CSCE 315 Project 1 Handoff Documentation

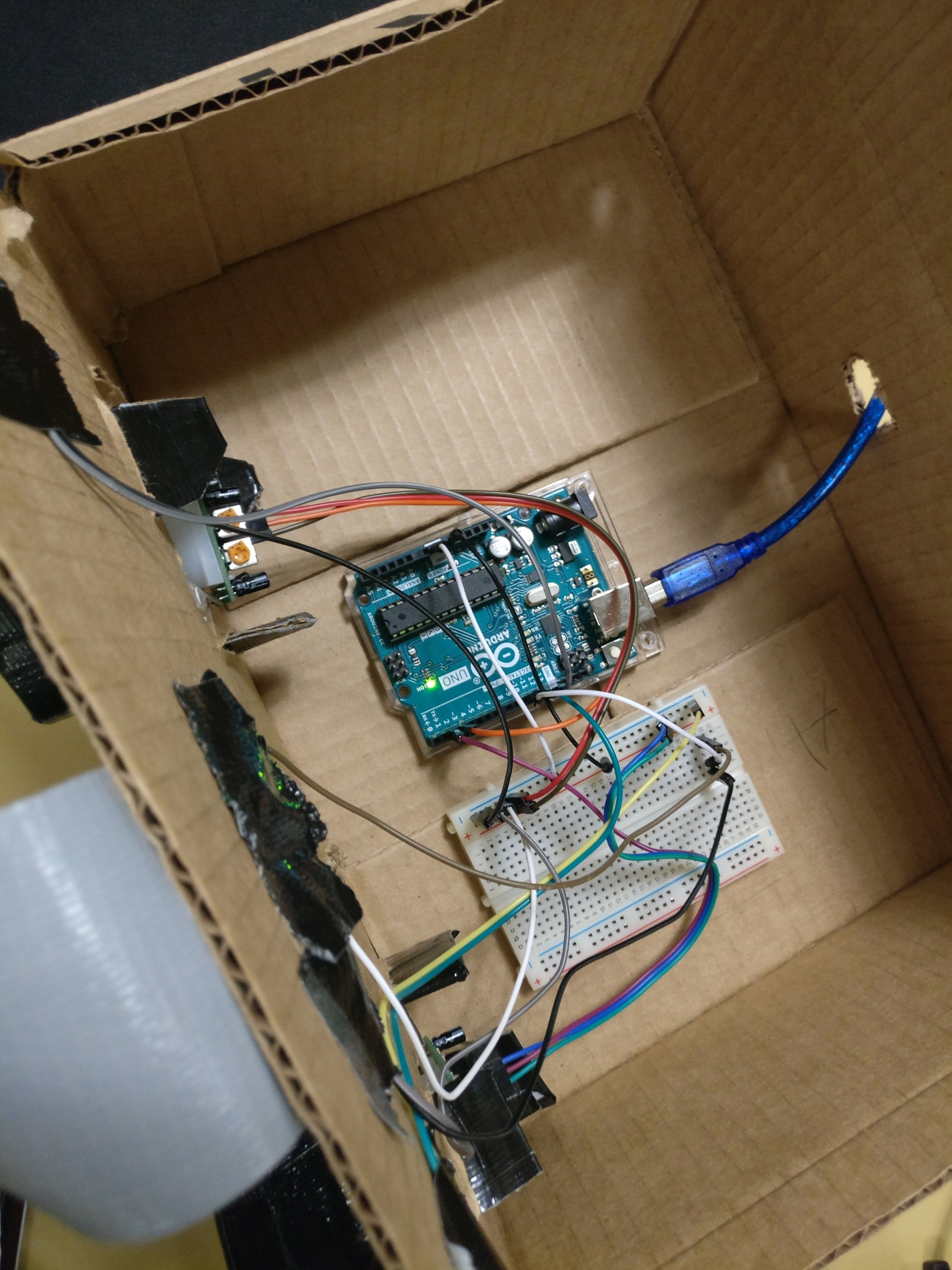
**Overview:** This project is a traffic counting device that detects foot traffic through an area and then sends the data to a database. The data is then retrieved from the database and displayed through a webpage.

**Part 1: The Arduino and Python**

**The Arduino Uno** is the core of the project; essentially it is a microcontroller that is easily programmable and can use a wide variety of sensors, motors, and more. Usually, the Uno is programmed via the Arduino IDE which makes it easy to program the Uno for any task the user desires. For this project though we could not use the Arduino IDE and we instead had to program the Uno with Pyfirmata.

**Pyfirmata** is a python interface of Firmata – which is a protocol for communicating with microcontrollers. The downside to using this is that there isn’t much code existing for Python Arduino projects and many sensors don’t have a library to easily be used. Pyfirmata’s documentation can be found online.

**PIR Motion Sensors** were the type of sensors that we used for this project. We used 2 sensors in order to tell if someone was coming in or coming out. When someone was coming from the “exit” direction we would not count them, but when they came from the “enter” direction we would count them and then send the data to the database. The sensors were positioned a few inches apart and had dividers in-between them so they wouldn’t count people coming from the other direction.



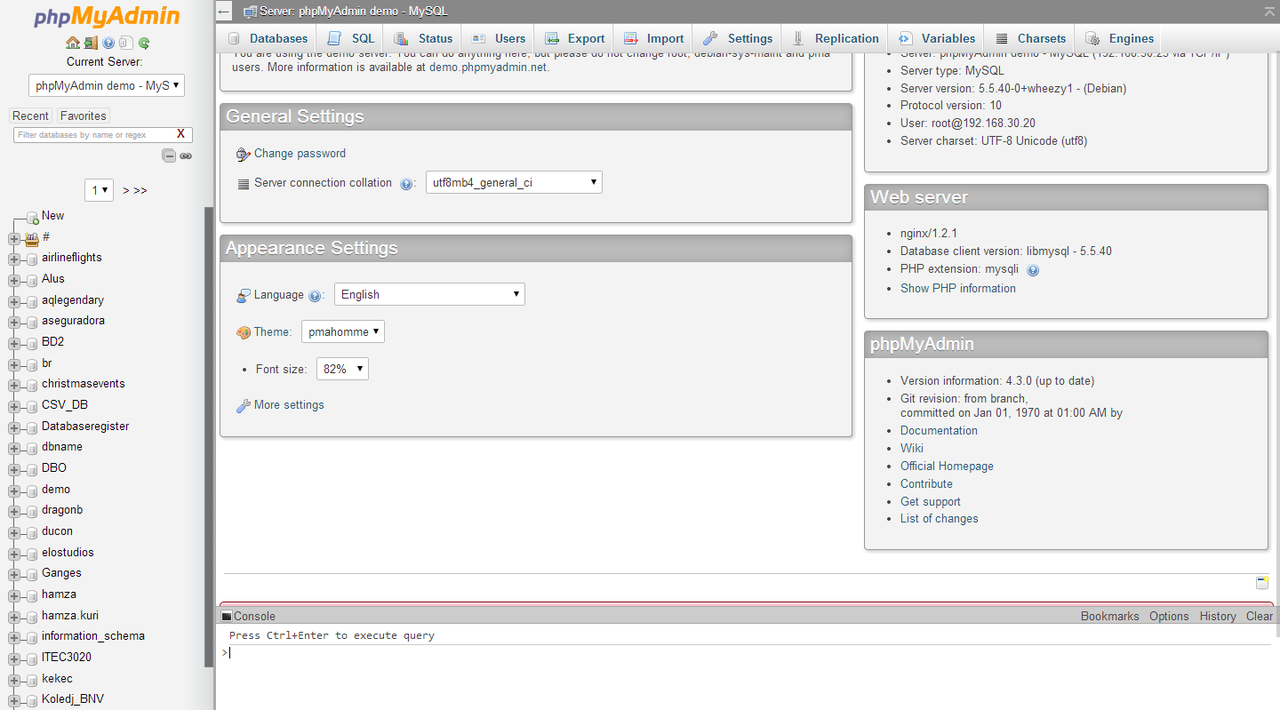
**Setup:** In this project the Uno is tethered to a computer (A laptop in our case) and lies in a box with the sensors. A complete photo of the project is found below. We also included 4 led lights in order to know when the sensors are done resetting.

**Part 2 The Database**

The database we used was phpMyAdmin which uses SQL to manage tables and such. In order to send and receive data from the phpMyAdmin you first need to create your database which can be done by claiming it on the CSE website. After you’ve claimed your database you can then create new tables and inserted or remove data using SQL.

**Sending Data:** Sending data requires a connection to the database followed with an SQL query of insert. In our project we would simply send a “1” every time someone tripped the “enter” sensor. The database would automatically create a timestamp which is useful for receiving data.

**Receiving Data:** This is trickier than sending data because it requires a select query between 2 dates. We accomplished this issue by using the SQL BETWEEN operator – which let us select data that was between 2 dates.



**Part 3 The Webpage**

Our webpage used a combination of PHP, html, and JavaScript in order to select and display data. All of the Webpage files have to be in the user’s projects folder otherwise the webpage will not be able to use PHP.

PHP: PHP is the backend of the webpage. PHP was used to select data from the database and to perform other functions. We had a different file for each option – one for displaying the last week, one for the last day and so on. Once the file gets the data from the database using the SELECT query, it can then pass on the data in order to display it.

JavaScript: We used JavaScript for the visual part of the webpage – after getting the data from the PHP file we can then create a chart. We used Charts.js which is an open source JavaScript charting tool that lets us display our data. In order to get the data from PHP to JavaScript we had to use Ajax. The result is a chart that can display our data.