CANOPY

Michael Chapman, CSE, John Curci, EE, Justin Thibodeau, CSE, and Zachary Windoloski, CSE

***Abstract* — Two of the largest contributors to home energy costs in the United States are heating and cooling [1]. The Canopy automated shade system will help consumers cut down on energy use by controlling the amount of sunlight that enters a home. The system integrates with the Nest automated thermostat over WiFi, in order to determine whether the home is currently being heated or cooled. This information is combined with input from a light sensor at each shade unit, in order to determine the position of the shade. Nest integration allows Canopy to take advantage of Nest's machine learning, which improves the system's heating and cooling efficiency. Another major focus of the system is convenience. Through a custom Android application, users can control their shades remotely, and view the light levels at each window equipped with a Canopy shade system. Settings controlled through the app will determine the behavior of the shades, and buttons on the shade unit will allow for direct control of the shades.**

I. Introduction

The motivation for our project comes from the high cost associated with heating and cooling homes and buildings. In 2013, the average American with natural gas spent $680 per year on residential heating [2], and those with air conditioners spent an additional $280 per year in 2010 [3]. Considering the fact that 26% of American adults don't have any money saved, and approximately 38 million households live paycheck to paycheck, this cost of about $1000 per year is a significant burden on many people [4]. However, heating and cooling buildings is far more than just a financial problem. It poses a problem on the global scale as well, as the use of HVAC systems takes a direct toll on the environment. It is estimated that air conditioners release approximately 100 million tons of carbon dioxide every year [5], and in 2006, approximately 8% of US Carbon Dioxide emissions came from residential HVAC [6].

There are several existing solutions that attempt to solve this problem by making HVAC systems more efficient. The Nest thermostat, which has received significant publicity in the past few years, saves energy by providing more intelligent temperature control compared to a standard thermostat [7]. According to a recent report, the Nest thermostat reduces homeowners' costs by an average of 12% on their heating bill and 15% on their cooling bill [7]. Ceiling fans made by the company Big Ass Fans provide another solution to the problem. These fans integrate with the Nest thermostat, and make Nest more effective by allowing the system to control the airflow in a building [8]. The energy savings are most significant in the summer, when the added air movement allows a user to raise the temperature on their thermostat without feeling warmer [8]. Big Ass Fans claims that for every degree the Nest thermostat is raised in the summer, customers will save 5% on their energy bill [8]. A third solution is provided by the company Lutron. Lutron makes an automated shades system that saves energy by providing season-based settings: "winter warm" keeps the shades open to let in light during the winter, while "summer cool" closes the shades to keep out light during the summer [9]. Since sunlight is a significant source of heat gain in a home, controlling the light entering through windows depending on the season helps make air conditioners more efficient in the summer and heaters more efficient in the winter [10].

Like Lutron, we will create an automated window shades system that controls indoor temperature by changing the amount of sunlight coming in through the windows. However, we will provide added customization beyond Lutron's two energy-based settings. This will allow users more fine grained control over the light entering their windows. In addition, like Big Ass Fans, we will integrate with the Nest thermostat to provide additional intelligence and energy savings. The specifications for our design are shown in Table I.

TABLE I
Specifications

Outdoor light sensor	+/-50% precision 0.5-120,000 lux
Shades	Width – 23" Maximum extension – 64" Height and slat angle control
Device communication	IEEE 802.11
Direct communication security	PSK Encryption
Web based data storage integrity	Encrypted Signatures
Power	Standard US 120V Wall AC
Remote control	Android app

II. Design

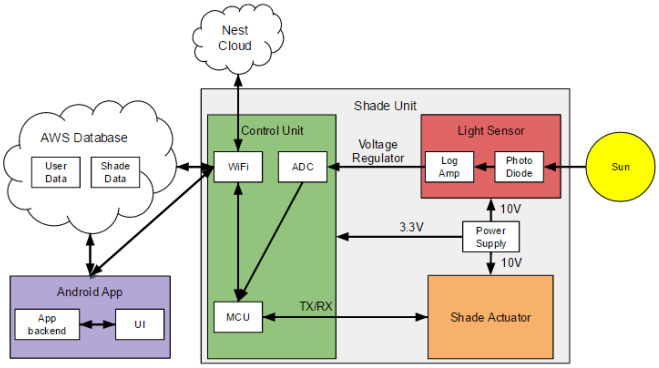

Figure 1: Canopy Block Diagram

A. Overview

Figure 1 above shows the design of our automated shade unit. The main control unit is a WiFi integrated microcontroller which will continually communicate with the Nest cloud and the user. This control unit will serve as a processing and communication hub, with inputs coming from the Nest cloud, the user via Android app and an online database, and a light sensor. For our control unit, we considered three ideas: a centralized hub controlling multiple shades, an external server acting as the hub, and distributed control units built into each shade unit. Distributed units were chosen since each device would require a microcontroller and some form of wireless communication regardless of our approach. Considering that our computation requirements are relative low, microcontrollers placed within each unit will be sufficiently powerful for this application. We will use a photodiode-based light sensor to measure outdoor light levels. This sensor element was chosen for its ability to detect a wide range of light levels including darkness and bright sunlight. The Canopy Android application will allow the user to control his or her shades remotely, and to view the light levels at each window in their home as measured by our light sensors. The output of our system is a shade actuator which opens and closes the window shade. For our actuator, we considered a stepper motor but decided on a servo motor because it is able to provide more precise position control. For a power supply, we considered battery or solar powered approaches, but determined that using wall AC would be more convenient for the user and would allow for continuous operation in all weather conditions. We will use a switched-mode design to provide a compact and energy-efficient power source to provide the necessary voltages and currents to each system.

B. Block 1: Light Sensor

One of the core functions of Canopy is to provide an additional input for Nest's intelligent heating and cooling system. This is achieved using a light sensor, which measures the brightness level at each window in the user's home, and feeds that information into the Canopy control unit. This serves the dual purpose of improving the overall efficiency of the Nest system, and helping Canopy's algorithms balance the energy savings associated with opening or closing the shade against the user's light level preferences.

The sensor is based around the BH1620FVC-TR light sensing IC from Rohm semiconductors, which provides a linear output current across brightness levels spanning five orders of magnitude [11]. This relationship is shown in Figure 3. The LOG101AID logarithmic transimpedance amplifier from Texas Instruments was used to convert this signal, which included currents in the range of nanoamperes at the low end, into a voltage that could be read in by our microcontroller [12].

The light sensor subsystem must be able to measure changes of approximately +/-50% in bright from any starting point in the range of 0.5 lux (bright moonlight) to 120,000 lux (very bright sunlight) [13]. For testing, the sensor will be placed in a window for several days and the output voltage will be noted periodically. The output will be compared to the expected outdoor brightness levels on the corresponding calendar date and time of day. The result will be discarded if the sun is obscured by clouds. The test will be considered a failure if the sensor does not respond linearly to changes in outdoor brightness level within the specified range. While test coverage and precision will be limited by weather conditions, this test is expected to satisfy the tolerance of our design.

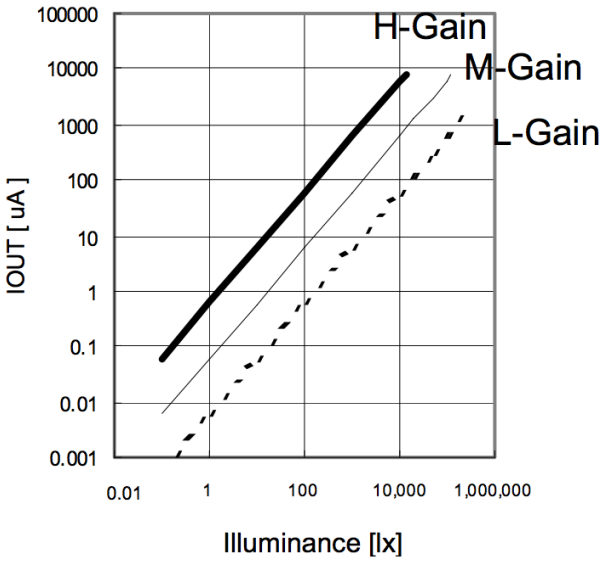

Figure 2: Gain vs Light Level BH1620FVC-TR

C. Block 2: Shade Actuator

The shade actuator is a key component of the system that is relied upon to adjust the angle of the horizontal blinds. For our project, the team chose Achim Home Furnishing's brand of

venetian blinds to allow for granular light levels [14]. The actuator must meet a few general requirements: first of all, it needs to have enough torque to rotate the shaft that turns the blinds. We calculated that the blinds require a maximum of 4 kg-cm to rotate, so the actuator must be rated for at least 4 kg-cm of standing torque. In addition, the blinds move from -90 degrees (closed, pointing down) to 0 degrees (fully open) to 90 degrees (closed, pointing up) for a full rotation of 180 degrees. Therefore, our actuator must be capable of rotating at least 180 degrees.

Having considered these requirements, the Dynamical AX-12A servo was chosen as our actuator [15]. The AX-12A provides a maximum of 10kg-cm of torque at 10V, which is more than sufficient for our purpose [15]. In addition, the AX-12A provides 300 degrees of rotation [15]. This allows us to achieve the full 180 degree rotation without any modifications. In order to perform the rotation, the motor is coupled to the rotating shaft inside the top of the shade. As the motor turns the shaft, the threads running through horizontal blinds move up or down, which changes their angle of the horizontal blinds.

Unlike most servos that are controlled with a PWM signal, the communication with the AX-12A is done via half duplex UART, where one wire is used for sending and receiving data [15]. Two 0xFF bytes indicate the start of the signal, followed by two bytes for the ID of the AX-12A unit, one byte for the length of the signal, and bytes for the instruction or error message. Currently, this half duplex UART is connected directly to the TX port on the microcontroller, so we can only send commands to the motor. In order to allow for both sending and receiving, the motor's half duplex UART must be converted into the full duplex used by our microcontroller, where separate ports are used for transmitting (TX) and receiving (Rx). We will build a simple tri-state buffer circuit shown below in Figure 3 to do this conversion [15].

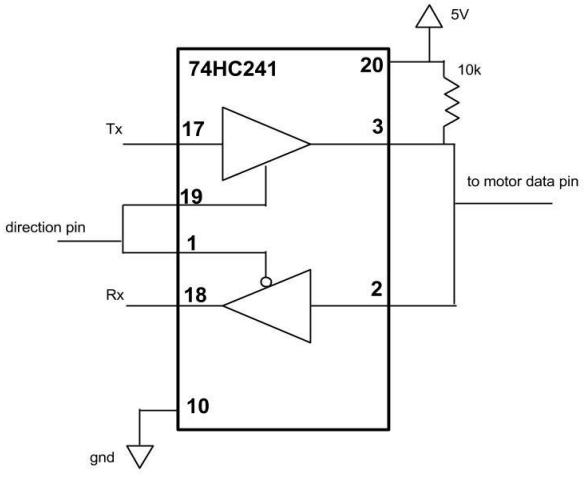

Figure 3: Full Duplex to Half Duplex

D. Block 3: Control Unit

Our control unit is the brains behind the entire canopy device. It must be able to take in an analog voltage signal from our light sensor and translate this into a usable light level value. It needs to be able to send and receive bytes of data over a TX/RX connection to our motor. These signals will be used to set shade positions, and receive feedback from the motor. It needs to be capable of communicating to the internet in order to access user defined settings held in an Amazon Web Services database [16], Nest thermostat information from Nest's own data cloud [17], and be able to communicate directly with a user's phone in order to be set up on their in-home network.

The Adafruit Huzzah board [18] was chosen to perform these tasks for several reasons. The onboard ESP8266 processor provides us a microcontroller with full WiFi support to act as both an access point and as a client. There are low power modes available that allow the board to disable WiFi communication in order to save power, which will allow the system to sense data more frequently than it communicates over the internet. Also included is a watchdog timer, which allows the board to be put into a very low power state when not actively sensing or communicating, while still able to receive an interrupt signal to wake up at specific times or when the user interacts with the device.

When acting in access point mode, we have the board respond to GET requests with an xml list of the networks it is able to see, and respond to POST requests containing an SSID and password by connecting to that network and saving the information to its internal EEPROM. We have the password encrypted using a pre-shared key so we do not send it in the request as raw text. When the board is switched into client mode, it connects to the network and is now able to communicate with the internet. Initially, it sends an NTP request via UDP to get the current UTC time. This will eventually occur on a daily basis and any time the device is restarted to ensure that the correct time is set. Once it receives the time, it has all the extra information needed to send requests to our database instance. The board will first send a query to the database to get all data for the Shade entry corresponding to this shade (a predefined ID stored in the controller), in order to get the most up to date user requested behavior. Currently the only setting available is a direct control over the shade allowing it to be set to 1 of 4 positions: closed, half open, full open, and reverse half open. This was done to let us focus on communication between all pieces of the project, before focusing on the behavior algorithm itself. Once we know the requested behavior, the microcontroller will send a packet of data over its TX connection to our shade actuator containing the corresponding command whereupon we will be able to see the shade angle change.

After setting the shade angle, the controller will read the current input from the light sensor attached to its ADC connection as a value between 0 and 1023. This value will then pushed back to the same table entry in the database so the user can see the current lighting conditions. This will be expanded to include information about the current position of the shade as this will depend on its behavior algorithm and not be set

directly by the user. To inform the behavior algorithms, we will be implementing the Nest API in order to check the current behavior of the rest of the in-home heating/cooling system, and determine whether opening or closing the shade will assist in moving the inside temperature closer to the desired temperature.

To handle the required data storage, and model user to device relationships, we created a DynamoDB instance within Amazon Web Services (AWS) [16]. AWS provides us with a database that requires pre-shared credentials in order to access data, allowing us to ensure that all data is coming from a trusted place. It is also scalable, meaning that while testing we are only using a very small database, but when this project would eventually grow to multiple shades and users, the database will scale up to match demand.

E. Block 4: Mobile Application

The mobile application is how the user communicates with their shades and the overall shade system. The user will be able to add or remove shades from their system, and group them together by rooms. They will be able to use the application to apply settings for their shades, mainly their behavior mode and a schedule. The user will be able to create a custom schedule to define what shade positions and behaviors for different times of the day. The behavior modes we currently plan are manually set and energy efficiency. Depending on which is selected, further customization is added. If manually set is chosen, the user will select the shade position instantly. Energy efficiency does not require further customization as it tells the shade to act in whichever way will help make heating or cooling the most efficient. These settings will be applied to a single shade, a room of shades, or to all of the user's shades depending on how the user sorts them.

Aside from main functionality the application will need to accomplish a few other tasks. When a shade is added to a system, the application will connect the shade to the user's local wireless network. To do this, the application will connect directly to the shade's MCU network card which will act as an access point. After connecting, the user will select their network from a list of visible networks, enter the password if one exists, and enter the shade key which is used for shade identification. After the shade is on the network, the application will prompt the user for access to their Nest account, and communicate with Nest to generate an access key which the control unit will use to query the Nest cloud.

When deciding which mobile platform to develop in, there were three main choices, Android, iOS, and Windows. Windows had limited libraries and tutorials so it was quickly ruled out. The two remaining were very similar in offered functionality, but we decided on Android since the physical devices each group member owned ran Android.

Because mobile application development is not taught in any of our courses, we learned almost everything this semester. We used android's multiple tutorials and code samples as our primary sources of information [19]. Since the application integrates with Amazon Web Services (AWS) for data storage, tutorials and code samples provided by Amazon were also used during development.

In order to save settings and shade statuses, and to share that information between the application and the control unit, we needed a common database. We chose to use AWS's DynamoDB [16]. The application will be able to push and pull data from this database through by using provided libraries created for Android by Amazon.

Development for this application is being done through Android Studio [20]. It allows for the easiest way to create and then test the application. Here the application can be built through the IDE using the Java programming language. Android Studio also offers an emulator which we used to test application without having to load it onto a physical device. This allowed for testing of the application and testing between the application and AWS. Additionally, to ensure that the correct data was being sent to the database we were able to view the table itself through the AWS website.

III. Project Management

Our goal for MDR was to build up each subsystem enough in order to see communication between all of the blocks where connections should be made. The MDR goals for each group member are shown in Table II. The shade actuator needs to talk to the MCU to determine which way it should position the shades via the motor. That task was completed by Michael Chapman. In order for the application to show cross block communication it needed to be able to push and pull data from the AWS database. This was accomplished via a simple Android interface created by Zachary Windoloski. In order for the system to have information on current light levels, the sensor needed to provide a signal to the MCU that it could

TABLE II
MDR Goals

Team Member	Goal	Status
Michael Chapman	Shade Actuator Controlled by MCU	Complete
John Curci	Light Sensor outputting voltage to MCU	Complete
Justin Thibodeau	MCU I/O, network setup, AWS communication	Complete
Zachary Windoloski	App with simple UI and AWS communication	Complete

properly read. That task was completed by John Curci. The final subsystem, the MCU, needed to read the inputs from the light sensor, send signals to the shade actuator, and connect to the AWS database over WiFi. All of those features were completed by Justin Thibodeau.

Now that we established minimal functionality and cross subsystem communication, fleshing out of each subsystem is the next step. The shade actuator will need to provide feedback to the MCU to communicate the status of the motor. In addition, the actuator will need to have position calibration along with manual controls directly attached to the shade for the user. Multiple additions need to be done to the application. First the application will need to accommodate multiple shades with more settings per shade, as well as accommodating multiple users. The application will also need to offer an integrated method of connecting the control unit to the user's network, as well as authorizing the control unit to communicate with Nest. The control unit will need to be able to communicate with the Nest cloud, as well as run the algorithms to determine what position the shade should be in based off of all current data. Finally, the overall device needs a main power supply for the motor, control unit, and light sensor.

Our team is comprised of one EE and three CSE majors. This is working well since our project has a significant software focus. We tasked the EE in our group, John, with the light sensor and power systems since that is where his strengths lie, and he will also help with coding. Michael, a CSE, has been working with the shade actuator which is mechanically heavy and has done a great job getting that subsystem working with the control unit. Zachary, also a CSE, has been responsible for the mobile application and has been collaborating with Justin to coordinate AWS database communication and setup. Justin, the third CSE and Team Manager has been responsible for the control unit's interfaces, WiFi setup and communication, as well as keeping the team on track. So far every team member has accomplished the tasks that were assigned to them and met the deadlines for those tasks. The group has been meeting at least twice weekly: once with our advisor, and once without. The meeting with our advisor gives us a chance to ask questions when problems arise that we are unsure about and get invaluable guidance. In our team meeting without our advisor we each spend some time talking about the progress that has been made and where we plan to go next. This time has also been used to work together on pieces that cross subsystems, such as the AWS database. We managed to stay on the same page through these meetings as well as email contact and will continue this practice going forward.

IV. Conclusion

As stated above, for MDR we had proposed to have simplified working versions of each of the 4 main subsystems and to have them able to communicate with one another. We accomplished this and showed significant progress towards a full working prototype. Going forward, we plan to focus on each system's functionality and turning the project into a more self-contained, user friendly, and usable unit. A Gantt chart of our future schedule is show in Figure 4. At CDR, we will have a functioning power supply so that we are no longer reliant on multiple separate units. We will have implemented integration with the Nest cloud using the associated API. Our app will contain the functionality to set behavior modes, add and

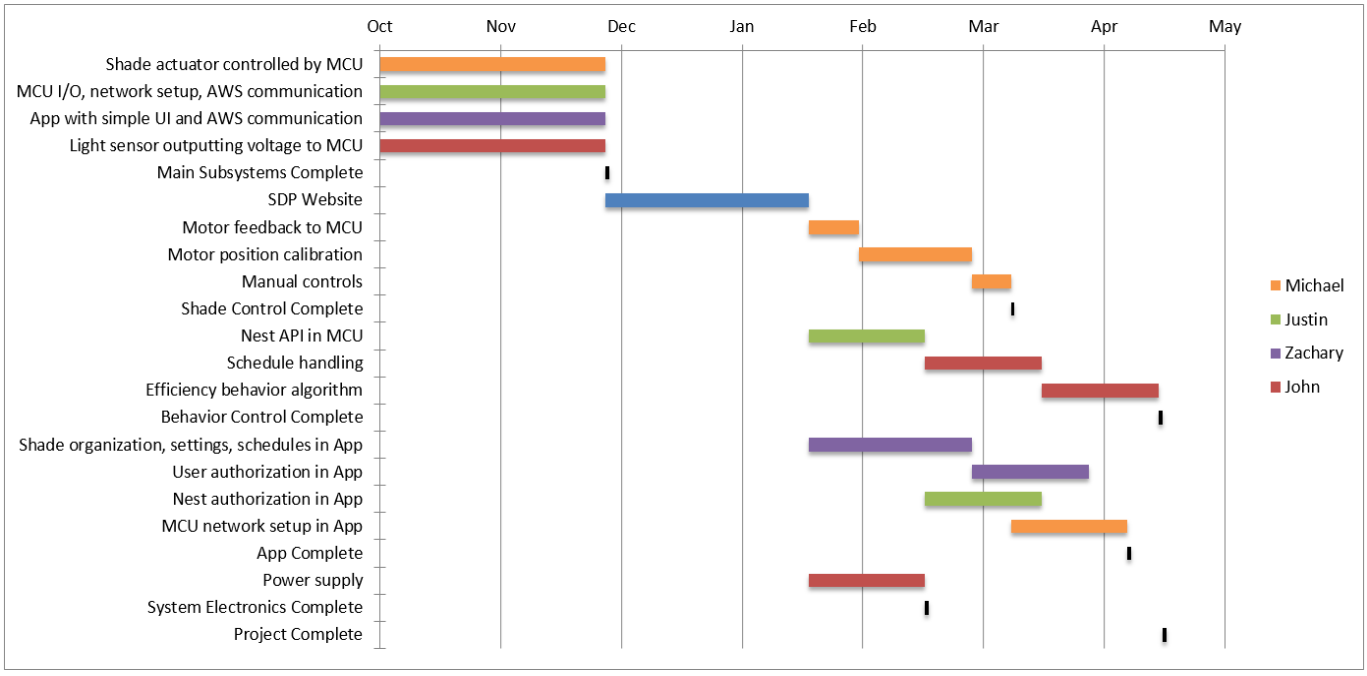

Figure 4: Canopy Gantt Chart

organize shades, and set schedules. Our shade actuator will have a calibration system giving feedback to the control unit in order to more accurately control the position of the shade. This will also allow for the implementation of manual controls.

After CDR, our shade actuator system will be complete, allowing Michael to shift his focus towards helping with the app. Specifically, he will be implementing direct communication between the app and the Canopy, which will facilitate wireless setup on the user's home network. With the power supply, our electronics will be finished and John will shift his focus to incorporating light sensor data into driving the energy efficiency behavior. Zachary will continue focusing on the app, adding in user authentication to handle multiple users and ensure system integrity. Justin will be working with the Nest API and simulator to further incorporate temperature into the control unit's behavior. He will also be implementing app-side authorization to allow the device to access a user's Nest account. We do not foresee any problems in the future and look forward to having a fully functional system.

ACKNOWLEDGMENT

We would like to thank our Faculty Advisor, Professor Csaba Andras Moritz, for being a guiding hand as we made design decisions for our project and for helping us refine our ideas. We would also like to thank our faculty evaluators, Professor T. Baird Soules and Professor Zlatan Aksamija, for giving us useful feedback and criticism at various stages of development. Regarding technical advice, we would like to thank Professor Joseph Bardin for his advice on the amplifier stage of the light sensor, as well as Professor Charles Malloch for his advice on implementing WiFi capabilities, and Daniel Bergman for his help with the motor setup and the associated machining.

REFERENCES

[1] Energy.gov, 'Tips: Your Home's Energy Use | Department of Energy', 2015. [Online]. Available: http://www.energy.gov/energysaver/tips-your-homes-energy-use. [Accessed: 09- Dec- 2015].

[2] Eia.gov, 'Heating costs for most households are forecast to rise from last winter's level - Today in Energy - U.S. Energy Information Administration (EIA)', 2013. [Online]. Available: http://www.eia.gov/todayinenergy/detail.cfm?id=13311#tabs_SpotPrice Slider-3. [Accessed: 09- Dec- 2015].

[3] Carbonrally.com, 'Carbonrally – air conditioner costs', 2015. [Online]. Available: http://www.carbonrally.com/challenges/22-air-conditioner-costs. [Accessed: 09- Dec- 2015].

[4] CreditDonkey, "23 Dizzying Average American Savings Statistics", 2016. [Online]. Available: https://www.creditdonkey.com/average-american-savings-statistics.html. [Accessed: 03- Jan- 2016].

[5] Energy.gov, "Air Conditioning | Department of Energy", 2016. [Online]. Available: http://energy.gov/energysaver/air-conditioning. [Accessed: 03- Jan- 2016].

[6] C2es.org, 'Buildings Overview | Center for Climate and Energy Solutions', 2015. [Online]. Available: http://www.c2es.org/technology/overview/buildings. [Accessed: 09- Dec- 2015].

[7] J. McGrath, "Nest studied customers' heating and cooling savings, and the results are in", *Digital Trends*, 2015. [Online]. Available: http://www.digitaltrends.com/home/how-much-money-does-nests-smart-thermostat-save/. [Accessed: 02- Jan- 2016].

[8] Big Ass Fans Introduces Integration with Nest Learning Thermostat » Big Ass Fans, "Big Ass Fans Introduces Integration with Nest Learning Thermostat » Big Ass Fans", 2014. [Online]. Available: http://www.bigassfans.com/big-ass-fans-introduces-integration-nest-learning-thermostat/. [Accessed: 02- Jan- 2016].

[9] Lutron.com, "Shading Solutions from Lutron Provide Energy Saving Light Control", 2016. [Online]. Available: http://www.lutron.com/en-US/Residential-Commercial-Solutions/Pages/Residential-Solutions/ShadingSolutions.aspx. [Accessed: 03- Jan- 2016].

[10] Savewithsrp.com, "SRP: Home inspection to reduce heat gain", 2016. [Online]. Available: http://www.savewithsrp.com/DIY/heatgain.aspx. [Accessed: 03- Jan- 2016].

[11] Rhom Semiconductor. (2012, Feb.). "BH1620FVC Ambient Light Sensor IC" [Online]. Available: http://rohmfs.rohm.com/en/products/databook/datasheet/ic/sensor/light/bh1620fvc-e.pdf [Dec 28, 2015].

[12] Texas Instruments. (2002, May.). "LOG101 Logarithmic and Log Ratio Amplifier" [Online]. Available: http://www.ti.com/lit/ds/symlink/log101.pdf [Dec 28, 2015].

[13] M. Burke, *Handbook of machine vision engineering*. London: Chapman & Hall, 1996.

[14] Amazon.com, "Amazon.com - Achim Home Furnishings Luna 2-Inch Vinyl Blind, 23 by 64-Inch, Mahogany - Window Treatment Vertical Blinds", 2016. [Online]. Available: http://www.amazon.com/Achim-Home-Furnishings-64-Inch-Mahogany/dp/B00FJDVDU4. [Accessed: 02- Jan- 2016].

[15] Support.robotis.com, "AX-12/AX-12+/AX-12A", 2016. [Online]. Available: http://support.robotis.com/en/product/dynamixel/ax_series/dxl_ax_actuator.htm. [Accessed: 09- Dec- 2016].

[16] Amazon Web Services, Inc., 'AWS | Amazon DynamoDB - NoSQL Cloud Database Service', 2015. [Online]. Available: https://aws.amazon.com/dynamodb/?hp=tile. [Accessed: 09- Dec- 2015].

[17] Developer.nest.com, "Nest Developers", 2015. [Online]. Available: https://developer.nest.com/. [Accessed: 30- Dec- 2015].

[18] Learn.adafruit.com, 'Overview | Adafruit HUZZAH ESP8266 breakout | Adafruit Learning System', 2015. [Online]. Available: https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout/overview. [Accessed: 09- Dec- 2015].

[19] Developer.android.com, "Getting Started | Android Developers", 2016. [Online]. Available: http://developer.android.com/training/index.html. [Accessed: 25- Jan- 2016].

[20] A. Overview, "Android Studio Overview | Android Developers", *Developer.android.com*, 2016. [Online]. Available: http://developer.android.com/tools/studio/index.html. [Accessed: 25- Jan- 2016].