Who:

Bader Albader

Jacob Tran

Tyler Valentine

Garrett Senor

Chuck Mezhir

Title:

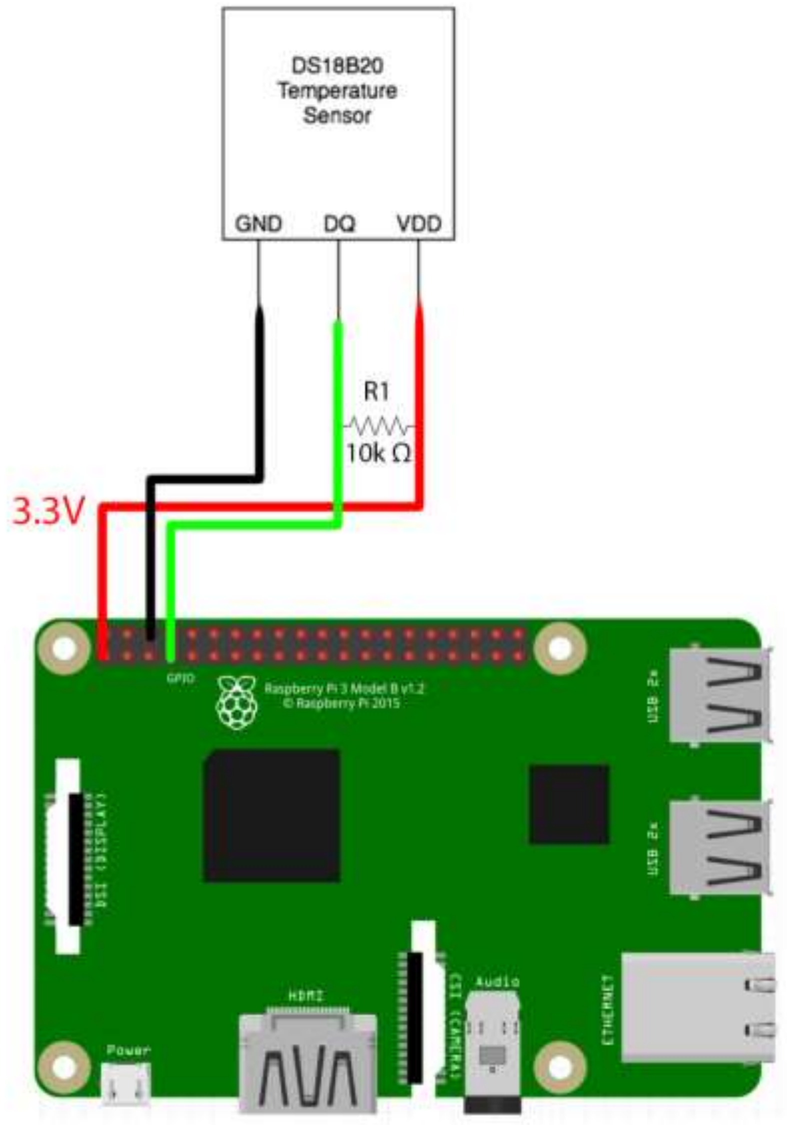
Kong

Automated Tests:

By Bader and Jacob

Hardware Test:
Description: This test is to ensure that the sensors being used are working effectively. The sensors used are a temperature sensor and a humidity sensor.
Type: White box test.
Setup:
1. A Raspberry Pi 3 was used as the micro controller. The temperature sensor was connected to a 10K Ohm pull down resistor in order to read data. The sensor is also connected to 3.3V on the Pi and ground on the Pi.
2. A Python script was written that reads the temperature and humidity from GPIO pins on the Pi. Once the temperature is read is it fed to the backend database.

Results:
1. Temperature, humidity, data and time is printed out when the script is run.
2. The same data points are then updated on the database.



Sensor Test:

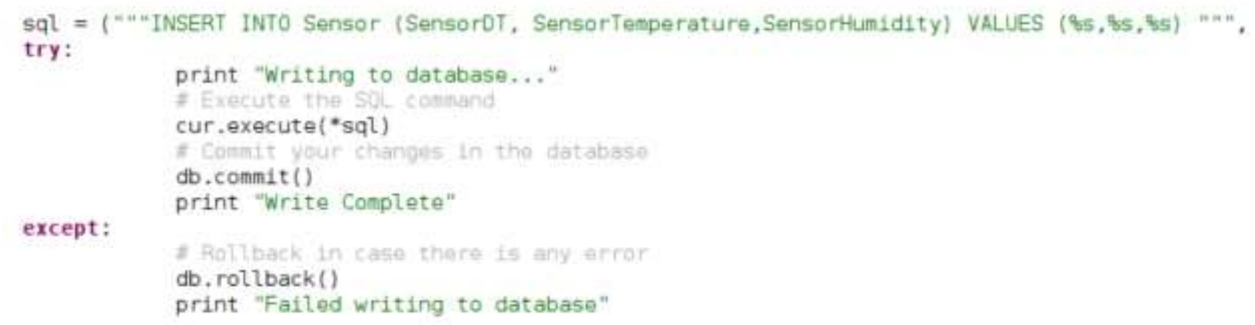

```
sql = ("""INSERT INTO Sensor (SensorDT, SensorTemperature,SensorHumidity) VALUES (%s,%s,%s) """,
try:
            print "Writing to database..."
            # Execute the SQL command
            cur.execute(*sql)
            # Commit your changes in the database
            db.commit()
            print "Write Complete"
except:
            # Rollback in case there is any error
            db.rollback()
            print "Failed writing to database"
```

Figure 1. PyUnit Test

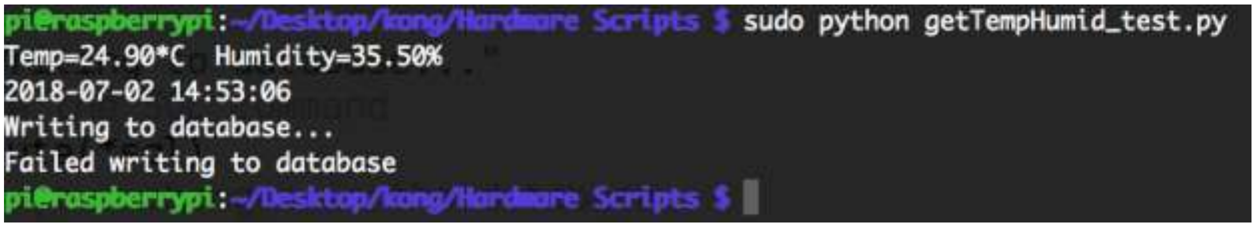

```
pi@raspberrypi:~/Desktop/kong/Hardware Scripts $ sudo python getTempHumid_test.py
Temp=24.90*C  Humidity=35.50%
2018-07-02 14:53:06
Writing to database...
Failed writing to database
pi@raspberrypi:~/Desktop/kong/Hardware Scripts $
```

Figure 2. Failed to connect database

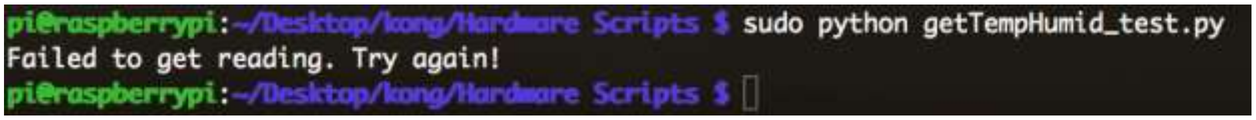
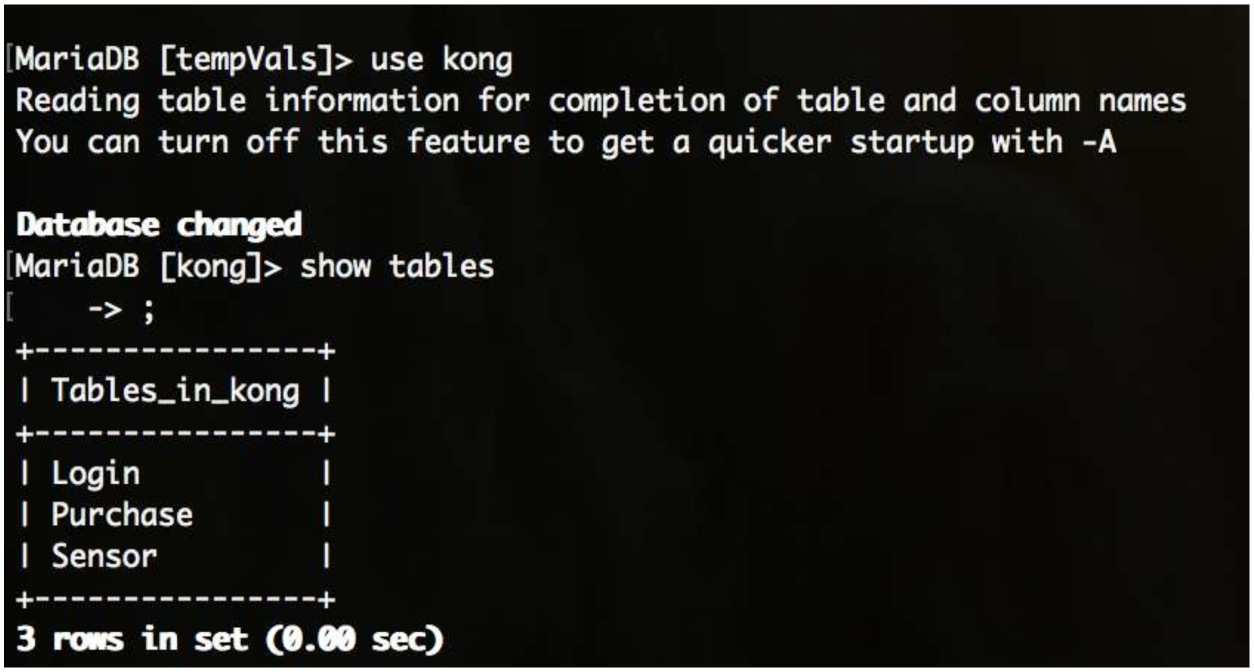

```
pi@raspberrypi:~/Desktop/kong/Hardware Scripts $ sudo python getTempHumid_test.py
Failed to get reading. Try again!
pi@raspberrypi:~/Desktop/kong/Hardware Scripts $
```

Figure 3. Failed to read sensor

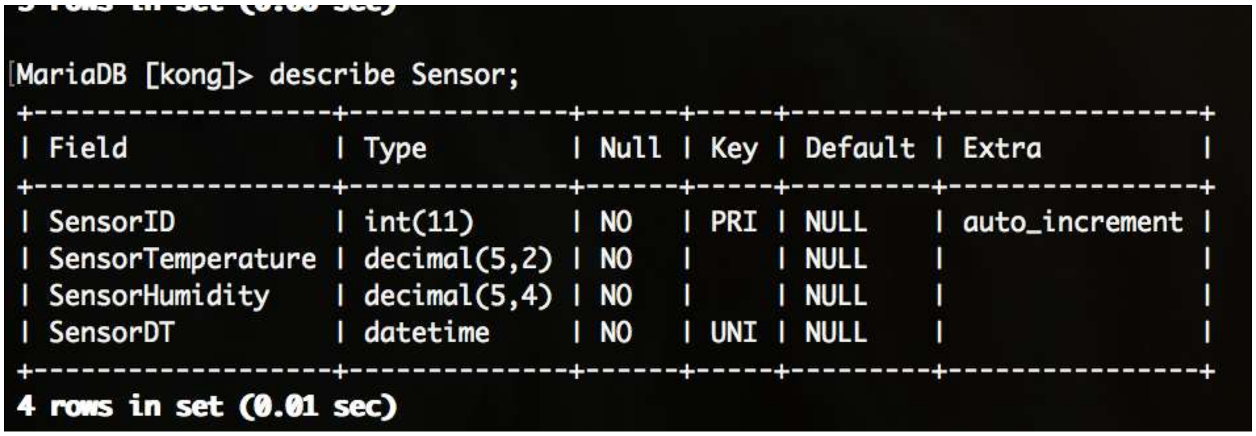
Database Test:
Description: This test is to ensure that the database is running and that data is being updated
Type: White box test.
Setup:
1. We used python scripts to test the communication between the database and the Raspberry Pi 3. We temporarily used dummy data for the humidity data since our sensor is not set up
2. We then verify if the new input to the database is as expected.

Results:
1. Database updated after the script is run.
2. Results are displayed in the mysql database.



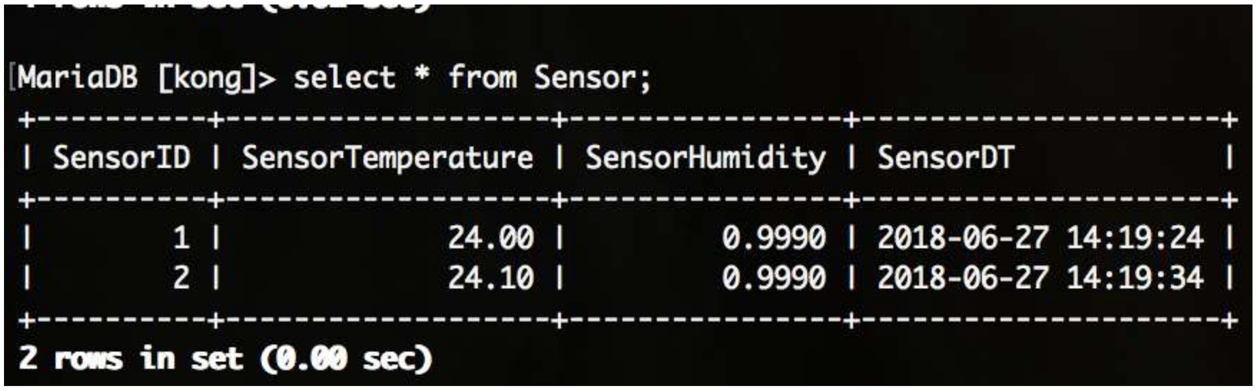
```
[MariaDB [tempVals]> use kong
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
[MariaDB [kong]> show tables
[    -> ;
+----------------+
| Tables_in_kong |
+----------------+
| Login          |
| Purchase       |
| Sensor         |
+----------------+
3 rows in set (0.00 sec)
```

Figure 1. 'Kong' Database Table.



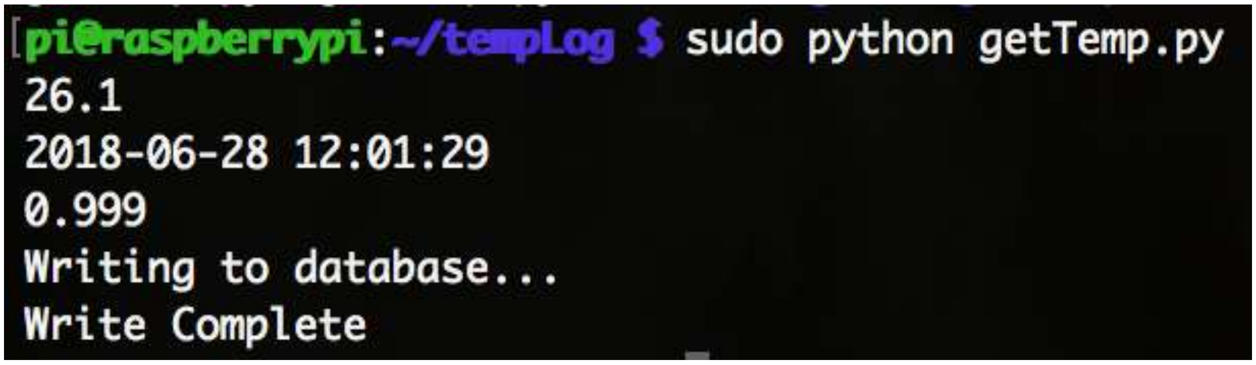
```
[MariaDB [kong]> describe Sensor;
+-------------------+--------------+------+-----+---------+----------------+
| Field             | Type         | Null | Key | Default | Extra          |
+-------------------+--------------+------+-----+---------+----------------+
| SensorID          | int(11)      | NO   | PRI | NULL    | auto_increment |
| SensorTemperature | decimal(5,2) | NO   |     | NULL    |                |
| SensorHumidity    | decimal(5,4) | NO   |     | NULL    |                |
| SensorDT          | datetime     | NO   | UNI | NULL    |                |
+-------------------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Figure 2. 'Kong' Database, 'Sensor' Table data types



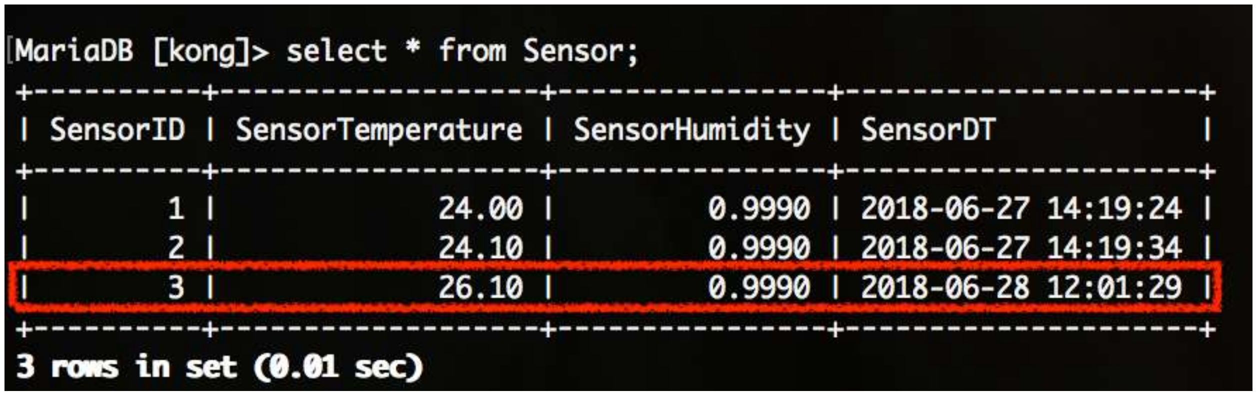
```
MariaDB [kong]> select * from Sensor;
+----------+-------------------+----------------+---------------------+
| SensorID | SensorTemperature | SensorHumidity | SensorDT            |
+----------+-------------------+----------------+---------------------+
|        1 |             24.00 |         0.9990 | 2018-06-27 14:19:24 |
|        2 |             24.10 |         0.9990 | 2018-06-27 14:19:34 |
+----------+-------------------+----------------+---------------------+
2 rows in set (0.00 sec)
```

Figure 3. 'Kong' Database, 'Sensor' Table imported data.



```
pi@raspberrypi:~/tempLog $ sudo python getTemp.py
26.1
2018-06-28 12:01:29
0.999
Writing to database...
Write Complete
```

Figure 4. Running script:



```
MariaDB [kong]> select * from Sensor;
+----------+-------------------+----------------+---------------------+
| SensorID | SensorTemperature | SensorHumidity | SensorDT            |
+----------+-------------------+----------------+---------------------+
|        1 |             24.00 |         0.9990 | 2018-06-27 14:19:24 |
|        2 |             24.10 |         0.9990 | 2018-06-27 14:19:34 |
|        3 |             26.10 |         0.9990 | 2018-06-28 12:01:29 |
+----------+-------------------+----------------+---------------------+
3 rows in set (0.01 sec)
```

Figure 5. 'Kong' Database, 'Sensor' table, updated data after script run.

Manual Test:
Front End: Tyler and Charles

- **Test 1**

  Feature set added: Login and Password integration into the menu bar:

  What we did for testing: After login and password fields were added into html code dummy user identities were entered to test the functionality of the system. Results of the testing confirmed (prior to database integration) that our dummy identities would, in fact, route the user to the proper landing page and leaving the fields blank would prompt the user to enter the required information. These results matched our expectations for this aspect of the landing page.

  User Story: User Jenny wants to log into her account and check the status of her greenhouse. She enters the appropriate log in credentials (her email address and chosen password) into the login field on the main page. Upon hitting the log in button she is taken to the data.html page which successfully displays the current temperature and humidity in her greenhouse as well as historical data encompassing the last 24 hour period, week and fortnight. **Test Successful**

Purpose: Verify User Story 1

Server beng used: Local Host

Prerequisites for this test: homepage and data page are created and user is in database

Software Version:

Browser: Google Chrome Version 67.0.3396.99 (Official Build) (64-bit)

Operating System: Windows 10 64-bit

Required Configuration: No special setup needed

Test Case Name:	**Test 1**	**Test ID #:**	1
Description:	Testing to see that a user with an existing username and password can log into their account and access temperature and humidity data, as well as logout	**Type:**	

Tester Information

Name of Tester:	Tyler Valentine	**Date:**	7.2.18
Hardware Ver:	**1.01**	**Time:**	14:00
Setup:	Website running on local host		

Step	**Action**	**Expected Result**	**Actual Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Input username	username field fills with entered text	username was displayed on screen	x			
2	input password	password field fills with starred put text protecting the user	password is starred out in the password field	x			
3	login	user hits the login button and is directed to the panel	after logging in, the user is presented with their panel	x			
4	logout	user hits the logout button and is directed to the home	hitting the logout button successfully redirects the user to the home screen	x			
5							
6							

Overall Test Result: Testing was successful and website performed as intended.

- **Test 2**

  Feature set added: Creating a new user account:

  What we did for testing: When a customer decides they want to purchase the product they are prompted to enter new username and password information. We needed to test to make sure that this information was successfully recorded in preparation for handling by the database and back-end. Testing confirmed that information was collected and was ready for storage. Additionally, when any of the required fields were not properly entered the user was prompted to provide said information before a new account would be created. Finally, we confirmed that was an account was successfully created the user was passed to the proper confirmation page.

  User Story: User Aidan has visited the main page and is interested in buying our product. He decides to purchase and clicks the buy now button. He is routed to the signup page and asked to create a username (his email address) and password as well as shipping and payment information. He enters a username that is already in use and is prompted to create a new username. Upon creation of a valid username he is routed to the success page letting him know that his account has been successfully created. He can now expect to receive his product in the mail and begin accessing data online once hardware setup has been completed. **Test Successful.**

Purpose: Verify User Story 2

Server beng used: Local Host

Prerequisites for this test: homepage and signup page are created and linked to database

Software Version:

Browser: Google Chrome Version 67.0.3396.99 (Official Build) (64-bit)

Operating System: Windows 10 64-bit

Required Configuration: No special setup needed

Test Case Name:	**Test 1**	**Test ID #:**	2
Description:	Testing that a new user can "purchase" the product and set up a new account.	**Type:**	

Tester Information

Name of Tester:	Tyler Valentine	**Date:**	7.2.18
Hardware Ver:	**1.01**	**Time:**	22:00
Setup:	Website running on local host		

Step	**Action**	**Expected Result**	**Actual Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	User opens types in url of main page	Kong Grow home page opens	page opened as expected	x			
2	user clicks the purchase button on the main page to order the product for the first time	user will be directed to the signup page to create new account and provide payment/shipping details	signup page opens successfully	x			
3	user fills out all but one of the fields, simulating someone forgetting to provide all necessary data	website should prompt user to fill out all required fields	website points to the field that was left blank and displays the message, "please fill out this field"	x			
4	user fills out all fields but with a preexisting username	user will be directed to the signup fail page which informs the user that this username has already been taken	new webpage is displayed informing the user that the selected username has already been used	x			
5	user fills out all fields with valid inputs	user will be directed to the success page confirming that the order has been placed and the account has been created	success.html page is displayed to the user confirming purchase and account creation. Option to return to the homepage is displayed.	x			
6	user clicks on the return to homepage link	user is returned to the Kong Grow main page	user is successfully returned to the main page	x			

Overall Test Result: Testing was successful and website performed as intended.

- **Test 3**

  Feature set added: Displaying real-time and historical data on the webpage:

  What we did for testing: Test that graphing data plugin is displaying information properly. Also, test that real-time data has been updating as expected. Testing confirmed that data from the raspberry was being collected and sent to the database to be displayed on the website.

  User Story: User Miguel is untrusting of technology by nature. He wants to make sure that the hardware is has ordered from our website is working properly and that the results are being displayed properly online. He gets a digital thermometer at the local hardware store and takes measurements in his greenhouse at regular intervals to be compared with the data that is being displayed on our website. After careful review he finds the real-time data displayed on the sight matches his expectations based on his own measurements. Peace of mind is achieved! **Test Successful.**

Purpose: Verify User Story 3

Server beng used: Local Host

Prerequisites for this test: homepage and data page are created and user is in database

Software Version:

Browser: Google Chrome Version 67.0.3396.99 (Official Build) (64-bit)

Operating System: Windows 10 64-bit

Required Configuration: No special setup needed

Test Case Name:	**Test 1**	**Test ID #:**	3
Description:	Testing to see that the user's own temperature readings from their own temperature reading tool is the same as the displayed current temperature on their user panel after logging into the website	**Type:**	

Tester Information

Name of Tester:	Tyler Valentine	**Date:**	7.2.18
Hardware Ver:	**1.01**	**Time:**	19:00
Setup:	Website running on local host, manually collect temperature data		

Step	**Action**	**Expected Result**	**Actual Result**	**Pass**	**Fail**	**N/A**	**Comments**
1	Input username	username field fills with entered text	username was displayed on screen	x			
2	input password	password field fills with starred put text protecting the user	password is starred out in the password field	x			
3	login	user hits the login button and is directed to the panel	after logging in, the user is presented with their panel	x			
4	check temperature	current temperature is displayed on the panel	the user sees the current temperature on their screen	x			
5	compare to user's own readings	current temperature reflects the user's readings	the user's results and the device's results are within reasonable error				
6							

Overall Test Result: Testing was successful and accurate data is displayed.

Scrum Notes July 2, 2018

Tyler Valentine: Since the last meeting Tyler worked on the main page, signup page and successful purchase page including duplicates for failure to login cases. He also worked on testing for milestone 4 and collaborated with the other group members on style decisions for the website. Going forward he is working on final touches for the website, presentation slides and final presentation format.

Charles Mezhir: Since the last meeting Charles has completed the data.html page, integrating a graphing plugin so that real-time and historic data can be displayed on the user's data page. He also ran test cases for milestone 4, helping to complete the front end aspect of that assignment, and worked with the rest of the group on style decisions for the website. Going forward he is working on finalizing the data representation for the website, presentation sides and final presentation format.

Bader Albader (scrum master!): Since last week Bader has continued to work extensively on using the raspberry pi to collect temperature and humidity data and relay that information to the database. He also worked with Jacob on writing the code for and performing actual test cases for the hardware/database side of milestone 4. Additionally, he found presentation templates that were appropriate for milestone 5 and presented them to the group. Going forward he is working on presentation slides, actual presentation details and providing data to Charles to finalize the population of the data.html page.

Jacob Tran: Jacob has been working with Garrett on linking the database to website with nodejs. He has also been acting as lead on preparing the lecture slides and formatting the actual presentation. He worked with Bader to create code and actual test cases for the hardware/database aspect of milestone 4 and formatted all of our contributions for said milestone into one document. Additionally, he had a hand in the finalization of the webpage, helping with photoshopping background image files and making style decisions. Going forward he is continuing to make progress on milestone 5, helping with nodejs, and finalizing presentation plans.

Garrett Senor: Garrett has continued to learn and implement the nodejs aspect of this project. This has turned out to one of the most difficult and most foreign concepts for the group to tackle and Garrett has been at the forefront of this activity. He also contributed with style for the web design and presentation slides. Going forward he will continue to work on communication between the webpage and the database, milestone 5 and finalizing the presentation.