Solar Decathlon Smart Home Control

Technical Reference Manual

Table of Contents

Project Information

Introduction

The Solar Decathlon Smart Home Control app provides a monitoring and control system for Clemson's entry into the U.S. Department of Energy's Solar Decathlon for 2015. The home is equipped with multiple sensors that have access to a computer within the house. The computer stores information about the house for users of the home control app.

The purpose of this document is to not go over basic usage of the app; instead, this document is to serve as a technical reference for anybody interested in knowing what is going on underneath the hood of our app. Our goal is to create a document that outlines all of the details of our app, so that knowledge is easily transferrable to outside parties (including future CPSC 482 students). If you have any questions about our app that cannot be answered by this document, then we have failed in that goal. Nonetheless, I encourage all interested parties to get in touch with any of the team members if necessary (using the contact information included in the Development Team section below).

Target Devices

We aim to support all devices that can run iOS 8, which consists of: the iPhone 4s, 5, 5c, 5s, 6, 6 Plus, and iPod Touch 5th generation. If time allows, we will extend support to the iPad Air and iPad Mini sizes.

Target Audience

The target audience for our app is comprised of three parties: students working on the home, contest attendees, and contest panel judges. Students working on the home (including those not on our software team) will benefit from being able to monitor and adjust conditions within the home for testing, comfort, and convenience. We suspect that the contest attendees that visit our home will largely want to test out the capabilities of our system, so preparing for extreme use cases will be of utmost importance in order to appeal to these attendees. Lastly, we intend to develop and debug our project in such a way that would appeal to the contest judges, who are likely going to experiment with various usage patterns and inspect our project for standout innovation and creativity.

Schedule of Implementation

Date	Jackson	Joey	Alex
11/11/14	Dashboard prototype completed	External database created and dummy values inserted	Basic graph functionality implemented
11/18/14	All UI prototypes completed; uses charting	Create PHP scripts; Update Database design	Graph functionality completed; some user system completed; some User Manual completed
11/25/14	Constraints added for different screen sizes/devices	Admin functionality; help with the User & Technical Reference Manual	User system completed; User Manual completed; some Technical Reference completed
12/02/14	Added sound effects to interactions	Added gesture functionality	Technical Reference completed
12/05/14	First draft of Software and Documentation completed		
12/09/14	Final draft of Software and Documentation		
12/11/14	Final presentation (8:00 - 12 noon)		

Features

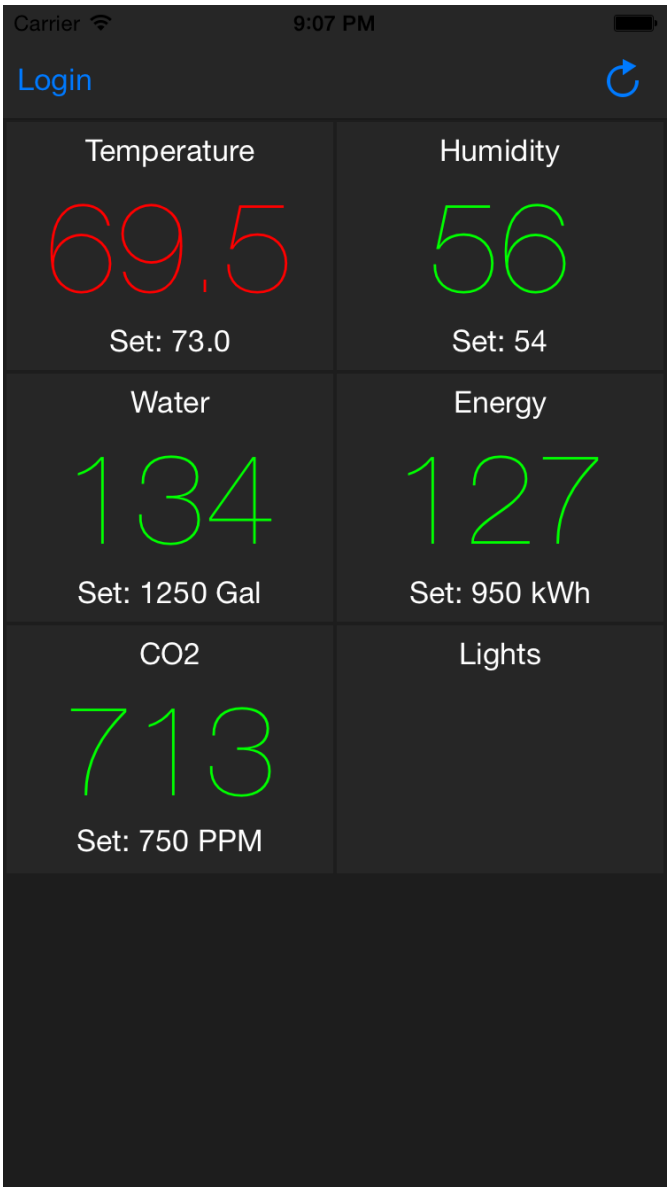


Figure 1: Dashboard view

Dashboard with Real-Time Metrics

The dashboard includes tiles for temperature, humidity, power consumption, water consumption, CO_2 ppm, lighting status, and motion detection. These tiles will be updated every minute (to save battery) or every time any view controller is loaded (viewDidAppear method) to ensure that relevant information is always readily available. The purpose of the dashboard is to give users a quick summary of all sensors within the home, while more detailed views can be accessed through tapped a specific tile.

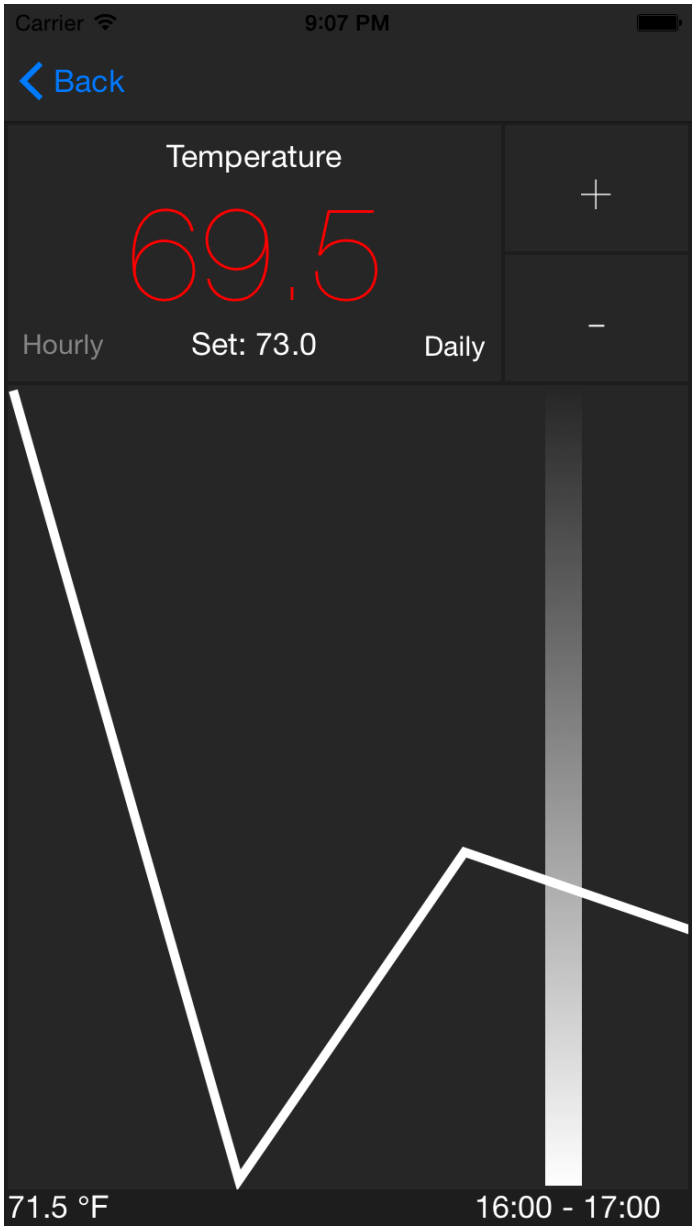


Figure 2: Detailed view for temperature

Detail View for Each Statistic

There are six individual views for each monitored condition (except for the lights and motion data), which show the historical time-series data for the chosen monitor in graph form. When the view is initially loaded, the user sees the current status of the sensor, along with an hourly breakdown (in a chart) of the sensor. Swiping right on the chart will allow the user to expand the data to show a daily, weekly and monthly breakdown of the sensor's data.

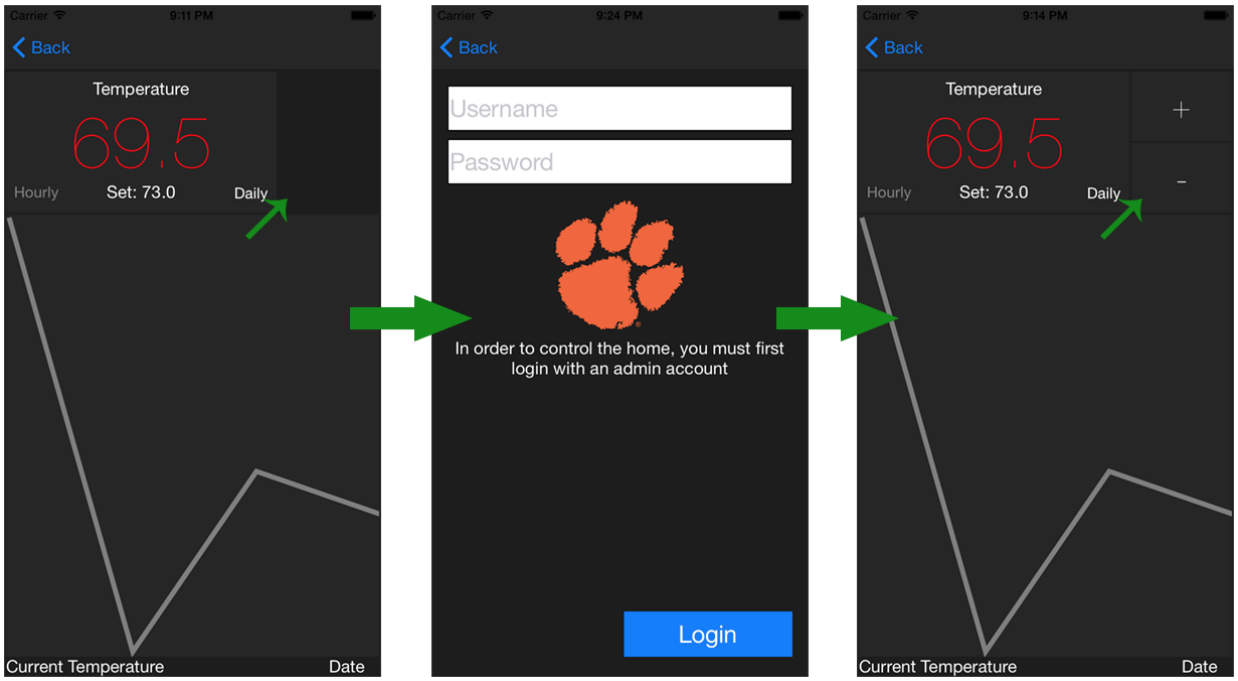
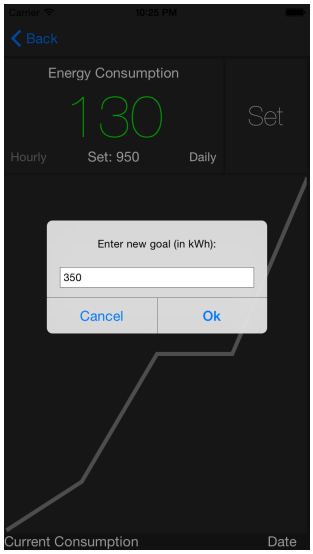


Figure 3: A user must login to an administrative account to be able to modify settings

Administrative Abilities

When a user is logged in as an administrator, the detail view is slightly different. Next to the sensor's current status is a small (+) and (-) so the admin can increase/decrease the preferred temperature, humidity, etc. Users without administrative permissions (guests) will only be allowed to view the statistics of the home, not modify them. The buttons will still appear, but will not be enabled.



The alternative to incrementing and decrementing values for some statistics is using a dialog box. This is particularly useful because some values (like CO2 Content) are measured in thousands and setting these values one at a time would be nearly impossible.

Similar to the (+) and (-) buttons, the Set button is disabled unless the user is signed in as an administrator.

Customizable Lighting Controls

The Philips Hue lighting set is a intelligent lighting set that is controlled by a hub in a home. This hub and the lights are networked together, and allow the hub to control both intensity and color of each individual light.

The Hue API was used to give the Smart Home Control app access to the individual lights, and a table was set up with all available lights. Tapping a light would select it and give the user the ability to control the light with ease.

Settings Bundle and Preferences Panel

This is used to save admin logins (if the user does not want to have to login each and every time the app is loaded). This is almost necessary, as without this ability, signing in would add considerable time to the usage of the app. The admin status of the user is checked at login, and each time a detailed view loads, in order to determine whether or not to enable the controls on that view.

Benefits

The Thunder Ducklings strongly believe that there are a number of benefits to implementing the Smart Home Control app. These benefits, while broad in scope, can be summarized into three main areas: convenience, assistance, and safety.

The convenience of the Smart Home Control app is one of the significant benefits to building the app. There are a number of scenarios with which it would be beneficial to have access to real-time metrics of your home. For example, you could keep tabs on your utilities to make sure money isn't being wasted, and even optimize your usage to save money on your monthly bills. You could also check and adjust these values from within the home, or remotely. If it is too cold in the morning, you could pull up the app to turn the thermostat up to a comfortable level. If you left home for the night and forgot to turn the lights off, you could easily pull up the app and turn them off with just a couple taps.

Usage of the Smart Home Control app goes beyond simply adjusting the thermostat, however. We believe that one of the biggest draws of the app will be for those who are disabled and unable to easily complete minor household tasks, such as turning off the lights or shutting the blinds. While the initial plans of our app only include temperature, humidity, CO_2 ppm, water usage, power usage, light status, and motion detector metrics, we intend to use every actuator available to us to make these daily tasks less tedious and more accessible to all users.

Lastly, we believe that the safety benefits are another large part of what will make the Smart Home Control app great. Not only will the app actively keep track of the air density of CO_2 molecules, but it will warn you if you approach an undesirable level and offer tips to air out your home. Additionally, with the array of sensors and actuators at our disposal, any user can remotely monitor their home--quickly seeing if an intruder has opened a door or turned on a

light, for example. Motion sensors will also play a key role in indicating if there is movement in the house when there shouldn't be.

Databases

External Database

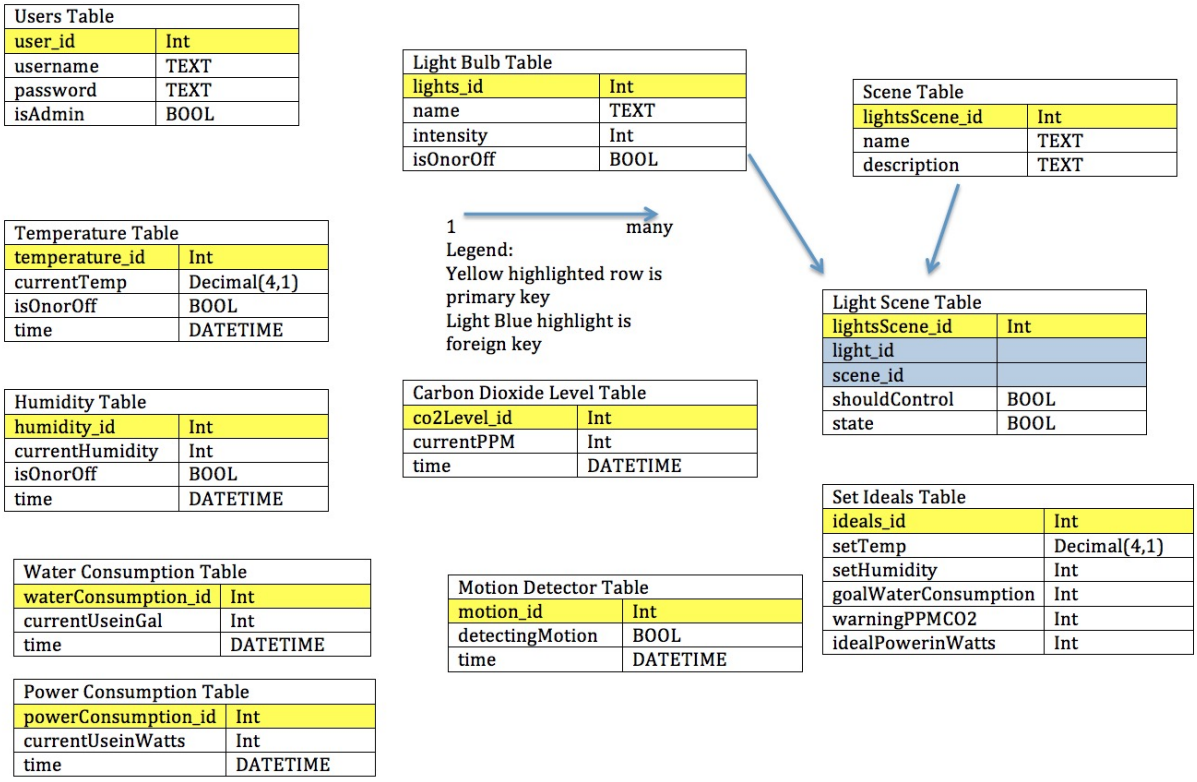


Figure 5: External database schema

The shouldControl field in the Lights Scene table has to do with creating a scene. The user needs the ability to turn certain lights on or off certain lights and not affect other ones. For example, if the user wants to turn on the kitchen light and the bedroom light on but not change the status of the front door light, then the should Control field in this scene would be set false for the front door light, but true for the kitchen and bedroom lights.

The lights scene table will hold foreign keys pointing to the lights table and the scene table. An entry in this table will hold a light, a scene, whether or not the light should be controlled and the current state. This table will hold all of the possible groupings of the lights possible and based on the Scene_id each bulb will be able to be controlled.

The time field in each table will record the date and time of each sensor. When a specific sensor adds a new entry into the database the exact date and time is recorded. The time will be useful in displaying graphical data for the user.

The Ideals table will only have 1 entry holding the values of each of the set values for the table. If the set temperature in the ideals table is different than the current temperature of the temperature table then the app will know to turn on an actuator. (The set ideals are being held in a separate table so there is not a lot of duplicate data)

The name value in the Light Bulb Table is used to signify the naming of a specific bulb (instead of the bulb being named Light bulb 35, for example, the user can name each of the individual bulbs they have placed in their house). The default intensity for a bulb when the bulb is turned on will be set to 100.

The User Table will be the mechanism we use to keep track of the logged in user, as app functionality depends largely on permission level. Not being logged in, or being logged in as a standard user, will allow you to view trends and current metrics for the various sensors. Being logged in as an administrator, on the other hand, allows the user access to modify settings, for example, adjusting the temperature.

Please refer to the internal database for a more detailed description of the other tables.

PHP Functions

The Smart Home Control app doesn't use any mobile device features such as GPS or the camera; instead, we simply store, retrieve, and modify values from a MySQL database using the following PHP functions.

- Add Admin script: This would be used to submit a new admin login to the database.
- Admin Login: This evaluates a given username and password against the database of usernames and passwords to determine whether or not a user is listed in the admin table.
- "Get All" scripts: These scripts return arrays of all of a specific value to the app. These include getAllHumidity, getAllPower, getAllTemperatures, and getAllWater.
- "Get Ideal" scripts: These return the values from an "Ideals" table within the database. This table contained the ideal or "set" values from all kinds of statistics. These include getIdealCO2, getIdealHumidity, getIdealPower, getIdealTemp, and getIdealWater.
- "Get Last" scripts: These scripts returned the last entered values for a given kind of statistic. These include getLastCO2, getLastHumidity, getLastPower, getLastTemperature, and getLastWater.
- "Insert" scripts: These insert new "current" values to their respective places in the database. These include insertCO2, insertHumidity, insertPower, insertTemperature, and insertWater.
- "Set Ideal" scripts: These set the new ideal value in the ideals table for the respective statistic. These include setGoalWaterConsumption, setHumidity, setIdealPowerinkWh, and setTemperaure.

For an example of each of these types of scripts, see the Appendix.

Development Team

Our development team is comprised of three undergraduate students from CPSC 482 (Fall 2014).

Jackson Dawkins is the team leader of the Thunder Ducklings and a junior Computer Science major with a Psychology Minor at Clemson University. He has a love for beautiful interfaces and keeping things simple, yet powerful and engaging.

Contact: jacksod@g.clemson.edu
Web: jakedawkins.me

Joey Costa is a junior Computer Science major at Clemson University.

Contact: jacosta@g.clemson.edu
Web: people.cs.clemson.edu/~jacosta/homePage.html

Alex Ross is a senior Computer Engineering major at Clemson University. He will be graduating in December and starting his job as a software developer near San Francisco, California.

Contact: alross@clemson.edu
Web: alexross.io

Acknowledgments

Thank you **Dr. Pargas** for adequately preparing us to take on such a challenging app, and for giving us the tools necessary to make it great.

Thank you **Dr. Blouin** for meeting with us, and providing for us the guidance needed to create the Smart Home Control app.

Thank you to **stackoverflow.com** for a lot of help in coding.

Thank you to our fellow **482/682 students** for joining us on this journey!

Appendix

Example PHP scripts

One of each category listed in “PHP Functions” section.
Sensitive database information removed.

AddAdmin.php

```
<?php

$dbhostname = '';

$dbusername = '';

$dbpassword = '';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}

if (isset ($_GET["username"], $_GET["password"]))
{
            $username = $_GET["username"];
            $password = $_GET["password"];

}
//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

//echo "Connected to Database"."<BR>";

    $sql_statement = "INSERT INTO users (username, password, isAdmin)
    VALUES ('$username', '$password', 1)";

$rec_insert = mysql_query( $sql_statement);

if(! $rec_insert )
```

```
{
  die('Could not enter data: ' . mysql_error());
}

echo "$username added to the database\n";

mysql_close($conn);
```

AdminLogin.php

```
<?php

$dbhostname = '';

$dbusername = '';

$dbpassword = '';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}

if (isset ($_GET["username"], $_GET["password"]))
{
            $username = $_GET["username"];
            $password = $_GET["password"];

}
if(empty($username) || empty($password))
{
echo ('Username or password field was empty');
}

//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

 $result = mysql_query("SELECT * FROM users WHERE username = '$username' &&
password = '$password'");
if (!$result) {
    echo 'Could not run query: ' . mysql_error();
```

```
    exit;
}
$row = mysql_fetch_row($result);
//echo "ARE YOU AN ADMIN?  ";
echo $row[3]; // the is admin value
// if this is a 1 then you are an admin.
// if this is a 0 then your not an admin
// if this is NULL then the data you entered is not in the database

mysql_close($conn);
```

getAllTemperatures.php

```
<?php

$dbhostname = 'mysql1.cs.clemson.edu';

$dbusername = 'ul0f8gzj';

$dbpassword = 'thunderducklings3';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{
  die('Could not connect: ' . mysql_error());
}

//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

//echo "Connected to Database"."<BR>";


$sql_statement = "SELECT * FROM temperature";
$result = mysql_query($sql_statement);


    while($row = mysql_fetch_assoc($result))
    {
        echo $row['currentTemp'] . " " . $row['time']. "<br>";
    }



//echo "Entered data successfully\n";
```

```
mysql_close($conn);
```

getIdealTemp.php

```
<?php

$dbhostname = 'mysql1.cs.clemson.edu';

$dbusername = 'ul0f8gzj';

$dbpassword = 'thunderducklings3';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}


//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

//echo "Connected to Database"."<BR>";


  $result = mysql_query("SELECT * FROM idealValues
    ORDER BY id DESC LIMIT 1");
if (!$result) {
    echo 'Could not run query: ' . mysql_error();
    exit;
}
$row = mysql_fetch_row($result);
//echo "Set Ideal temperature is:  ";
echo $row[1];
//echo "Entered data successfully\n";

mysql_close($conn);
```

getLastTemperature.php

```
<?php

$dbhostname = 'mysql1.cs.clemson.edu';
```

```
$dbusername = 'ul0f8gzj';

$dbpassword = 'thunderducklings3';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}


//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

//echo "Connected to Database"."<BR>";


  $result = mysql_query("SELECT * FROM temperature
    ORDER BY id DESC LIMIT 1");
if (!$result) {
    echo 'Could not run query: ' . mysql_error();
    exit;
}
$row = mysql_fetch_row($result);
//echo "Most recent temperature is:  ";
echo $row[1];
//echo "Entered data successfully\n";

mysql_close($conn);
```

insertTemperature.php

```
<?php

$dbhostname = 'mysql1.cs.clemson.edu';

$dbusername = 'ul0f8gzj';

$dbpassword = 'thunderducklings3';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);
```

```
if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}

if (isset ($_GET["currentTemp"]))
            $currentTemp = $_GET["currentTemp"];
//echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

//echo "Connected to Database"."<BR>";


  $result = mysql_query("SELECT * FROM idealValues");
if (!$result) {
    echo 'Could not run query: ' . mysql_error();
    exit;
}
$row = mysql_fetch_row($result);
//echo "Set temperature is:  ";
$setTemp = $row[1];
//echo $setTemp;

if($setTemp == $currentTemp)
{
$isOnorOff = FALSE;
}
else
{
$isOnorOff = TRUE;
}
    $sql_statement = "INSERT INTO temperature (currentTemp, isOnorOff, time)
    VALUES ($currentTemp, $isOnorOff, now())";

$rec_insert = mysql_query( $sql_statement);

if(! $rec_insert )

{

  die('Could not enter data: ' . mysql_error());

}
```

```
//echo "Entered data successfully\n";
 echo $isOnorOff;
 // print out whether the thermometer should be turned on or not.
mysql_close($conn);
```

setTemperature.php

```
<?php

$dbhostname = 'mysql1.cs.clemson.edu';

$dbusername = 'ul0f8gzj';

$dbpassword = 'thunderducklings3';

$conn = mysql_connect($dbhostname, $dbusername, $dbpassword);

if(! $conn )

{

  die('Could not connect: ' . mysql_error());

}

if (isset ($_GET["setTemp"]))
            $setTemp = $_GET["setTemp"];
echo 'MySQL Connected successfully'."<BR>";

mysql_select_db("SmartHomeExternal_l4mz") or die(mysql_error());

echo "Connected to Database"."<BR>";

    $sql_statement = "UPDATE idealValues
    SET setTemp = $setTemp";

$rec_insert = mysql_query( $sql_statement);

if(! $rec_insert )

{
  die('Could not enter data: ' . mysql_error());
}

echo "Entered data successfully\n";

mysql_close($conn);
```

References

1. **JawBone Chart View Library**
    - Charting library used to make the graphs for our 5 detailed views
    - https://github.com/Jawbone/JBChartView

2. **Octave Sound Library**
    - Sound effect library used to add sound to various button presses to improved the user experience
    - http://raisedbeaches.com/octave/

3. **Stackoverflow**
    - Various error fixes and general information reference
    - http://stackoverflow.com/questions/tagged/ios

4. **iOS Developer Library**
    - General information reference for iOS/Xcode/Objective-C
    - https://developer.apple.com/devcenter/ios/index.action

5. **iTunesU Stanford iOS Lectures**
    - Video lectures for iOS/Xcode/Objective-C
    - https://itunes.apple.com/us/course/developing-ios-7-apps-for/id733644550

6. **Josh Hull**
    - Help and especially useful database and php examples
    - http://www.joshuahull.net/blog/