# 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 finer 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

## 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.

## 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.

## 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.

## Customizable Lighting Controls

This includes being able to control individual lights with a simple tap on the light, or having the ability to create custom "scenes" where one can control a group of lights. The app will also give the user the ability to adjust the intensity of each of the bulbs.

Scenes are being implemented as a way to easily modify a group of lights so that individual lights don't have to be adjusted every time you leave the house, for example, and need to turn them all off. We envision Scenes to be a powerful tool with many use cases--such as parties or mood lighting. Automatic time-bases Scenes will be especially beneficial for utility optimization.

## 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), as well as to set an acceptable tolerance for each statistic. For example, if the preferred temperature in the home is set to 70 with a 3 degree tolerance, the user is not notified that the home is at an unacceptable temperature until it drops below 67 or rises above 73.

# 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

## Internal Database

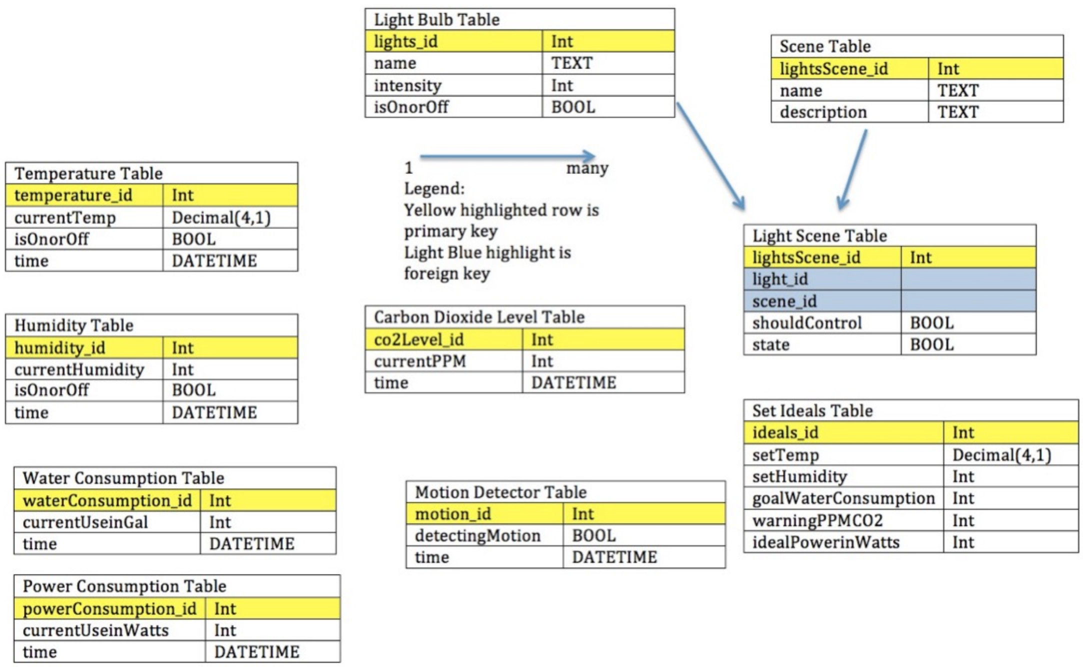


*Figure 1: Internal database schema*

Figure 1 depicts our planned internal database schema for the app. The internal and external databases will hold the same information (minus the users table)

The shouldControl field in the Lights Scene table has to do with creating a scene. The user needs the ability to turn on or off certain lights and not affect other ones. For example, if the user wants to turn on the kitchen light and bedroom light 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.

## External Database

| Users Table | |
|---|---|
| user_id | Int |
| username | TEXT |
| password | TEXT |
| isAdmin | BOOL |

| Light Bulb Table | |
|---|---|
| lights_id | Int |
| name | TEXT |
| intensity | Int |
| isOnorOff | BOOL |

| Scene Table | |
|---|---|
| lightsScene_id | Int |
| name | TEXT |
| description | TEXT |

| Temperature Table | |
|---|---|
| temperature_id | Int |
| currentTemp | Decimal(4,1) |
| isOnorOff | BOOL |
| time | DATETIME |

1 many
Legend:
Yellow highlighted row is primary key
Light Blue highlight is foreign key

| Light Scene Table | |
|---|---|
| lightsScene_id | Int |
| light_id | |
| scene_id | |
| shouldControl | BOOL |
| state | BOOL |

| Humidity Table | |
|---|---|
| humidity_id | Int |
| currentHumidity | Int |
| isOnorOff | BOOL |
| time | DATETIME |

| Carbon Dioxide Level Table | |
|---|---|
| co2Level_id | Int |
| currentPPM | Int |
| time | DATETIME |

| Set Ideals Table | |
|---|---|
| ideals_id | Int |
| setTemp | Decimal(4,1) |
| setHumidity | Int |
| goalWaterConsumption | Int |
| warningPPMCO2 | Int |
| idealPowerinWatts | Int |

| Water Consumption Table | |
|---|---|
| waterConsumption_id | Int |
| currentUseinGal | Int |
| time | DATETIME |

| Motion Detector Table | |
|---|---|
| motion_id | Int |
| detectingMotion | BOOL |
| time | DATETIME |

| Power Consumption Table | |
|---|---|
| powerConsumption_id | Int |
| currentUseinWatts | Int |
| time | DATETIME |

*Figure 2: External database schema*

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 admin, on the other hand, allows the user access to manipulating desired values--e.g., adjusting the temperature.

Refer to the external database for a more detailed description for the other tables.

## 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 starts his job as a software developer in February out near San Francisco, California.

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