**4.1 Overview**

This chapter gives an overview of the different parts of the circuit in brief. It mainly discusses about the requirements, theories, techniques and implementation of the whole system.

**4.2 Hardware and Software Requirements**

There are certain hardware requirements that need to be met to carry out this project. They are stated below:

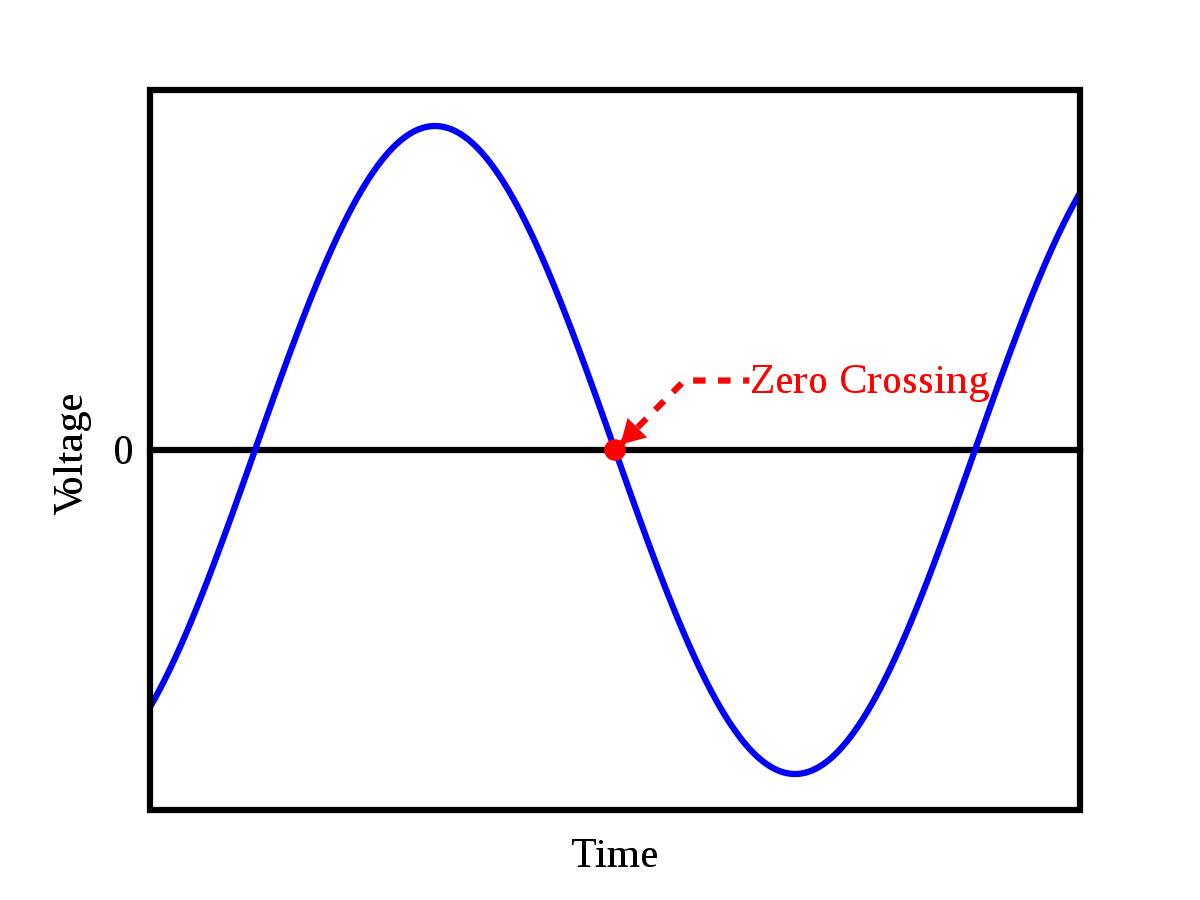
* Zero Crossing Detector Module
* Arduino Pro Mini
* ESP8266 Wifi Module
* Current Sensor Module
* Voltage Sensor Module

**4.3 Hardware**

The following descriptions give an idea about how we implemented different Hardware Modules in our project.

**4.3.1 Zero Crossing Detector (ZCD)**

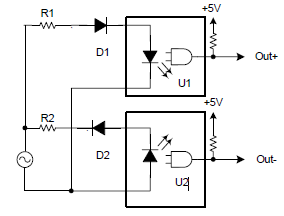
Zero crossing is a term commonly used in electronics, mathematics, image processing and sound. It is the point in a sinusoidal wave, where the mathematical function transits from a positive cycle to a negative cycle, or vice versa. Zero-Crossing detector is a device that is used mainly for measuring the frequency, period, or phase difference of a periodic AC signal by detecting the time period between two or more zero crossing points.



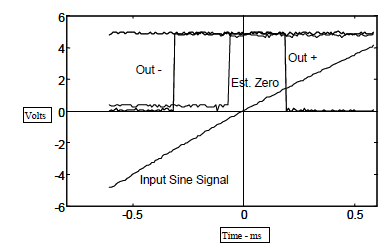
**Figure 4.3.1a:** Zero Crossing point of a sine wave.

There are many methods to detect zero crossing proposed by various authors. Some of them are:

* Zero crossing detection by interpolation – This uses two signals from a single source where one point is found just before the 0V of the main signal and the other point is found right after the 0V of the same signal. These two signals are then interpolated to converge towards a new point that is close to the 0V point. This method required computer processing.

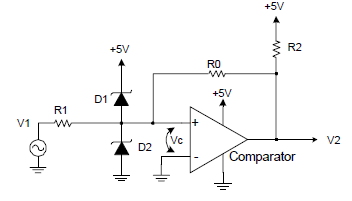


**Figure 4.3.1b:** Circuit for dual point interpolation method for detecting a zero crossing.



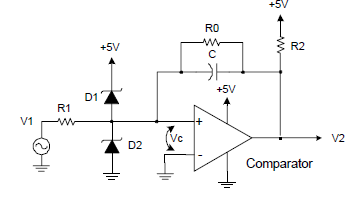
**Figure 4.3.1c:** Oscilloscope capture of the sine wave signal, optoisolator outputs and computed zero crossing.

* Comparator circuits with fixed hysteresis – This circuit takes the input of the main signal and compares the voltage with two reference voltages. One that is at zero and another that is close to zero. This prevents multiple zero crossing detection that occurs near the zero crossing point of the sinusoid.



**Figure 4.3.1d:** Resistive feedback hysteresis circuit.

* Comparator circuits with dynamic hysteresis. – This circuit works just like the fixed hysteresis circuit but has dynamic threshold voltages that further prevent multiple zero crossings. A capacitor is added at the feedback that adds to the positive feedback when the first zero crossing is detected. And then the feedback slowly decays overtime to dynamically change the threshold voltage.



**Figure 4.3.1e:** Dynamic hysteresis comparator circuit.

This theory is needed to know when a zero crossing is detected; the signal can be delayed at will, using a microcontroller to the input of a TRIAC gate. This will cause the TRIAC to switch on at a different phase of the AC source, hence controlling the output power towards the load.

For the purpose of this project a different approach was taken that was ideal for our application since it had isolated high voltage and low voltage sides. This has an advantage of lowering the risk of damage caused to the micro controller due to a power line fluctuation. And also isolation provides an exception of unwanted noise that may interfere with the micro-controller.



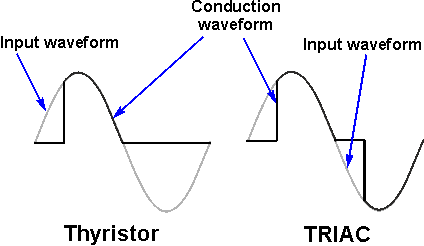
**Figure 4.3.1f:** Zero Crossing Detector.

The 230V AC signal is filtered out and rectified using a full bridge rectifier and then dropped across the 22k ohm resistor to make a voltage divider. The voltage drop across the 22k ohm resistor is approximately 11V. When the sinusoid reached zero the LED in the optocoupler turns off and is on during the rest of the cycle as the capacitor charges and discharges and triggers the transistor on and off. At this stage the signal is still not perfect yet, as the slew rate is much higher and might give wrong results when read from a microcontroller as the interrupt will trigger at rising or falling edge. Thus a Schmitt Trigger was introduced to the circuit to further process the signal. A Schmitt trigger is a bistable circuit in which the output increases to a steady maximum when the input rises above a certain threshold, and decreases almost to zero when the input voltage falls below another threshold. This is how a pure square wave will be achieved to make the signal more stable and precise.

**4.3.2 TRIAC Dimmers**

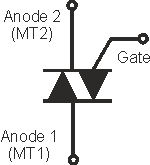
The TRIAC is a three terminal semiconductor device for controlling current. It gains its name from the term **TRI**ode for **A**lternating **C**urrent. It is effectively a development of the SCR or thyristor, but unlike the thyristor which is only able to conduct in one direction, the TRIAC is a bidirectional device.

The TRIAC is an ideal device to use for AC switching applications because it can control the current flow over both halves of an alternating cycle. A thyristor is only able to control them over one half of a cycle. During the remaining half no conduction occurs and accordingly only half the waveform can be utilized.



**Figure 4.3.2a: Typical / idealized TRIAC & thyristor switching waveforms**

The circuit symbol recognizes the way in which the TRIAC operates. Seen from the outside it may be viewed as two back to back thyristors and this is what the circuit symbol indicates.



**Figure 4.3.2b: TRIAC symbol for circuit diagrams.**

On the TRIAC symbol there are three terminals. These are the Gate and two other terminals are often referred to as an "Anode" or "Main Terminal". As the TRIAC has two of these they are labeled Anode 1 and Anode 2 or Main Terminal, MT1 and MT2.

The instantaneous pulses from the zero crossing detector is registered by the microcontroller which is then delayed and sent to the gate of the TRIAC at different intervals to change the conduction cycle of the AC waveform at the TRIACs output. This allows the user to change the overall voltage RMS voltage at will.



**Figure 4.3.2c:** TRIAC circuit.

The TRIAC and microcontroller are isolated through an optocoupler. There needs to be a galvanic separation between the microcontroller side and anything connected to the mains. **Galvanic isolation** is a design technique that separates electrical circuits to eliminate stray currents. Signals can pass between **galvanically isolated** circuits, but stray currents, such as differences in ground potential or currents induced by AC power, are blocked. The live wire of the AC mains is connected to one of the anodes of the TRIAC and the load is connected to the other end of TRIAC and the neutral line to complete the circuit. The gate is connected to the TRIAC side of the optocoupler, as the TRIAC inside the optocoupler is not rated for high currents.

**4.3.3 Current sensor**

Knowing the amount of current being delivered to a load can be useful in a wide variety of applications. For example, in low-power consumer products the supply current can be monitored to understand the system’s impact on battery life. The load current also can be used to make safety-critical decisions in over-current protection circuits.

Two types of current sensing is usually used, direct and isolated. In this project we will only be working with isolated current sensing which is based on Faraday’s and Ampere’s laws. A coil is placed, for instance a Current Sensor around a current-carrying wire and then a voltage is induced across the coil that is proportional to the current. This allows us to do a non-invasive measurement where the current sensing circuit is not electrically connected to the monitored system, hence not interfering with the entire system.

Direct current sensing is Ohm’s law based. A shunt resistor is placed in series with the system load and then a voltage is generated across the shunt resistor that is proportional to the system’s load current. The voltage across the shunt can be measured by differential amplifiers such as operational amplifiers (op amps). But this method is an invasive measurement as the shunt resistor and sensing circuitry are electrically connected to the system. Therefore, direct sensing typically is used when galvanic isolation is not required. The shunt resistor also consumes some amount of power, which may not be desirable.

This project will be focusing on the current sensing that provides us with galvanic isolation by using a current transformer of 2000:1 ratio.



**Figure 4.3.3a:** using current transformer.

The current coming through the load will be passing through a coil that will generate a voltage in the secondary coil. The goal is to add a DC voltage to the generated voltage so that a microcontroller can read the signal. Any negative voltage is undesirable as the microcontroller can read any voltage that is below 0V. This can damage it or give inaccurate readings. There needs to be a burden resistor across the secondary coil as this defines the output voltage range. This designer needs to choose this resistor according to the current carrying capacitance of the wire and or circuit as well the safe voltage that the ADC can read without damaging it. And also through judgment on how much current will be typically measured for the system. The circuit above solves this problem of a negative voltage by adding a DC offset to the generated voltage on the secondary coil. Two resistors of the same value were placed in series on one of the nodes on the secondary side and was supplied the reference Vcc and Gnd as shown in the circuit diagram. This provides a DC offset of 2.5V (if Vref is 5V) that makes sure the microcontroller can sample all the values. The DC offset is then removed using a simple code, programmed inside the microcontroller that implements a High Pass filter which removes the DC offset.

**4.3.4 Voltage sensor**

**AC voltage measurement** is implemented by converting AC voltage into proportional DC Voltage using rectifier and filter circuits, similar to current sensors. Similar to DC voltage measurement Voltage divider is constructed using two 1M Ohm resistors and two 10K resistor to step down the voltage. Two 5V Zener diodes are used to protect Arduino from accidental excess voltages. The 10K resistors need to be chosen wisely for calibrating the voltage that can be read by an ADC. Typically the stepping down of the voltage is achieved using a voltage step down transformer which also provides galvanic isolation. But voltages transformers are usually very bulky and will increase the overall form factor of the whole system along with the cost. This is why it was decided to make one without a transformer as shown in the figure below.



**Figure 4.3.4:** Voltage sensor.

**4.3.5 Arduino Pro Mini**

Arduino is a full blown development board that is an open-source electronics platform which is easy-to-use with its dedicated compiler. Arduino boards are used to write digital outputs or read various analog and digital inputs from various sensors. It operates by telling the board what to do by sending a set of instructions to the onboard microcontroller. To do so, the Arduino programming language (based on Wiring), and the Arduino Software (IDE), is used. There are many boards out there provided by the manufacturer.

And some of them are:

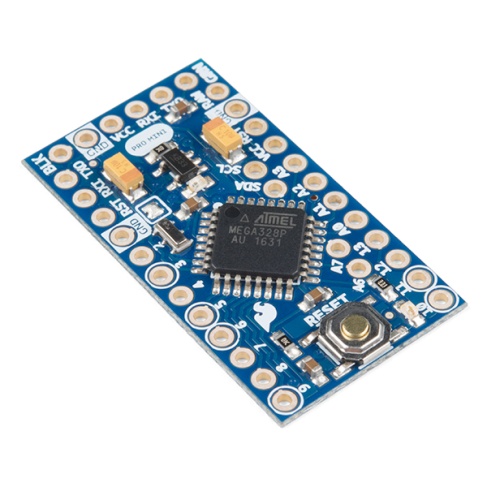
Arduino Uno R3

Arduino Mega

Arduino Lilypad

Arduino Nano

Arduino Pro Mini



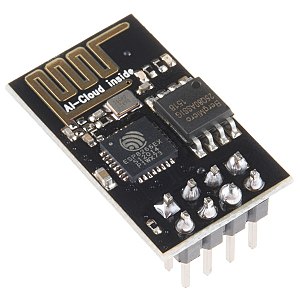
**Figure 4.3.5:** Arduino Pro Mini.

For this project, mainly the Arduino Pro Mini will be used for its cheap price, small form factor, a desirably large flash memory, speed and availability.

**4.3.6 ESP8266**

ESP8266 is a wifi SOC (system on a chip) produced by [Espressif Systems](http://espressif.com/). It provides full internet connectivity in a small package in a highly integrated chip design.

ESP8266 is used as an external Wifi module by using the standard AT Command set Firmware, provided as a factory default, by connecting it to any microcontroller using the serial UART, SPI or I2C protocol or directly serve as a Wifi-enabled microcontroller, by programming a new firmware using the provided SDK.



**Figure 4.3.6:** ESP 8266 Wifi Module.

This board has been around for a few years now, and has been mostly used in IoT applications, where we may want to add connectivity to a Arduino project. It has been adopted widely and is facilitated by a very cheap price, ranging from 190tk to 800tk which depends on the features offered by the manufacturers.

This board is used to transfer and receive various commands to and from the web interface to the Arduino through the UART protocol. This device runs strictly on 3.3V and anything above will damage it and anything below will render it useless, as it requires significant amount of power to stay connected wirelessly. This also was has a feature to wirelessly update the firmware over the air (OTA) that writes the received firmware in the EPROM and flashes it from there. This is extremely helpful for us as the firmware can be updated even after the system has been deployed.

The RX pin needs to be connected to the TX pin of the Arduino through a voltage divider network to ramp it down to 3.3V, since the TX pins on the Arduino supplies 5V. The CH\_PD needs to be connected to VCC. The TX pin needs to connect to Arduino’s RX pin (No need for any additional components in between). And all the GND pins need to be connected at a common ground node.

When uploading the code, the GPIO 0 pin needs to be grounded before starting up the module to upload any code. And then reset the module without the ground pin connected for the code to run. This needs to be repeated every time a code is uploaded, i.e, Ground GPIO 0, Start the module, Upload code, take the ground wire off from GPIO 0, Reset. These steps need to be repeated for every consecutive uploads. Making dedicated buttons for ground and reset is advised to make things easier.

**4.3.7 Measuring the frequency of AC signal by Zero Crossing Detector (ZCD)**

Generally, reading the frequency of the AC signal of the mains is not required as it is mandatory for all power generation companies to maintain a certain frequency all the time. But since a zero crossing detector is being used in our system, it will be a great troubleshooting feature to implement in our system to check if everything is in order. That is, the indication would be, if there is a change in frequency it would be safe to assume that the zero crossing detector might have malfunctioned.

The frequency is measured by the microcontroller by receiving a zero crossing pulse and starting up a counter. The counter will keep running until the next zero crossing point is detected. And then calculate the frequency by inversing the time period that is accumulated by the counter. This measure will give us a frequency of 100Hz if everything is in order, as the zero crossing point occurs twice in a 1 cycle of a 50Hz signal. Thus the frequency read is then divided by 2 to get the main line frequency. A few consecutive reading generated by this method, when averaged, will give us a far more accurate reading.

**4.3.8 Web application**

Django is written in Python and is a free and open-source web framework that follows the model-view-template (MVT) architectural pattern. Established as an independent non-profit organization, it is maintained by the Django Software Foundation (DSF), Django makes sure that the creation of complex, database-driven websites can be made with ease. It emphasizes reusability and plugging in of components, rapid development, and the principle not repeating the same thing each time a new app is built. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Views:

A view function, or view for short, is simply a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of either a Web page, a redirect, a 404 error, an XML document, or an image etc. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it’s on your Python path. There’s no other requirement. The convention is to put views in a file called “views.py”, placed inside the project or application directory.

Models:

A model is the single, definitive source of information about the database. It contains the essential fields and behaviors of the data you're storing. Generally, each model maps to a single database table. The basics: Each model is a Python class that subclasses django.db.models.Model.

These models are use to store the voltage, current, power factor, apparent power and real power information that will be called when the charts are populated inside the HTML files that will be rendered by the views class.

Application Programming Interface (API):

Application Programming Interface, commonly known as API is a software intermediary that allows two applications or two different programming languages to interact with each other. It is a set of routines, protocols, and tools for building software applications.

JSON encoding and decoding:

Django REST framework is a powerful and flexible toolkit for building Web APIs. The API is created in JSON format for this project because it is very easily encoded and decoded with only a few lines of code before the data is ready for consumption.

Websockets (Django Channels):

WebSockets represent a long awaited evolution in client/server web technology. They allow a long-held single TCP socket connection to be established between the client and server which allows for bi-directional, full duplex, messages to be instantly distributed with little overhead resulting in a very low latency connection. In other words, Django calls this “Channels”. Channels is a project to make Django able to handle more than just plain HTTP requests, including WebSockets and HTTP2.

How it was made:

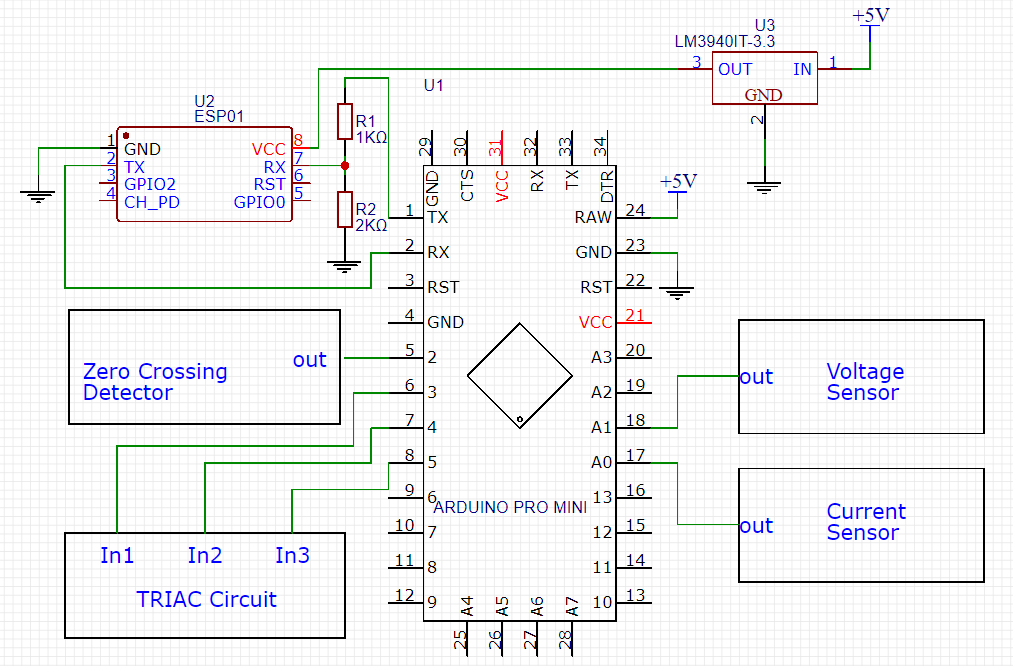
At first a template was created using the Django framework where the HTML file was rendered. Then javascript was introduced along with JQuery to create the sliders for controlling the home appliances. But these sliders will be useless if they do not send and receive real time data. And thus websockets were implemented using the Django channels. Later the monitoring web app for the power line was implemented using a javascript plug in called HighCharts for a new HTML page. The data inside the charts were populated using the Django REST framework API but then again, the charts were not updating in realtime and had to be refreshed every time to generate new data. Thus websockets were implemented again to populate the charts with new data as soon as a new websocket JSON text was received.

**4.3.9 Communication**

A stream of JSON text containing the slider information is sent to the webserver that is sending that data to any channel that is connected to it, in this case, the wifi module inside the home automation system. This text is then sent via the UART protocol to the Arduino which the decodes the JSON text into objects and then controls the dimming of the lights by firing the gate of the TRIAC at different user defined intervals proportional to the sliders.

The power data is generated from the Arduino and then encoded into JSON text that is then sent to the ESP8266 wifi module using the same UART protocol via serial. This text is then received and sent to the web server via websockets. Same as previous, the Django channels handles this request and send this text to any consumer that is connect to this link. In this case, the charts app is connected to this channel from where it receives this text and decode the text into JSON objects and then populates the charts.

**4.3.10 Implementation and device setup**



**Figure 4.3.10:** Pin connections for the modules

The whole system was connected was ultimately connected as follows:

The zero crossing detector’s output pin was connected to the Arduino’s interrupt pin which is Pin 2. The triggering inputs for the TRIAC circuit was provided by the microcontroller’s pin 3,4 and 5 for 3 different TRIAC switches. The voltages reading coming out from the voltage sensor were connected to the ADC pins of the Arduino A1. Similarly, the current sensor’s output was connected to arduino’s analog input A0. Since the ESP8266 is a 3.3V module, a voltage regulator was used to step down the 5V supply to a 3.3V supply to power the module. The RX pin of the ESP8266 wifi module was connected to the arduino’s TX pin through a voltage divider network as the Arduino gives out a 5V peak voltage that the ESP cannot handle. Thus the voltage divider network effectively converts the voltage to 3.3V peak. The TX pin of the ESP was direct connected to the Arduino that’s gives out a 3.3V peak, but no signal processing is required as the Arduino can read 3.3V peak as a logic HIGH and ground as a logic LOW. The rest of the modules are powered through 5V DC supply, these were not shown to maintain clarity in the schematic. A common ground was used throughout the system.

**4.4 Summary**

In this chapter various methods of zero crossing detection was discussed and in the end it was shown how a different version of zero crossing detector was made along with circuit schematics. A schematic of the TRIAC circuit was also given along with a description about how it works with a microcontroller and a zero crossing detector. Implementation of current and voltage sensors were discussed for power calculations. The communication between the hardware and software interfaces was also discussed in details along with the implementation of the web server using the Django framework and its supporting modules. And lastly it was shown how all the different modules were connected to each other using the microcontroller as a brain and worked together to build our project which is an IoT based Home Automation System.