and city personnel alike can monitor air quality and take appropriate actions. The data obtained from the sensors is to be made public, so that air quality trends can be tracked all around the city. With these requirements met, the network would provide secure, real-time, and reliable data on the air quality in the city of Denton.

With the aim of creating a robust and open source of air quality data for the city of Denton, the Open Denton group has set out to establish a network of air quality sensors to gather and compile that data. In order to help achieve this goal, our group has been tasked with the following:

- Create a web client to display data in a highly readable manner.
- Create an API that records and displays air quality data in a readable manner
- Implement a scalable database to hold and model recorded data
- Integrate sensors and controllers that will read and relay data to the database over the internet
- Create documentation that allows community members to add sensors to the network

This document will serve as the guiding documentation for the implementation of this project.

2 Overall Description

For our product, our goal is to design a web based client, a database, and an API to get the data from one to the other. The web based client should be accessible, up-to-date and easy to use. The database should contain as few errors as possible, should be scalable, and presented in a time-series format. End users should be able to access the website at anytime from as many device types as possible. Equally important, if a user wants to contribute a node, this needs to be made as easily as possible as well.

2.1 Functions

The web client will perform the following functions:

- 2.1.0: Display air quality statistics in an easy to read, graphical format
- 2.1.1: Provide users with a map of all sensors within city limits
- 2.1.2: Provide a method of looking up historical data
- 2.1.3: Show what sensors are online and offline
- 2.1.4: If a sensor is offline, show when it was last online or last “seen”
- 2.1.5: Display data for a single sensor
- 2.1.6: Provide a help menu for users needing help with the system
- 2.1.7: Allow a user to login to an account
- 2.1.8: Allow a user to create an account
- 2.1.9: When logged in, show a map of where the user’s sensor’s are located
- 2.1.10: When logged in, show sensor information
- 2.1.11: When logged in, show statistics from the user’s sensor’s
- 2.1.12: Provide location data of a sensor, either via address or through coordinates.
- 2.1.13: Verify that users are able to create an account.
- 2.1.14: The map should use different icons for sensors that are online and offline
- 2.1.15: Users should be able to add sensors into the network
- 2.1.16: Users should be able to delete their sensors from the network

The database will perform the following functions and have the following qualities:

- 2.1.17: Store data from the sensor(s)
- 2.1.18: Accept user reported data
- 2.1.19: The system should accept data from any sensor, regardless of type.
- 2.1.20: The database should be presented in a time series format

Some other requirements of this project include:

- 2.1.21: It should be able to store any air quality value that a user is able to provide with their sensor
- 2.1.22: Verify that the data reported from the sensor is not a outlier, and somewhat matched the data reported from other sensors.
- 2.1.23: A large volume of users should be able to use this system without issue.
- 2.1.24: The system should be secure.
- 2.1.25: Raw data should also be made available for download

2.1.26: The system should refresh itself with new data pulled from the sensors every five minutes.
2.1.27: System should notify a user if the data that is being pulled is an outlier

2.2 Use Cases

This system is designed to be used to report the level of air pollution and the quality of the air dependent on human pollution, the environment, and weather circumstances, to be recorded and reported by both ourselves and other users.

2.3 Operating Environment

The operating environment for this project is presumably a server that houses all of the database information, that is then recorded and fed to a dashboard for users to read and interpret the statistics presented by the users.

2.4 User Documentation

Users will be primarily using this system in one of two ways: they will either be accessing the information via the dashboard (at which point, there would be no need for documentation), or they will be inputting the information recorded by their devices, at which point the system would need to document what the users have input into the system and verify that information to make sure that it is correct and accurate. All user inputs would need to be documented in order to keep the legitimacy of our statistics reported.

2.5 Assumptions and Dependencies

We are assuming, while building these requirements, than the information reported by the users would be reported by the device they have made and own, not information submitted by the user's hand. We are also assuming that our database can correctly store all of this information separate from the dashboard, and that the dashboard merely reads out the information to the user.

2.6 Project Constraints

We are constrained in terms of time and the resources given to us. We will have no control over the machine that will be given to us to record all of this information with, and we are constrained by a year time limit in order to accomplish everything we have set out to do. As a result, while this project should not push the year long time limit, it may lack the polish and finish that a fully realized system with a longer timeline would normally have.

3 External Interfaces

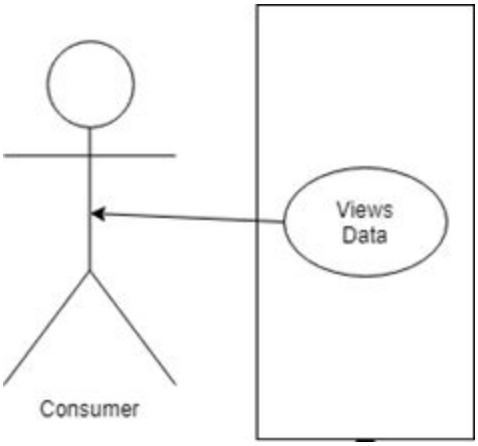
3.1 User

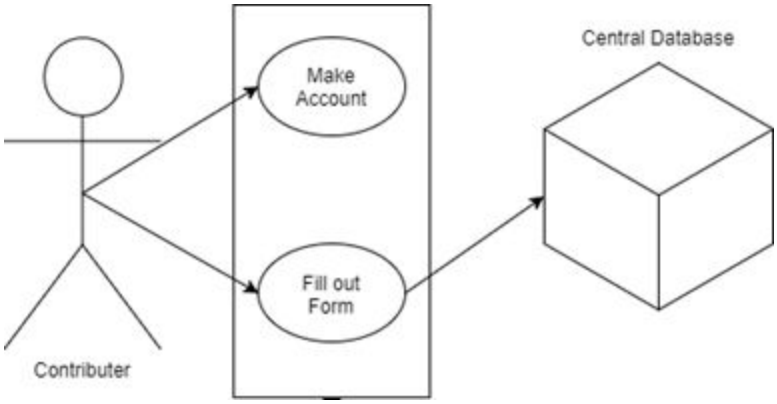
The user interface will be a responsive web based design, so that the project can be accessible from virtually any device with an internet connection. The readings should be shown graphically, using graphs, meters, and more, so users know what the levels are, if they're unhealthy or not, and can view trends over time.

As appropriate, charts from the Data Viz Project will be selected to display the data in a way that is friendly to users. Some of these may include an angular gauge, a pie chart, line graphs, grouped bar graphs, and more.

The user interface will be browser based, and should not be restricted to specific browsers or clients. It should adapt to fit both a desktop computer screen and a mobile screen. It should be available to use on all web browsers, such as Microsoft Edge, Mozilla Firefox, Google Chrome, and Mac's Safari.

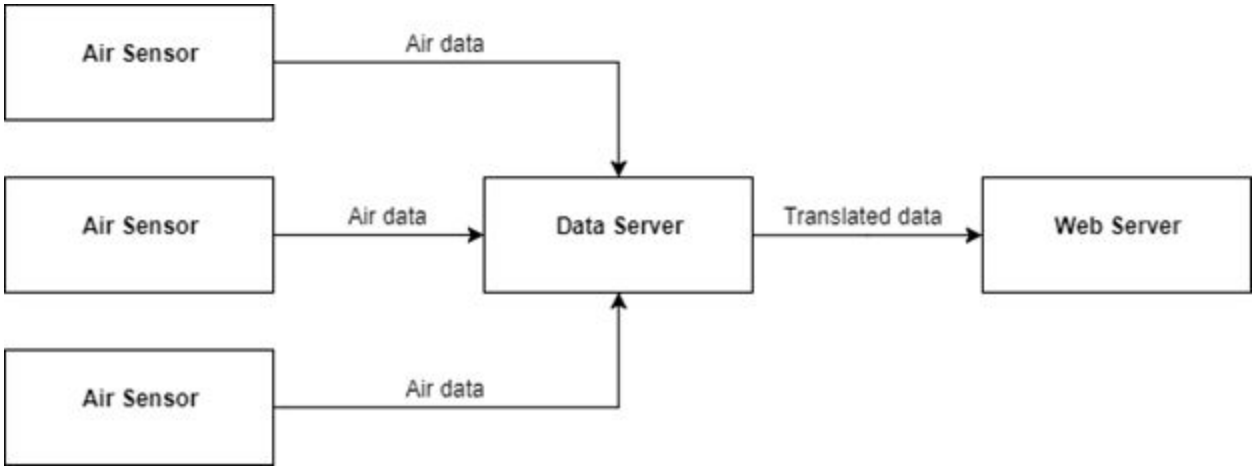
Two different user types were defined in meetings with the client- a consumer and contributor. The consumer simply views data on the UI, while the contributor will be able to create an account and fill out a form in order to contribute a sensor to the network. These functions are shown in the following diagrams.





3.2 Hardware

The project should be able to load data into a database from the air quality sensors, then take the data from the database and represent it in a human readable format. A diagram showing these details is shown on the next page.



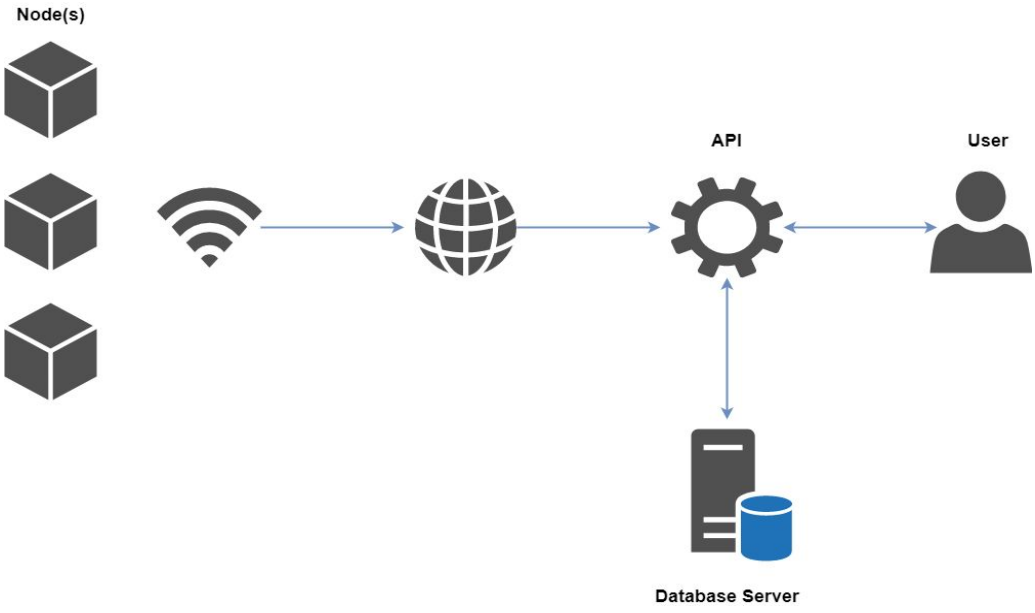
3.3 Software

This product will have several interactions between components. The sensors will send data, and the data will be populated into a database. A middle agent will then take the data and analyze it and make it more user friendly, and then the user interface will then take the prepared data and display it. A diagram is shown of the way this system will work in Section 6.0—System Features.

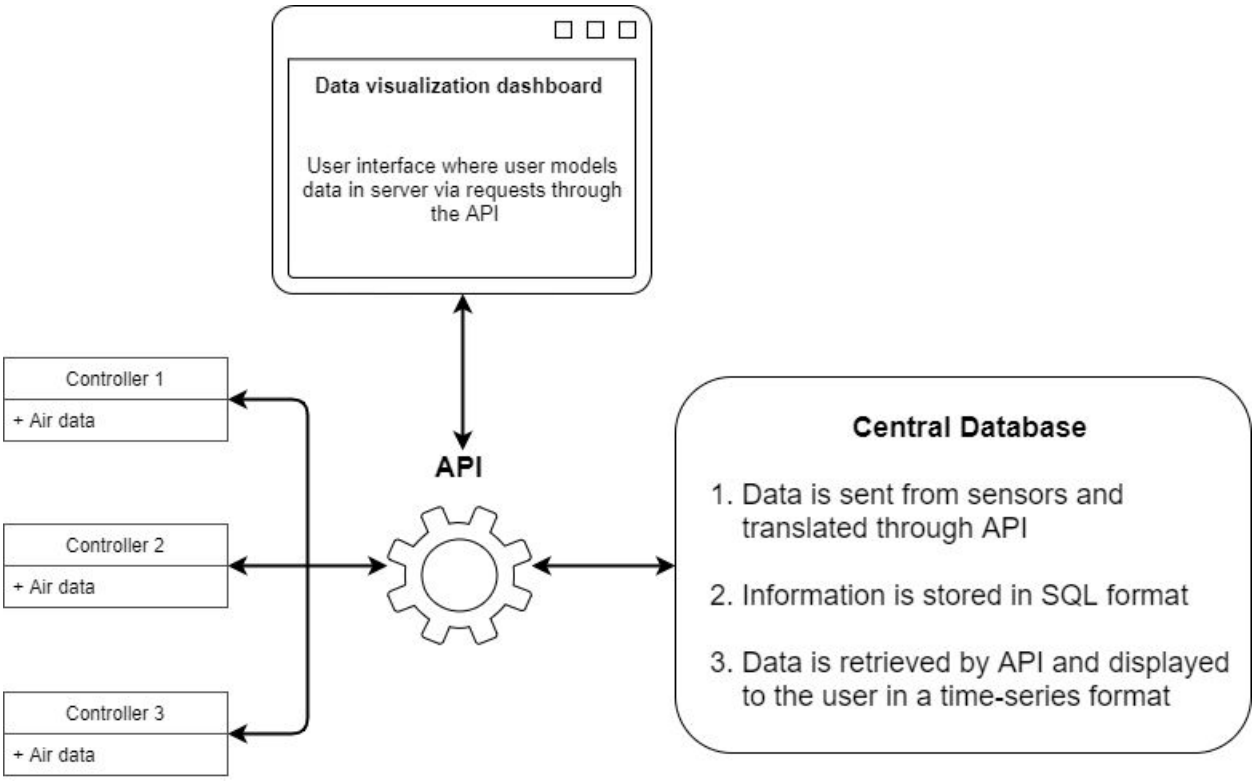
Messages will mainly be flowing out of the database into the UI, but some will be taken in the opposite direction, specifically when a citizen wants to connect a sensor to the network. This case is shown in section 2.1—User.

3.4 Communications

The main form of communication in this system is done over the internet. The sensors will be connected to the internet either via WiFi or a cell connection which will allow the data to be read into the database. A diagram of this model is shown below.



The system shall be comprised of several distinct functional requirements that outline the basic roles required for desired operation. They are detailed in a high-level orientation in the diagram below:



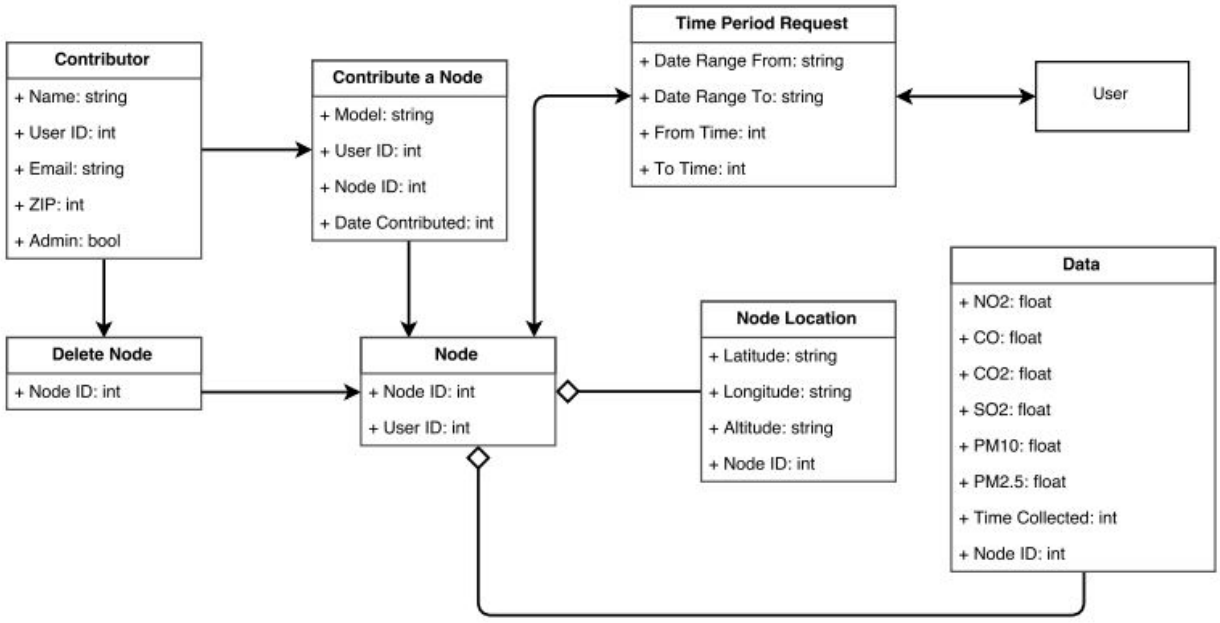
In order to keep the scope manageable at this stage, the overall vision of the system has been condensed into three primary functional requirements:

- A central time-series database
- A Middle API that translates data
- Data visualization dashboard

Later on, requirements will be broken down from these main three as needed. Additional requirements will be detailed as they become apparent.

4 Class Diagrams

The following depicts the different data that will be stored in the system and what the database might look like.

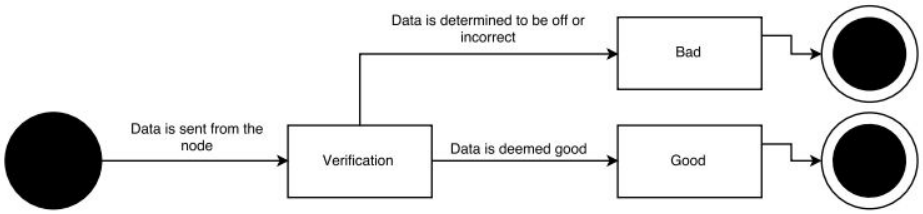


5 State Diagrams

The following diagrams depict items that were considered to be stateful. They include a calibration flag, a manager/admin flag, collecting data, and adding a node.

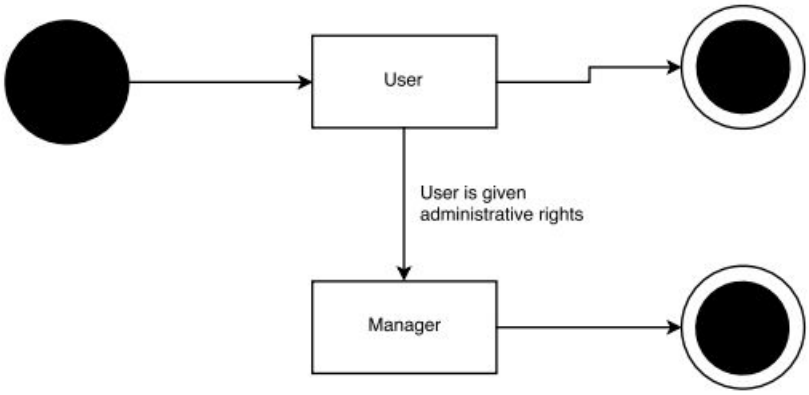
5.1 Calibration Flag

When a node provides consistent data that is shown to be significantly greater than or less than the values provided by other nearby nodes, it can be assumed that the node needs to be calibrated.



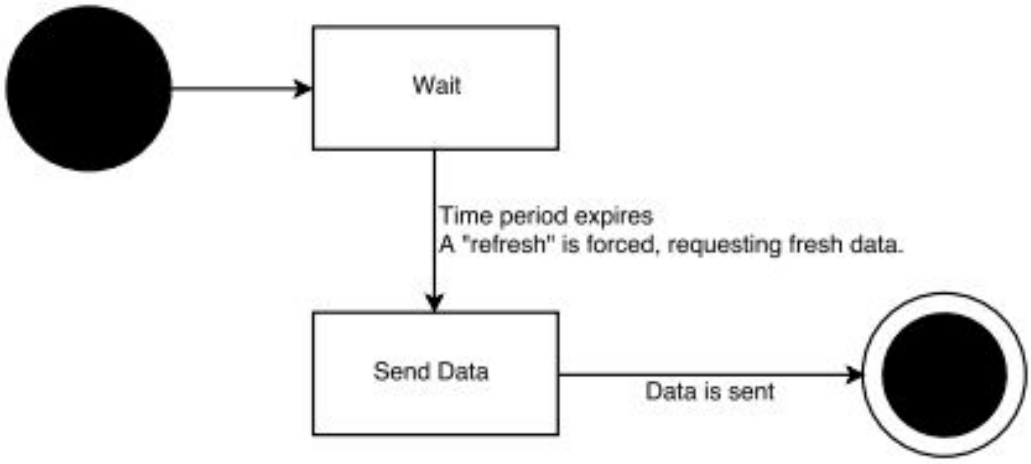
5.2 Manager/Admin Flag

Some users are considered managers or administrators of the system. It needs to be known that they have this superiority, and thus a flag is given.



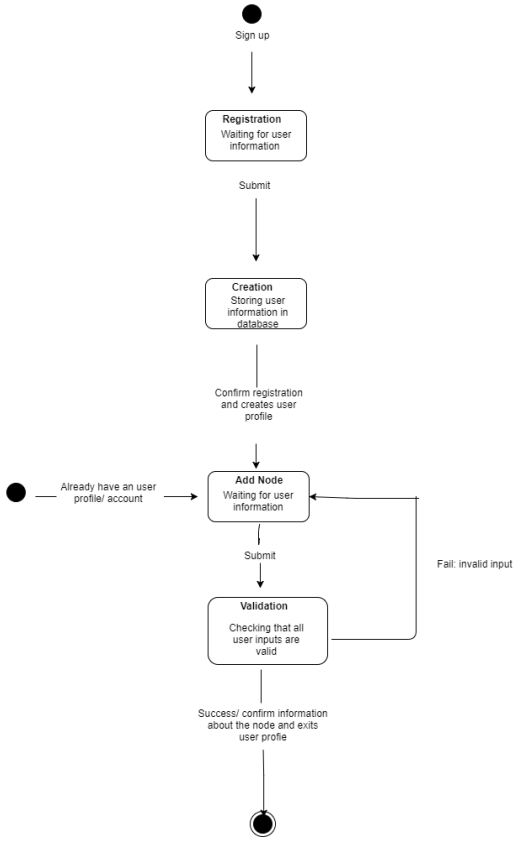
5.3 Collect Data

When the node collects data, this is stateful because the node only needs to sample the data and send that data to the server every five minutes. A diagram is shown below.



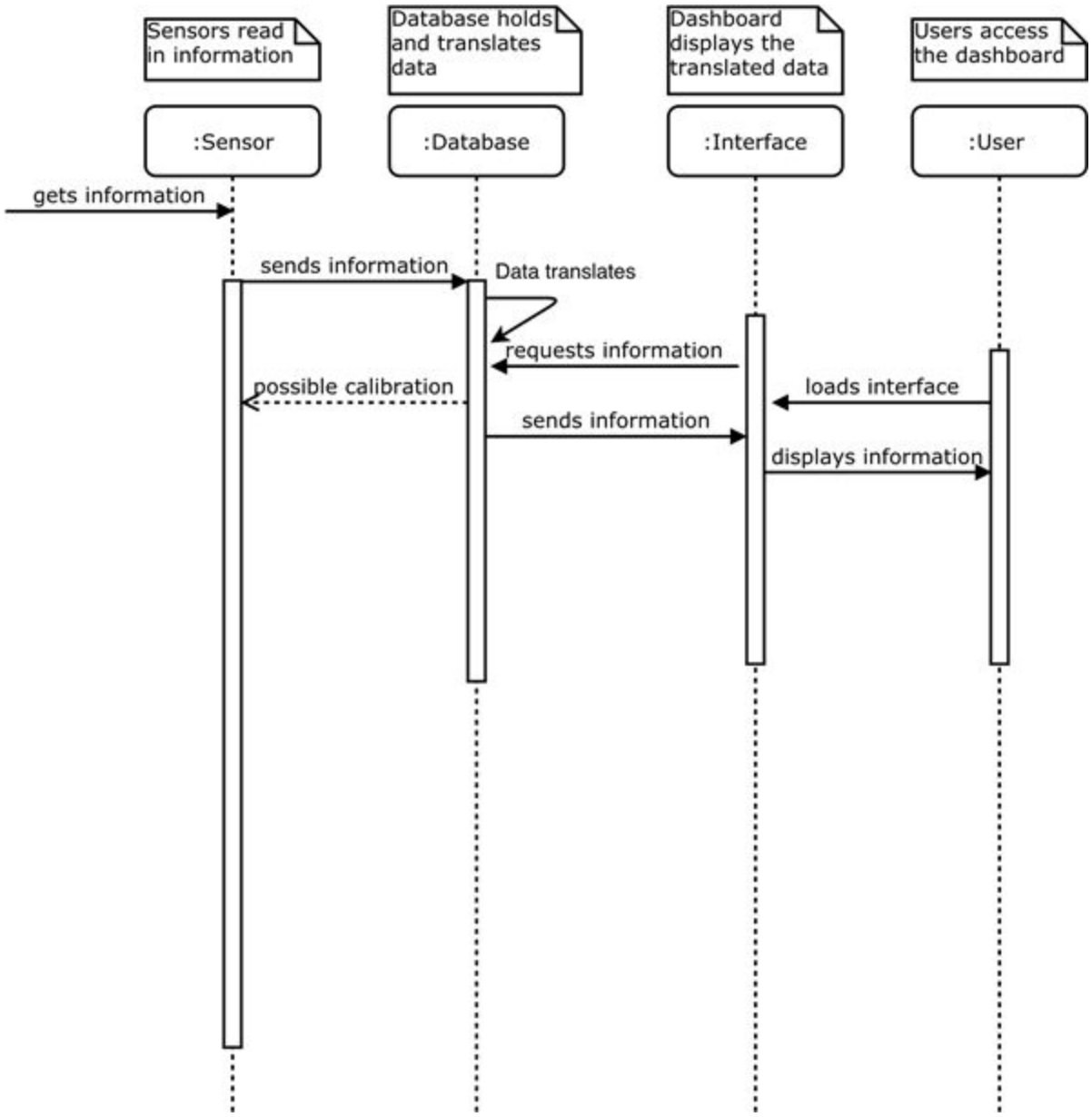
5.4 Adding a Node

The following is the process that a user goes through to add a node, and the states that are encountered.



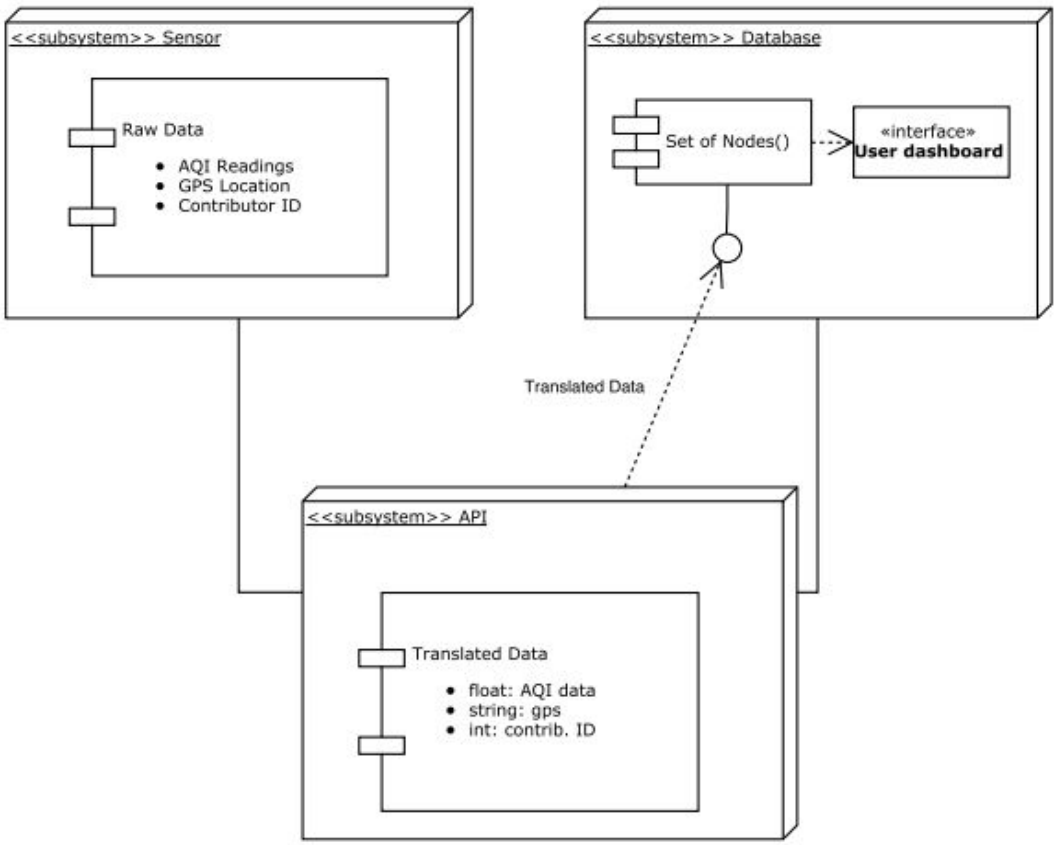
6 Interaction Diagram

This depicts the different interactions that exist between the different components of the system.



7 Component Diagram

The following depicts the location of different system components.



8 Package Diagrams

The following diagram depicts where each component “lives” on a system.