TECHNICAL UNIVERSITY OF CRETE

DEPARTMENT OF ELECTRICAL

AND COMPUTER ENGINEERING

Energy and Security Management in Homes based on Algorithm for Optimal Control, using an Experimental Setup with Low Cost Arduino/Android Technologies

Kostoglou Stylianos

Thesis Committee Members:

Prof. Kalaitzakis Konstantinos (Supervisor)

Prof. Stavrakakis Georgios (Member)

Dr. Sergaki Eleftheria (Member)

Chania, February 2017

Acknowledgements

I would like to express my sincere gratitude to my Thesis Committee, advisor Prof. Kalaitzakis Konstantinos, Prof. Stavrakakis Georgios, and Dr. Sergaki Eleftheria, for their continuous support of my research, for their patience, motivation, and immense knowledge. Their guidance helped me research and expand my knowledge in the fields of current sensing technologies and Android development, automatic control, optimal control and Fuzzy Logic Control. I would also like to thank Prof. Dollas Apostolos who is my mentor in embedded systems.

Last but not least, I would like to thank my parents and my friends for supporting me throughout my studies.

Abstract

One of the main goals of automation in a smart home is the regulation and control of the home entertainment systems and household activities, in order to achieve the atmospheric comforts (e.g. heating, ventilation, atmosphere humidity) and simultaneous keep the user's desired electric power consumption threshold. Therefore, the aim of this thesis is to present an embedded artificial intelligence control system based on a proposed algorithm and applied to the Arduino platform, for decision support in power flow management. For this, an embedded Arduino system is developed which among other receives the environmental values (e.g. temperature), to be processed in a Fuzzy controller and return the output as a recommendation to control the electric home devices.

To achieve the above goals, a real time control algorithm based on fuzzy logic theory is developed, in order to be implemented on the ARDUINO UNO microcontroller board, using sensors and Android technologies.

At first, the proposed fuzzy algorithm is designed and simulated using MATLAB's Fuzzy Toolkit. Thereafter, this fuzzy algorithm is implemented through ARDUINO UNO tool kit, using the eFLL (Embedded Fuzzy Logic Library) library which is a standard library for Embedded Systems. The eFLL is developed by Robotic Research Group (RRG) at the State University of Piauí (UESPI-Teresina).

The fuzzy algorithm consists of two collaborated fuzzy controllers. The first one has as input: the internal temperature of the home and the external temperature of the environment. The output is the current power level settings of home appliances. The second fuzzy controller has input: the output of the first fuzzy controller, the user's preferences for the daily upper limit of power consumption, the devices' total energy consuption and the data of the internal sensor temperature. The output is the set point value of the heating boiler.

The platform consists of an Arduino based node with network connectivity which is able to exchange data with the developed Android app. The app can be used to monitor sensor data and set the power level settings or deactivate appliances remotely. Sensor data can also be graphically represented on charts or specifically designed Internet of Things gauges on the Android app.

Automations are also built to control energy consumption. The user is able to activate or deactivate specifically programmed automatic modes. Once such modes are activated, appliances will automatically shut down when certain conditions are met, e.g. once the reading of the light sensor reaches a certain point, all the light emitting devices will automatically turn off.

The Android app is built according to the best practices and offers a wide variety of options to the end user. Many other state of the art features are incorporated in the platform, such as voice commands, ajax navigation that contribute to the best possible user experience.

The platform is also open source, modular and can easily integrate any other type of sensor or actuator with minimal or no code changes. Moreover, the industrial design is sized and implemented taking care of the platform cooling.

Περίληψη

Ένας από τους κύριους στόχους των αυτοματισμών σε μία έξυπνη κατοικία είναι η ρύθμιση και ο έλεγχος των συστημάτων οικιακής ψυχαγωγίας και των δραστηριοτήτων της κατοικίας, προκειμένου να επιτευχθούν οι ατμοσφαιρικές ανέσεις (π.χ. θέρμανση, αερισμός, υγρασία της ατμόσφαιρας) και ταυτόχρονα να κρατήσει μέσα στα επιθυμητά όρια του χρήστη την κατανάλωση ηλεκτρικής ενέργειας. Ως εκ τούτου, ο στόχος της παρούσας εργασίας είναι να παρουσιάσει ένα ενσωματωμένο σύστημα τεχνητής νοημοσύνης ελέγχου με βάση έναν προτεινόμενο αλγόριθμο υλοποιημένο στην πλατφόρμα Arduino, για την υποστήριξη λήψης αποφάσεων στον τομέα της διαχείρισης της ροής ισχύος. Για το σκοπό αυτό, έχει αναπτυχθεί ένα ενσωματωμένο σύστημα μέσω του Arduino το οποίο, μεταξύ άλλων, λαμβάνει τις περιβαλλοντικές τιμές (π.χ. θερμοκρασία) οι οποίες υποβάλλονται σε επεξεργασία μέσω ενός ελεγκτή ασαφούς λογικής, ο οποίος με τη σειρά του επιστρέφει την έξοδο σαν σύσταση για τον έλεγχο των ηλεκτρικών συσκευών στην κατοικία.

Για την επίτευξη των παραπάνω στόχων, έχει αναπτυχθεί ένας αλγόριθμος ελέγχου πραγματικού χρόνου που βασίζεται στην ασαφή λογική, προκειμένου να εφαρμοστεί στον μικροελεγκτή Arduino UNO, χρησιμοποιώντας αισθητήρες και τεχνολογίες Android.

Αρχικά ο προτεινόμενος αλγόριθμος ασαφούς λογικής σχεδιάστηκε και προσομοιώθηκε με τη χρήση του εργαλείου Fuzzy toolkit στο Matlab. Στη συνέχεια, ο αλγόριθμος ασαφούς λογικής υλοποιήθηκε στο Arduino UNO,χρησιμοποιώντας την βιβλιοθήκη eFFL (Embedded Fuzzy Logic Library) η οποία αποτελεί μία βασική βιβλιοθήκη των ενσωματωμένων συστημάτων. Η eFFL έχει αναπτυχθεί από την Robotic Research Group (RRG) στο Κρατικό Πανεπιστήμιο της Piauí (UESPI-Teresina).

Ο ασαφής αλγόριθμος αποτελείται από δύο ασαφείς ελεγκτές. Ο πρώτος έχει σαν είσοδο: την εσωτερική θερμοκρασία της κατοικίας και την εξωτερική θερμοκρασία του περιβάλλοντος. Η έξοδος είναι οι τρέχουσες ρυθμίσεις του επιπέδου ισχύος των οικιακών συσκευών. Ο δεύτερος ελεγκτής έχει σαν είσοδο: την έξοδο του πρώτου ασαφούς ελεγκτή, τις προτιμήσεις του χρήστη για το ημερήσιο άνω όριο της κατανάλωσης ενέργειας, την συνολική κατανάλωση ενέργειας των συσκευών της κατοικίας και την εσωτερική θερμοκρασία της κατοικίας. Η έξοδος είναι η τιμή σημείου ρύθμισης του λέβητα θέρμανσης.

Η πλατφόρμα αποτελείται από ένα Arduino βασισμένο σε κόμβους με σύνδεση δικτύου οι οποίοι μπορούν να ανταλάσσουν δεδομένα μέσω της αναπτυγμένης εφαρμογής Android. Η εφαρμογή μπορεί να χρησιμοποιηθεί για την παρακολούθηση των δεδομένων του αισθητήρα και τη ρύθμιση του επιπέδου ισχύος ή την απενεργοποίηση των συσκευών από απόσταση. Τα δεδομένα των αισθητήρων μπορούν επίσης να απεικονιστούν γραφικά σε διαγράμματα ή σε ειδικά σχεδιασμένους IoT μετρητές στην εφαρμογή Android.

Επίσης έχουν σχεδιαστεί αυτοματισμοί για να ελέγχουν την κατανάλωση ενέργειας. Ο χρήστης έχει τη δυνατότητα να ενεργοποιήσει ή να απενεργοποιήσει ειδικά προγραμματισμένες αυτόματες λειτουργίες. Όταν αυτές οι λειτουργίες ενεργοποιηθούν, οι συσκευές θα κλείσουν αυτόματα όταν πληρούνται συγκεκριμένες προϋποθέσεις, π.χ όταν η τιμή του αισθητήρα φωτός γθάνει ένα ορισμένο όριο, όλες οι συσκευές που εκπέμπουν φως θα σβήσουν αυτόματα.

Η Android εφαρμογή αναπτύχθηκε βάσει των βέλτιστων τεχνικών και προσφέρει ποικιλία δυνατοτήτων στο χρήστη. Επίσης στην πλατφόρμα ενσωματώθηκαν και χαρακτηριστικά όπως φωνητικές εντολές, αλλαγή σελίδων μέσω ajax που συμβάλλουν στην βέλτιστη δυνατή εμπειρία χρήσης.

Η πλατφόρμα είναι ανοιχτού κώδικα, επεκτάσιμη και παραμετροποιήσιμη ώστε να μπορεί να δεχθεί οπουδήποτε άλλου είδους αισθητήρα ή ενεργοποιητή, με ελάχιστες ή καθόλου αλλαγές στον κώδικα. Επίσης ο βιομηχανικός σχεδιασμός έχει σχεδιαστεί και τροποποιηθεί σε μέγεθος, έτσι ώστε η πλατφόρμα να έχει την κατάλληλη ψύξη.

Table of Contents

Table of Figures

1. Introduction

1.1 Objectives and Contribution

The notion of buildings as “machines for living in”, as pioneering modernist architect Le Corbusier put in the 1920s, morphs to fit the technologies and issues of the day. In the ‘70s, it was energy efficiency. In the ‘80s, computer technology spawned ‘smart’ buildings sporting automated controls and pre-configured information systems. The latest crop of technologies include microelectromechanical systems that combine sensors and actuators, wireless sensor networks, and fuzzy logic control schemes, and has the makings of a sophisticated nervous system. Home automation is an emerging technology and also a need of today. The main objectives of home automation are controlling, management and co-ordination of home appliances in a comfortable, effective and secure way. Artificial Intelligence (AI) is evolving as a technology for developing automatic systems that can perceive the environment, learn from environment, and can make decision using case based reasoning. Using Vision capability, knowledge based, learn ability, decision making and reasoning the AI provides a better solution for almost all automatic systems. The goal of this Thesis is to develop a home automation systems which can utilize the AI tools in order to increase the effectiveness, powerfulness etc. We propose and create solutions for the following three main issues:

1) How to connect devices in home environment using networking technologies.

2) How to use technologies for communication between home devices using Arduino.

3) How to control and manage home appliances under user’s comfort and finance constraints using AI.

4) How to implement and design an industrial prototype.

A number of well matured and well settled standards, have been already developed and are commercially available solutions for handling the issues 1) and 2). Although, the issue 3) is the more prominent and beneficial feature of the automation system, the issue 3) is not commercial developed as much as the other issues. In the present Thesis, we will concentrate to develop an industrial prototype solution on the third issued of home automation i.e. management of home appliances and particularly, AI based home appliance controlling and

management, in which the system action to be taken depends upon the present and previous environment conditions.

All of the well-known manufacturers offer such solutions that are capable of monitoring power consumption, temperature and other parameters. These readings are easily accessible to the end user, who is able to adjust his usage accordingly. However, although the competition on commercially available Internet of Things products is fierce, the cost to acquire such a system is still relatively high. As a result, the users are still unwilling to invest in current measuring systems and IoT automations. The financial and environmental benefits of such systems are not short term. Furthermore, all these solutions are not easily expandable: Every manufacturer uses different protocols in communication and data exchange. As a result each module is only compatible with a specific vendor and a specific product lineup. Many commercial IoT solutions, also reached their end-of- support date within two years of their debut. Last but not least, the non-open source nature of all commercial IoT products, makes it difficult to troubleshoot and unofficially expand them with custom made modules.

Because of all these above, the goal of the Thesis is to create a low cost, open source, current sensing platform that is easily expandable and modular. Data exchange between the application and the base station must be achieved using publicly documented protocols and APIs. Energy saving automations must also be incorporated into the platform, in order to minimize power consumption and reduce the user's environmental impact.

A system automation that allows the home owner to remotely switch appliances on or off, arm a home security system, control temperature gauges, control lighting, program an entertainment system or home theater, and carry out other related tasks.

1.2 Thesis Outline

The first chapter is an introduction to present the purpose and contribution of this thesis, and furthermore an introduction to the types of home automation systems and how these systems can utilize the AI tools in order to increase the effectiveness.

Chapter two is incorporated to present the hardware and software concepts, methods and the technologies used, in this Thesis in a theoretical base.

Chapter three describes the use of the above technologies in the development of the current sensing platform, the design and the system's architecture for the present Thesis.

Chapter four presents an implementation of an optimal control algorithm using fuzzy logic. Chapter five describes the industrial design of the project and how the platform can be installed.

Chapter six outlines the evaluation of the performance of the built platform.

The last chapter of the Thesis contains useful conclusions we extracted during the development stages of the platform and future improvements and extensions that can be implemented in the present project.

1.3 What is Automated and Networked Home

One definition of an automated and networked home is "An automated and networked home is one in which every appliance can be remotely managed from anywhere on the Internet with a simple Web browser". The general goal of the automatic-home movement is to use networking technology to integrate the devices, appliances and services found in homes so that the entire domestic living space can be controlled centrally or remotely. Home wiring, the advance home developers are installing, typically adds several thousand dollars to the cost of a new home, and it is usually Ethernet or coaxial cable -- or some combination of both -- with other technologies in the mix. The network is being designed to make possible remote operation of appliances connected to the network (Qadeer, December 2012).

1.4 Services Provided by the Home Automation

A home automation system provides a large number of services which can broadly classified into following four categories:

1) lE management of appliances

2) Remote controlling of appliances

3) Efficient utilization of home resources

4) Enhancing home security

Comfortable management includes automatic adjustment of AC (air conditioning) setting, fan regulation setting etc. Remote controlling services include accessing devices from remote location and setting them ON/OFF. Efficient utilization includes running the home appliances at their optimal setting (setting at which we get the required output at minimum cost). Last category of service includes all those services which are used for securing the home environment.

In this Thesis we will see the implementation of all these categories of services one by one and will see the application of AI techniques in implementation (Qadeer, December 2012).

1.5 Artificial Intelligence (AI) and Knowledge Based Systems (KBS)

AI is the collection of powerful and rigorous programming techniques studying the nature of intelligence by building computer systems, and the application of these concepts in solving real-world problems. The growth in the areas of AI has been increased significantly from the last decade. There exist a number of AI tools that make an automation system more sophisticated but here we will discuss the knowledge based systems only as it is used frequently.

Knowledge Based System: A knowledge-based system (KBS) is an AI based system that contains a significant amount of knowledge in an explicit, declarative form. The area of KBS development has matured over the past two decades. It started with first-generation expert systems with a single flat knowledge base and a general reasoning engine, typically built in a rapid-prototyping fashion. This has now been replaced by methodological approaches that have many similarities with general software engineering practice. KBS development is best seen as software engineering for a particular class of application problems. These applications problems typically require some form of reasoning to produce the required results. In current business practice there is an increasing need for such systems, due to progression of information technology in our daily work. For home automation knowledge based systems can provide the base to store the user preferences and managing the home appliances accordingly.

The current research projects focus more on the creation of intelligent home, a home that is able to control and make decision on its own, which follow the guidelines set by occupants directly and indirectly. Due to this trend, the artificial intelligence (AI) is greatly used in the recent smart home research (Qadeer, December 2012).

1.6 Networking Technologies for Connecting Devices in Home Environment

As far as first issue is concerned, a number of standards have been developed for interconnecting the home devices and apparatus in a network so as to make their management much easier and comfortable. Summarized from a following are the main networking technologies used for connecting devices in home environment:

1) Direct cable connection

2) Bluetooth Connection

3) Phone Line

4) Ethernet

5) Radio (Free) Network

6) AC Network

(Qadeer, December 2012)

1.7 Technologies for Communication between Home Devices

There are following leading communication technologies in home environment:

1) UPnP (Universal Plug and Play) devices

2) X-10 based devices

3) Infrared devices

4) Bluetooth Devices

5) IP based devices

(Qadeer, December 2012)

1.8 Home Appliance Controlling and Management

There are two ways to manage and control the home appliances. First, by using traditional digital and microprocessor based systems. Second method for this is by using the sophisticated processing of artificially intelligent agents. If we consider the first choice, then it is somewhat

more common but at the cost of time efficiency as well as feature limitations. On the other hand AI based technique provides more efficient and featured services like easy video and audio processing, easy reasoning etc.

Such class of services is simplest of all types' services and has the structure as shown in the Figure below.

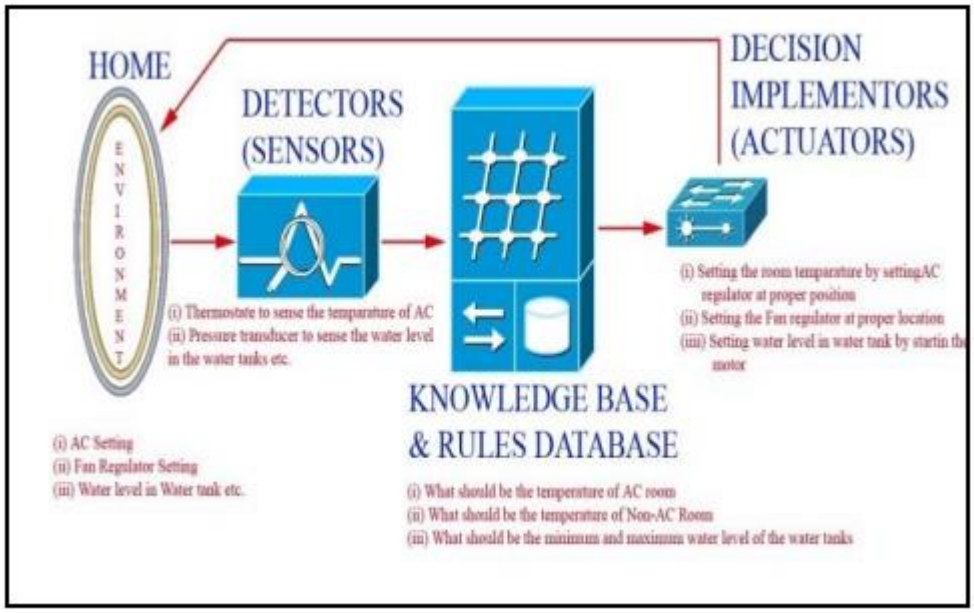

Figure 1-1. Typical structure of home automation system designed for user comfortability

Some features of such systems are:

1) These are closed loop systems

2) Sensors are transducers and other mechanism for sensing the current proximity condition e.g. sensing the room temperature

3) Actuators are simply the mechanism to change the environment according to the control signals received from the knowledge base.

4) Knowledge base (KB) is the centralized part of the system and is the main part to discuss here.

This KB system can be implemented by using simple digital circuitry or using microprocessor systems but both suffer from the problem of manual setting. In both the system user has to decide and change the threshold setting whenever there is a change in the environment.

E.g. in summer AC is used to down the temperature while in winter the same are used to up the temperature. So, user has to change the setting when season changes because the system can't learn from its experience.

But using the KB system having learning capability, system can adjust the threshold setting as it gets the experience in its environment. For this system we need a KB system that can learn from the experience that "what is comfortable temperature" for the home users and can adjust the setting of AC and fan regulator accordingly (Qadeer, December 2012).

1.9 Remote Controlling of Home Appliances

The typical structure of home automation system for this type of services is given in the Figure below.

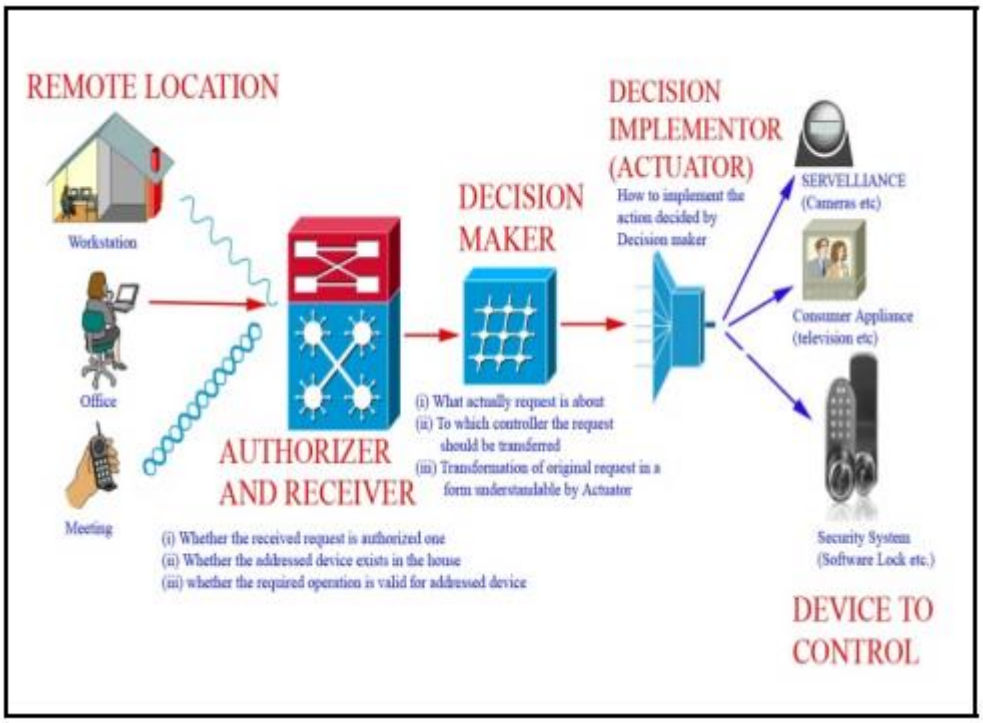

Figure 1-2. Typical structure of home automation system designed for remote management of home devices

Following are main points to consider about this class of systems:

1) Authorizer and Receiver is an electronic system capable to receive the control signal. As discussed in there exist a number of such systems any of them can be used for this purpose.

One important thing about this system is that it requires some authorization mechanism to ensure that the request is authorized one. For this purpose we can use some cryptographic techniques to encode and decode the request so that only authorized user can access the network.

2) The decision maker system is an AI based agent that can decide what action should be taken in response to received query.

E.g. suppose user just put the query that the room temperature should be x^o c. Now, this is the Decision Maker that will identify from its experience that the AC will maintain this temperature. And then it will determine setting of AC regulator corresponding to this particular temperature.

Of course, the same system can be implemented by some electronics circuitry but that will be more complex, less flexible and less featured as compared to this learning based (case based) AI agent system.

3) The third component of this system i.e. Actuator is similar to the action implementer in the previous system. So, we will not discuss anything about this here (Qadeer, December 2012).

1.10 Efficient Control of Home Appliances

The Figure below shows the typical structure of the system that can provide this class of services. The working of the components of the system is as follows:

1) The status poller continuously senses the environment conditions and forwards the sensed condition-factors to analyzer and knowledge based database.

2) The Analyzer receives the current environment condition from poller and with the help of knowledge base database it analyses the variation in the environment. Now, depending upon the variation Analyzer gives the instruction to Actuator.

3) Again here the actuator is similar to the actuator discussed.

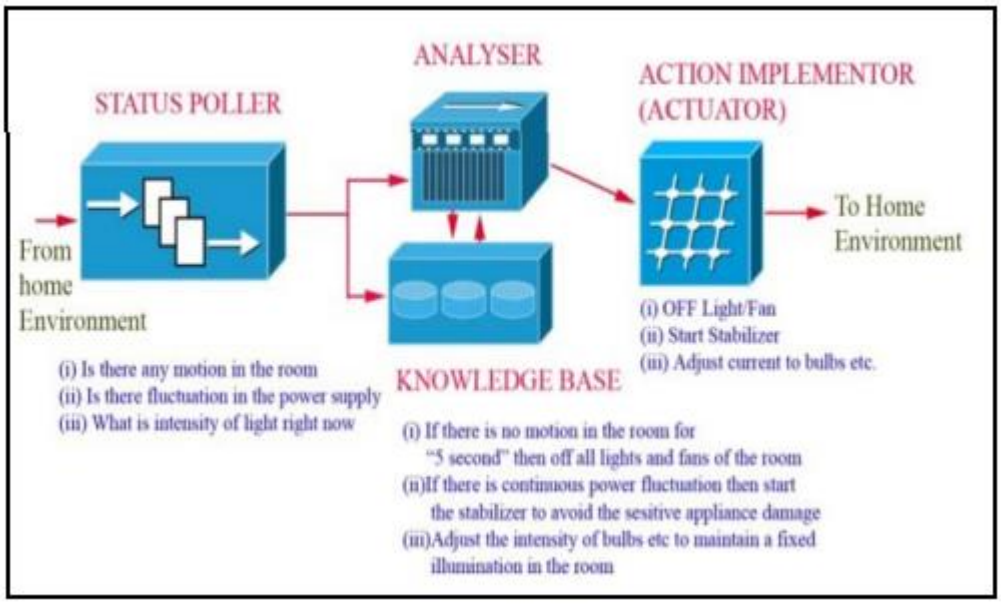

Figure 1-3. Typical structure of home automation system designed for optimizing resource utilization

The major differentiating factor of this system from previously discussed systems is that this is sequential while previous one was atomic, because, in this system the action to be taken depends upon the present and previous environment conditions (Qadeer, December 2012).

1.11 Use of AI in Secure Homes

The description of how system works is as follows:

1) Security sensors sense the environment for security threat.

2) The analyzer and synthesizer analyses the data received from the sensors and filter out any security threat. If it found any security threat, then it sends the sufficient information to reasoning system.

3) The Reasoning system apply the reasoning for detecting whether the security threat is really a threat, if yes then it fires the security alert system for alerting the user about this security threat by analyzer.

4) This analyzer is same as used at the second stage but opposite in nature. The previous converts the physical information into digital form while later one converts the digital information back to physical form to alert the user.

The biggest use of AI is in these systems. Here we can apply following tools of AI for various applications:

1) Video Processing for security threat analysis

2) Image Processing for security threat analysis

3) Audio processing for security threat analysis

4) Knowledge base system for Security system database

5) Case based reasoning for analyzer and synthesizer

6) Decision Making in Security Checking and Decision making

So these systems use the AI exhaustibly. Presently, even though, different technologies are used for implementing these systems but in the near future AI will be the only implementation technique behind these systems as it is proved to be more sophisticated and effective tool for listed applications (Qadeer, December 2012).

1.12 Fuzzy System for Smart Home

Fuzzy Logic (FL) is a problem-solving control system methodology which lends it to implementing in systems starting from small, simple, embedded micro-controllers to large, networked, multi-channel workstation or PC based data acquisition and control systems. FL provides a conclusion based on imprecise, ambiguous, vague, noisy or missing input information.

The main aim of home automation is to provide a convenient and efficient integration and inter-operation within the household appliances. The definition of linguistic rules is a suitable programming method for home automation systems that can be processed by a fuzzy-system. Based on this idea, in 1998 Harald J. Zainzinger developed a neuro-fuzzy controller. The inhabitants are watched by the system at controlling their devices. The home system detects every action related to the correct appliance such as increasing the room temperature by switching on the heater. This event changes the value of the existing control and is kept in a database that is user-specific. This gives the training-data for the neuro-fuzzy network optimization. The system is a hybrid neuro-fuzzy controller which is based on a generic fuzzy-perception. By a reinforcement-

learning algorithm, the system can learn fuzzy rules and sets. The adaptability of this system is the benefit of this implementation. This system also can be used to control other non-time critical applications, and not only limited to smart environment. In 2002, Hani Hagras, Graham Clarke, Martin Colley and Victor Callaghan proposed a smart-home automation system using a fuzzy Incremental Synchronous Learning (ISL) technique. The aim of the system is to provide a non-intrusive, life-long, online method for learning personalized behavior and anticipatory adaptive control for physical environments. The nature of the envisaged target environments are short "initialization" and long "life-long" learning. Generally, a learning process would be non-intrusive within a building. The agent is an augmented behavior based architecture that utilizes a set of parallel Fuzzy Logic Controllers (FLC), each forming a behavior. The behaviors are fixed for an Intelligent Building (IB) which includes the economy, Emergency and Safety behaviors. It can be dynamic and adjustable such as, Comfort behaviors (i.e. Behaviors tailored to inhabitant's real behavior). Each dynamic FLC has a parameter which can be changed with the Rule Base (RB) for each behavior. Each behavior is implemented as a FLC. It uses a singleton fuzzifier, max-product composition, product inference, height defuzzification and triangular membership functions. The system consists of two modes: initialization mode and control mode. During the initialization mode the system needs to monitor the inhabitant's behavior. After the initialization period, the ISL enters to a control mode. In this mode, to guide its control of the room's effectors, the rules learnt during the initialization period are used. Whenever the behavior of the user changes, there is a requirement to delete, add or modify rules in the rule base. ISL suspends environmental control and returns briefly to nonintrusive monitoring cycle during this event. It infers rule base changes essential to find out the new preferences of the users. This is transparent to the user, short cycled and distributed for the duration of the environment use. As such, it forms a life-long learning phase. Compare to other offline method, especially Mendel-Wang ANFIS approach, this method shows comparably better result. However, this has got added benefits adaptation to new users, online and is able to particularize rather than generalize.

In 2005, Faiyaz Doctor and Hani Hagras developed a life-long novel learning approach for intelligent agents which are embedded in intelligent environments. The agents aim

to implement vision of ambient intelligence in an intelligent inhabited environment (IIE). This was accomplished by providing ubiquitous computing intelligence in environment supporting the actions of users. Ambient intelligence is a new inform action paradigm. In this paradigm, people are empowered through a digital environment which is "aware" of their context and presence, and is responsive, sensitive and adaptive to their needs. The proposed technique, which is termed as AOFIS, is an unsupervised data-driven one-pass approach for membership functions and extracting fuzzy rules from data to learn a fuzzy controller which will model behaviors of user. Over a time period, the data is collected by observing the user in the environment. AOFIS system is shown in figure 1-4.

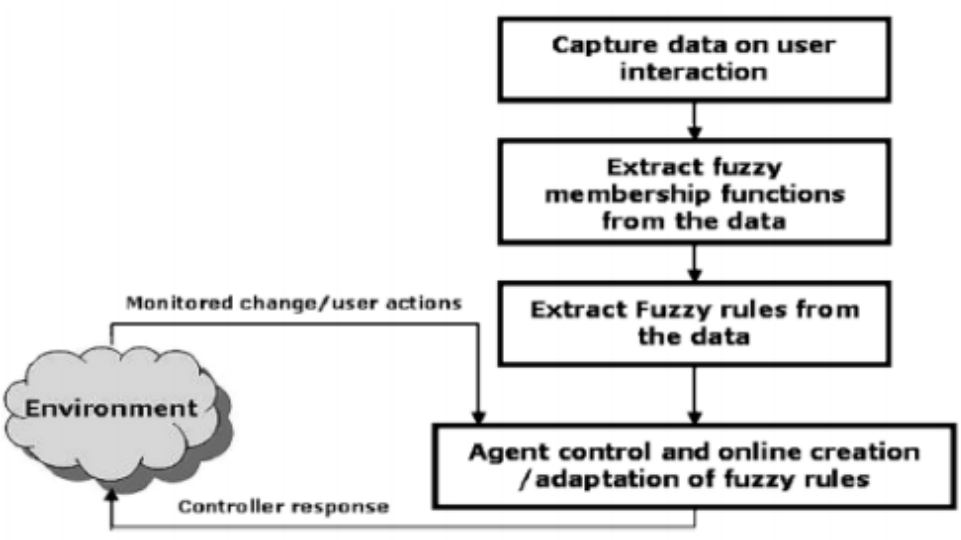

Figure 1-4. A flow diagram of the five phases of AOFIS

The experimental results show that the system learns the user particularized behavior and adjusted online for any changes in life-long learning mode in non-intrusive way. The system is very transparent and therefore, the user was not aware of the intelligently invisible responsive infrastructure of the environment (D, 1994) (Hagras H, 2002) (Mendel J, 1992) (F, 2005).

2. Hardware and Software Background and Methods

2.1 Hardware Development Boards

A hardware development board is a PCB (Printed Circuit Board) that contains a microprocessor and minimal logic circuits. It is often used as a prototyping development platform for applications and products. Unlike a general-purpose system such as a home computer or a single board pc, usually a development board contains little or no hardware dedicated to a user interface. It will only have some provision to accept and run a user-supplied program, such as downloading a program through a serial port (or a USB port and a FTDI chip) to flash memory.

2.1.1 Arduino Board and the ATMega 2560 Microprocessor

Arduino is an open-source project that created microcontroller-based kits for building digital devices that use sensors and control physical devices. The project is based on microcontroller board designs, produced by several vendors, using various microcontrollers (e.g. the ATMega 328). These systems provide sets of both digital and analog input and output (I/O) pins that can interface to various expansion boards (shields) and other circuits. The boards feature serial communication interfaces, including Universal Serial Bus (USB) on some models, for loading programs from personal computers. For programming the microcontrollers, the Arduino project provides an integrated development environment (IDE) based on the programming language processing, which also supports the languages C and C++. The advantages of using an Arduino board for embedded hardware development, include among others, the low production cost, the cross platform enabled Integrated Development Environments and the extensible software and hardware. (Arduino LLC, n.d.)

The Arduino Mega Development Board is based on the ATMega 2560: a high performance, low-power 8-bit microcontroller that combines 256KB of flash memory, 8KB of SRAM, 4KB of EEPROM and 86 GPIO Pins. These pins can be used as inputs and outputs for other electronic circuits. The maximum operating frequency of the crystal is 16Mhz, and the resolution of the ADC (Analog to Digital Converter) is 10bits. The ATMega 2560 offers many usage modes that can be used to either maximize performance or power efficiency. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue

functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset.

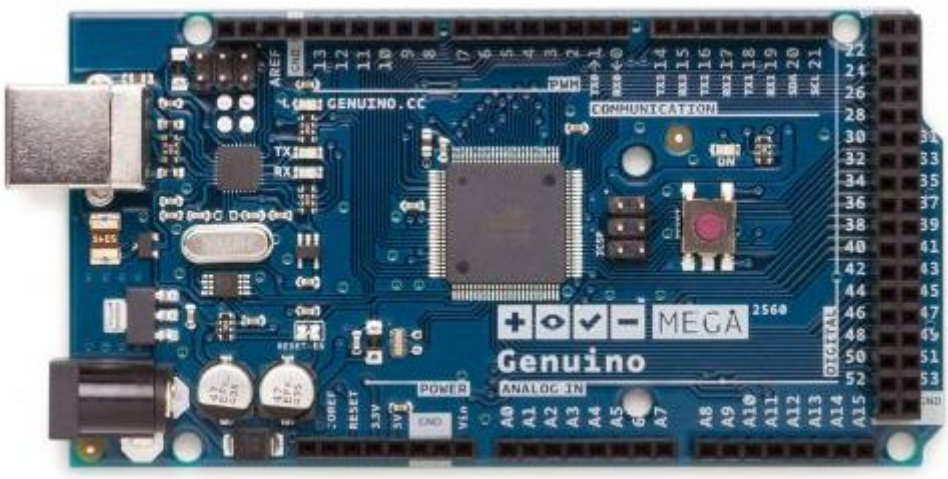

Figure 2-1. The Arduino Mega Development Board

In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run. (Atmel ATMega2560 Datasheet)

2.1.2 Arduino Ethernet Shield and Wiznet W5100 Ethernet Controller

Arduino and Arduino-compatible boards use printed circuit expansion boards called shields, which plug into the normally supplied Arduino pin headers. Shields can provide Ethernet Controllers, motor controls for 3D printing, Global Positioning System (GPS) connectivity, GSM connectivity, or other functionality.

The Arduino Ethernet Shield allows an Arduino board to connect to the Internet or any Local Network. It is based on the Wiznet W5100 ethernet chip. The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP connections. It supports up to four simultaneous socket connections. The Ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows

another shield to be stacked on top. The Ethernet Shield has a standard RJ-45 connection, with an integrated line transformer and PoE (Power over Ethernet) enabled. There is an onboard micro-SD card slot, which can be used to store files for serving over the network. It offers compatibility with all the Arduino boards. The Slave Select Pins (SS) is Pin 4, and must be enabled in order to access the SD Card Slot. The shield also includes a reset controller, to ensure that the W5100 Ethernet module is properly reset on power-up. The current shield has a Power over Ethernet (PoE) module designed to extract power from a conventional twisted pair CAT5 Ethernet cable.

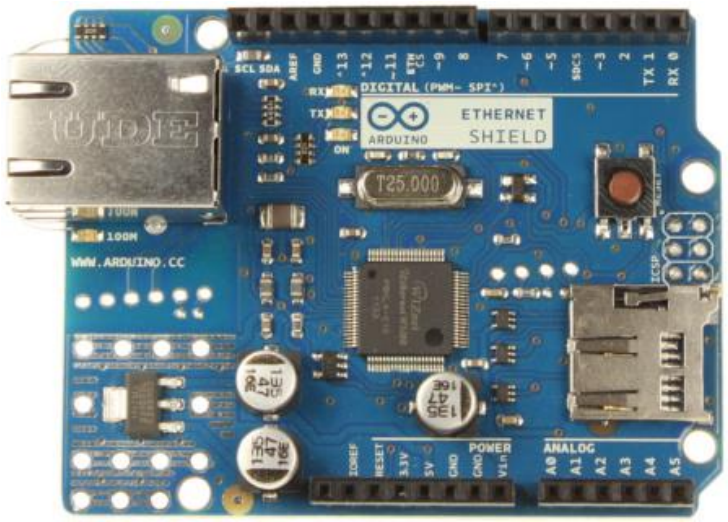

Figure 2-2. Ethernet Shield with the Wiznet5100 Network Controller

Arduino communicates with both the W5100 and SD card using the SPI bus (through the ICSP header). The ICSP headers are Pins 50, 51, and 52 on the Mega. The shield provides a RJ45 Ethernet jack, in order to connect the controller with a network enabled device such as a router or an access point. (Sparkfun, n.d.)

The W5100 includes fully hardwired, market-proven TCP/IP stack and integrated Ethernet MAC & PHY. Hardwired TCP/IP stack supports TCP, UDP, IPv4, ICMP, ARP, IGMP and PPPoE, which has been proven in various applications for several years. 16Kbytes internal buffer is included for data transmission. For easy integration, three different interfaces like memory access way, called direct, indirect bus and SPI, are supported on the MCU side. The Target Applications of the W5100, are, among others, Home Network Devices, Serial-to-Ethernet, and Wireless AP relays. (Wiznet5100 Datasheet)

2.2 Sensors and Actuators

A sensor is a device that detects and responds to some type of input from the physical environment. The specific input could be light, heat, motion, moisture, pressure, or any one of the environmental phenomena. The output is generally a signal (e.g. voltage or current) that is converted to human-readable display at the sensor location or transmitted electronically over a network for reading or further processing.

During the development of the current measuring platform, many parameters were taken into consideration, in order to select the most suitable sensors. Such parameters were: Cost, Sensitivity, Range, Precision, Resolution, Accuracy, Offset, Linearity, Response Time, Mean Time to Failure, Service Costs etc. All the sensor parameters are described below. (National Instruments, N.I., n.d.)

The sensitivity of the sensor is defined as the slope of the output characteristic curve or, more generally, the minimum input of physical parameter that will create a detectable output change. In some sensors, the sensitivity is defined as the input parameter change required to produce a standardized output change. In others, it is defined as an output voltage change for a given change in input parameter. The sensitivity error is a departure from the ideal slope of the characteristic curve.

The range of the sensor is the maximum and minimum values of applied parameter that can be measured. The dynamic range is the total range of the sensor from minimum to maximum.

The concept of precision refers to the degree of reproducibility of a measurement. For example, if exactly the same value were measured a number of times, an ideal sensor would output exactly the same value every time. But in real scenarios, sensors output a range of values distributed in some manner relative to the actual correct value. Some subtle problems arise in the matter of precision when the true value and the sensor's mean value are not within a certain distance of each other.

Resolution is the smallest detectable incremental change of an input parameter that can be detected in the output signal. Resolution can be expressed either as a proportion of the reading or in absolute terms.

The accuracy of the sensor is the maximum difference that will exist between the actual value and the indicated value at the output of the sensor. Again, the accuracy can be expressed either as a percentage of full scale or in absolute terms.

The offset error of a transducer is defined as the output that will exist when it should be zero or, alternatively, the difference between the actual output value and the specified output value under some particular set of conditions. The ideal curve will exist only at one temperature (usually around 25°C), while the actual curve will be between the minimum temperature and maximum temperature limits depending on the temperature of the sample and electrode.

The linearity of the transducer is an expression of the extent to which the actual measured curve of a sensor departs from the ideal curve. The use of a sensor in embedded systems, requires the understanding of the specific conditions that the specification is valid.

The response time of a sensor can be defined as the time required for the sensor output to change from its previous state to a final settled value within a tolerance band of the correct new value: Sensors do not change output state immediately when an input parameter change occurs. Rather, it will change to the new state over a period of time. This term can be defined in a manner similar to that for a capacitor charging through a resistance and is usually less than the response time.

Mean time between failures is the predicted elapsed time between inherent failures of a system during operation. MTBF can be calculated as the arithmetic mean (average) time between failures of a system. The terms is used in both plant and equipment maintenance contexts. The MTBF is typically part of a model that assumes the failed system is immediately repaired (mean time to repair, or MTTR), as a part of a renewal process. This is in contrast to the mean time to failure (MTTF), which measures average time to failures with the modeling assumption that the failed system is not repaired (infinite repair time). The definition of MTBF depends on the definition of what is considered a system failure. For complex, repairable systems, failures are considered to be those out of design conditions which place the system out of service and into a state for repair. Failures which occur that can be left or maintained in an unrepaired condition, and do not place the system out of service, are not considered failures under this definition. In addition, units that are taken down for routine scheduled maintenance or inventory control are not considered within the definition of failure.

An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system. An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure. The supplied main energy source may be electric current, hydraulic fluid pressure, or pneumatic pressure. When the control signal is received, the actuator responds by converting the energy into mechanical motion. An actuator is the mechanism by which a control system acts upon an environment. The control system can be simple (a fixed mechanical or electronic system), software-based (e.g. a printer driver, robot control system), a human, or any other input.

2.2.1 Current Sensor Microbot MR003-009.1

The Current Sensor Module selected for use in the proposed platform is the Microbot MR003-009.1. It carries the Allegro's ACS711ELCTR-12B-T hall effect-based linear current sensor, which offers a low-resistance (~1.2mΩ) current path. (Microbot)

Figure 2-3. The current sensor ACS711 in a breakout board

The sensor operates at 3.3V (up to 5V) and its analog voltage output has a sensitivity of 110mV/A centered at 1.65V (if powered at 3.3V) with a typical error of ±1% and a 100kHz

bandwidth. Optimized bidirectional current range is from -12.5A to +12.5A, but it allows survival of the device at up to 5× overcurrent conditions. (Allegro Microsystems LLC)

2.2.2 Temperature Sensor LM35D

The LM35 is an integrated circuit sensor that can be used to measure temperature with an electrical voltage output proportional to the temperature. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. (Texas Instruments T.I.)

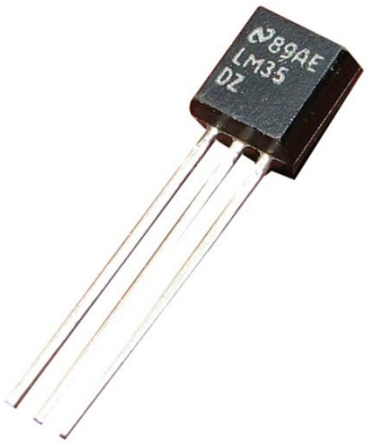

Figure 2-4. The LM35 Temperature Sensor

The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 device draws only 60 µA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 device is rated to operate over a −55°C to 150°C temperature range, while the LM35C device is rated for a −40°C to 110°C range (−10° with improved accuracy). The LM35-series devices are available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D devices are available in the plastic TO-92 transistor package. The LM35 device has a very wide 4 V to 30 V power supply voltage range, which makes it ideal for many applications. In noisy environments, TI recommends adding a 0.1 µF from V+ to GND to bypass the power supply voltage. Larger capacitances maybe required and are dependent on the power-supply noise.

2.2.3 Light Sensor TEMT6000

Photosensors or photodetectors are sensors of light or other electromagnetic energy. A photo detector converts light signals that hit the junction into voltage or current. The connection uses an illumination window with an anti-reflect coating to absorb the light photons. This results in creation of electron-hole pairs in the depletion region. Photodiodes and photo transistors are few examples of photo detectors. Solar cells are also similar to photo detectors as they absorb light and turn it into energy.

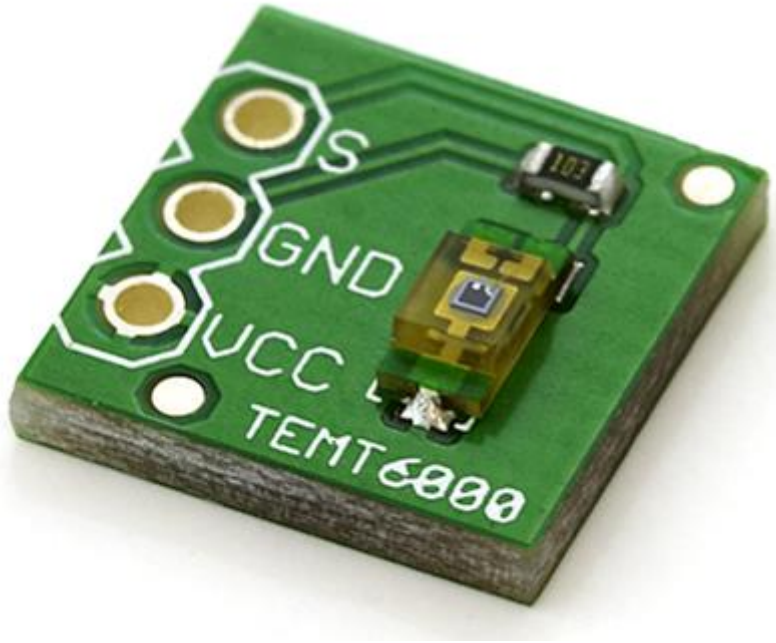

Figure 2-5. The TEMT6000 Light Sensor

The TEMT6000 that is selected for use in the proposed application, is an ambient light sensor made of a silicon NPN phototransistor in a miniature transparent 1206 package for surface mounting. It is sensitive to visible light much like the human eye and has peak sensitivity at 570 nm. (Vishay LLC)

2.2.4 Relay Module

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal.

Figure 2-6. Relay Module

The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations. A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

The Relay Module used in the proposed application contains the Songle SRD-05VDC-SL-C Relay, which is capable of switching high loads up to 10 A at 250 V AC or 10 A at 30 V DC. There is a flyback diode mounted on the selected module in order to eliminate flyback: The sudden voltage spike seen across the inductive load, when its supply current is suddenly reduced or interrupted. The module also contains a LED indicator and headers. (Songle Relay.,ltd)

2.3 Software, Frameworks and APIs

In order to develop the Current Sensing Platform, various Software, Frameworks, Libraries and Software Technologies were used. The Android Application, and Android-related automations were developed with the use of the Android Studio and the MIT App Inventor. The base station software was developed and debugged using the Arduino IDE and the Atmel Studio. The main User Interface of the Android Application was developed using Framework 7 and JQuery Mobile, while the miscellaneous UI elements (Gauges, Charts, Clocks) were developed using Canvas Gauges, Chart JS and CoolClock libraries.

Table 2-1. Software Used for Platform Development

Software used for Android application Development
Android Studio (**https://developer.android.com/studio/index.html**)
MIT App Inventor (**http://appinventor.mit.edu/explore/**)
Software used for Embedded Applications Development and Debugging
Arduino IDE (**https://www.arduino.cc/en/Main/Software**)
Atmel Studio (**http://www.atmel.com/products/microcontrollers/avr/start_now.aspx**)
Software and Frameworks used for UI/UX Development, Charts and Gauges
Framework 7 (**http://framework7.io/**)
Jquery Mobile (**http://demos.jquerymobile.com/1.4.5/**)
ChartJS (**http://www.chartjs.org/**)
Canvas Gauges (**https://canvas-gauges.com/documentation/**)
CoolClock (**http://randomibis.com/coolclock/**)
Miscellaneous Software and Software Technologies used:
Fritzing (**http://fritzing.org/home/**), XML, JSON, Matlab, Matlab Fuzzy Toolkit, FuzzyLib

2.3.1 Android Studio

Android Studio is the official integrated development environment (IDE) for Android platform development. It is based on JetBrains' IntelliJ IDEA software, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

It supports Gradle-based builds, Android-specific refactoring, ProGuard integration and app-signing capabilities. Furthermore, a template-based wizards can be used to create common Android designs and components. Last but not least it includes a rich layout editor that allows users to drag-and-drop UI components, and easily preview layouts on multiple screen configurations.

Applications are usually developed in Java programming language using the Android software development kit. (Google, n.d.)

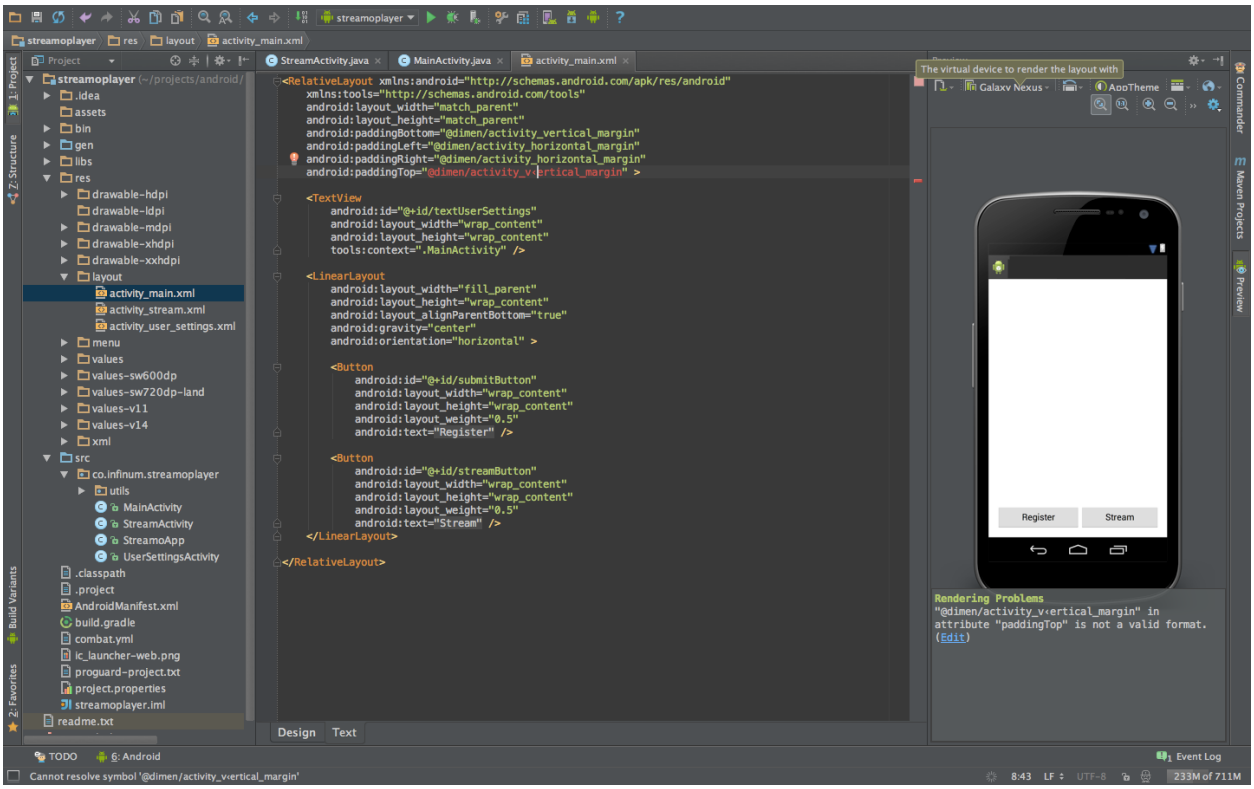

Figure 2-7. Android Studio IDE

2.3.2 MIT App Inventor

App Inventor is a cloud-based tool, which means you can build apps right in your web browser. It lets you develop applications for Android phones using a web browser and either a connected phone or emulator. The App Inventor servers store the work and progress of the development of the Android app.

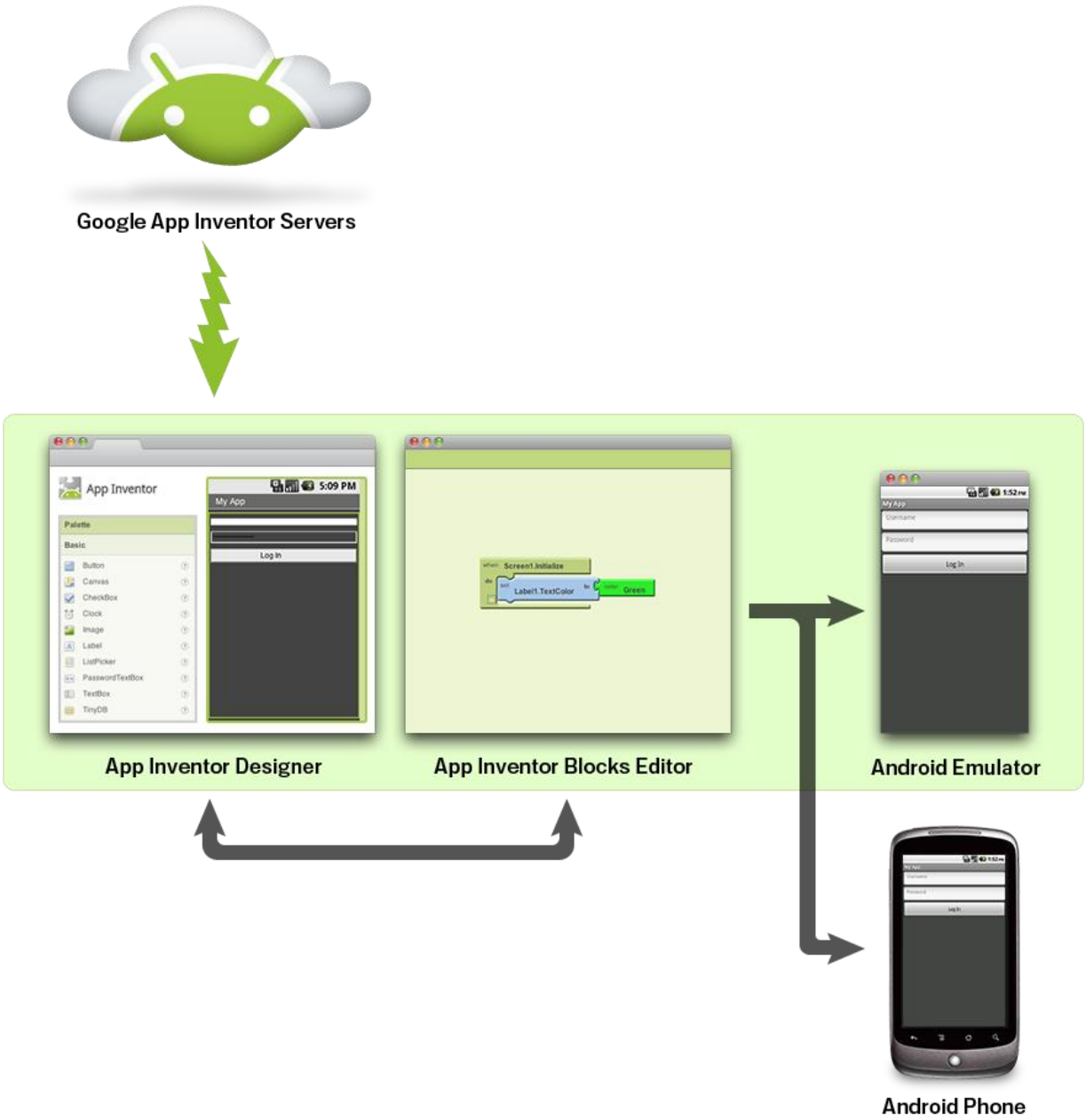

Figure 2-8. AppInventor Block Diagram

The app building components are the following: The App Inventor Designer, where you select the components for your app. The App Inventor Blocks Editor, where you assemble program blocks that specify how the components should behave. You assemble programs visually, fitting pieces together like pieces of a puzzle.

The components appears on the phone step-by-step as pieces are added to it, so it is possible to test the work in real time. After the development of an app is complete, it is possible to package the app and produce a stand-alone application file (apk) to install. The development of an Android App can also be achieved without an Android phone, using the Android emulator, software that runs on the computer and behaves just like the phone. The App Inventor development environment is supported for Mac OS X, GNU/Linux, and Windows operating systems, and several popular Android phone models. Applications created with App Inventor can be installed on any Android phone. An App Inventor Setup package must be installed and configured on the user's computer. (Massachusetts Institute of Technology M.I.T., n.d.)

Below, there are presented some examples with their blocks in MIT App Inventor. In the first example it is shown how the Button component can interact with Label, Checkbox and Textbox components. The second one is a traffic light example which is made using the if-else statement, while the third example repeats a text entered by the user using the for statement.

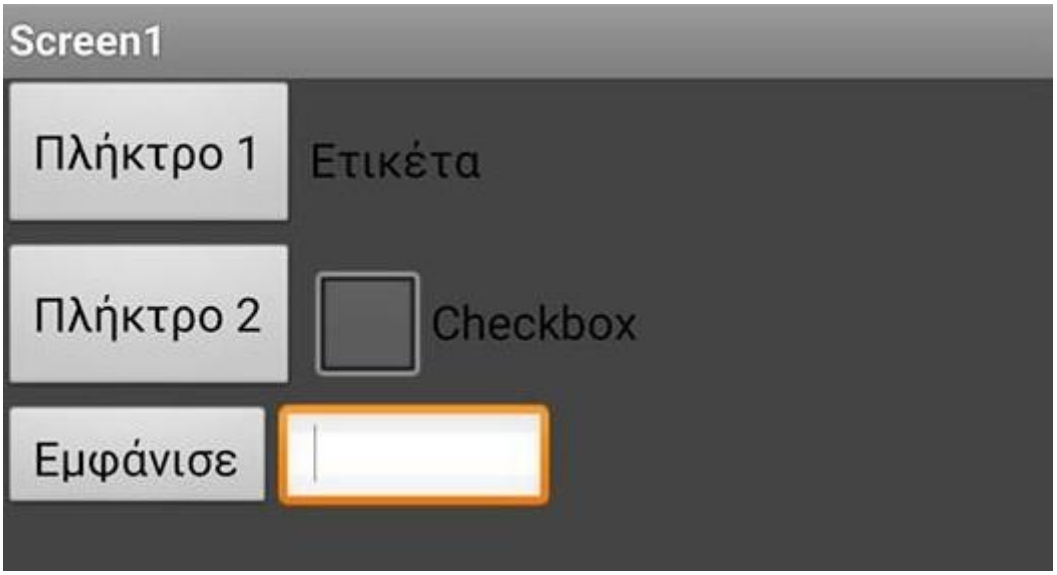

Figure 2-9. mytest1 Start Screen

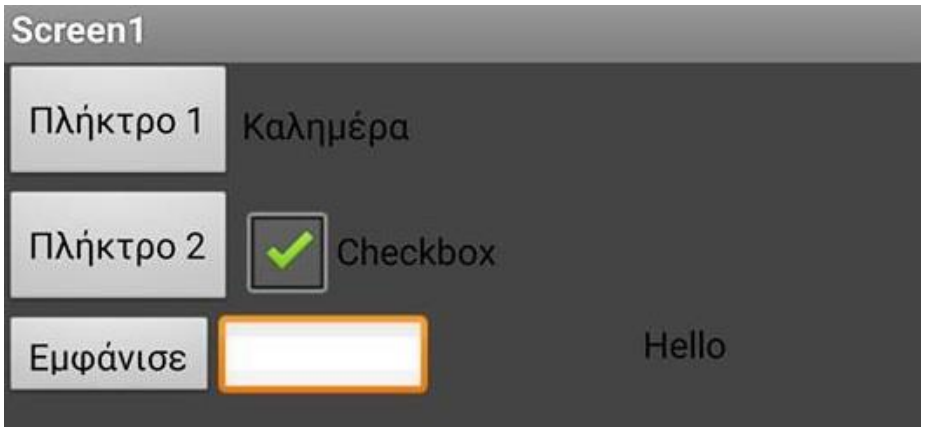

Figure 2-10. mytest1 Results

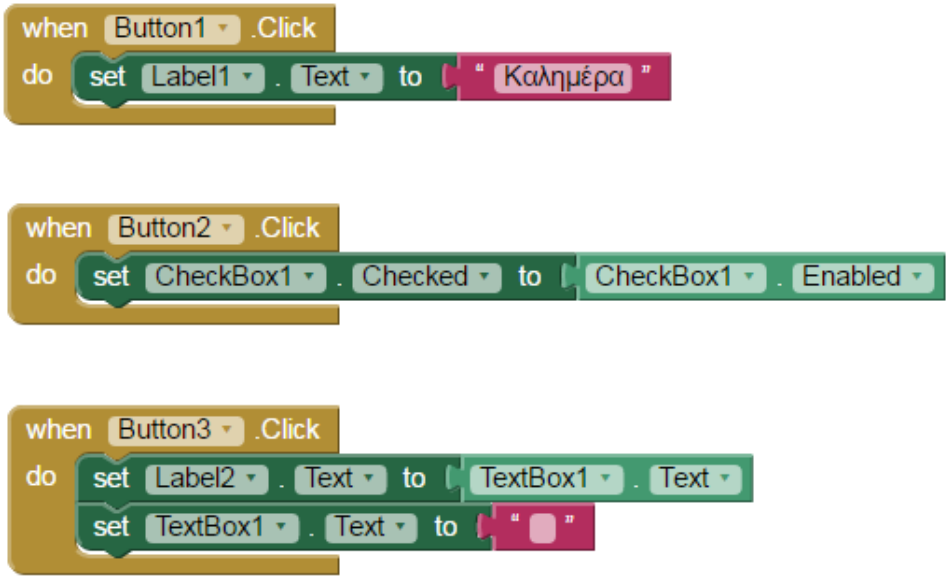

Figure 2-11. mytest1 Blocks

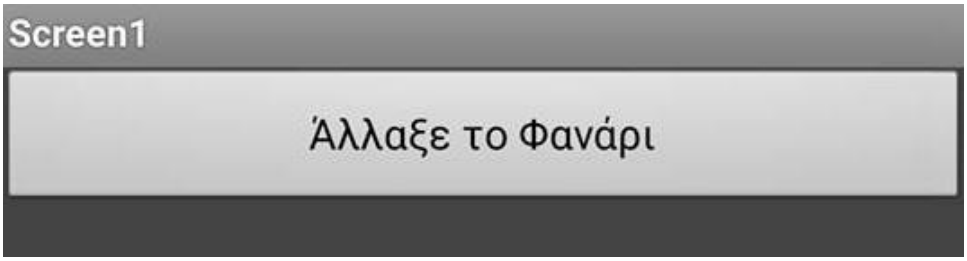

Figure 2-12. mytest2 Start Screen

Figure 2-13. mytest2 Results

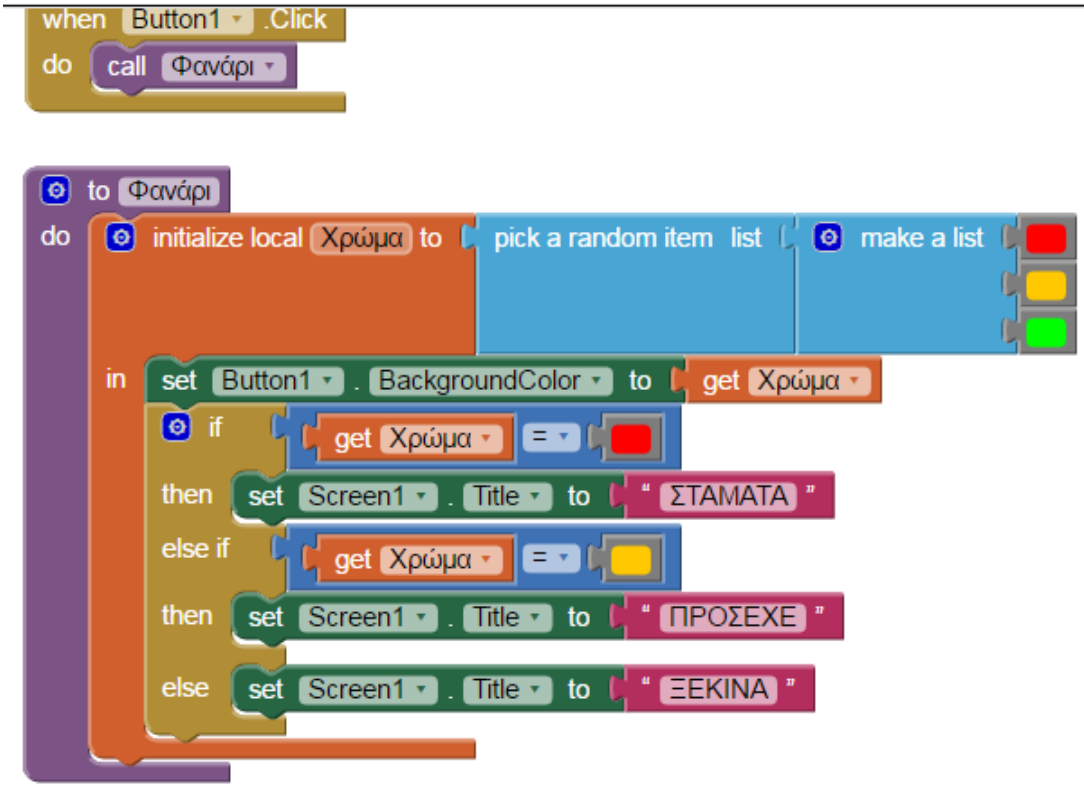

Figure 2-14. mytest2 Blocks

Figure 2-15. mytest3 Start Screen

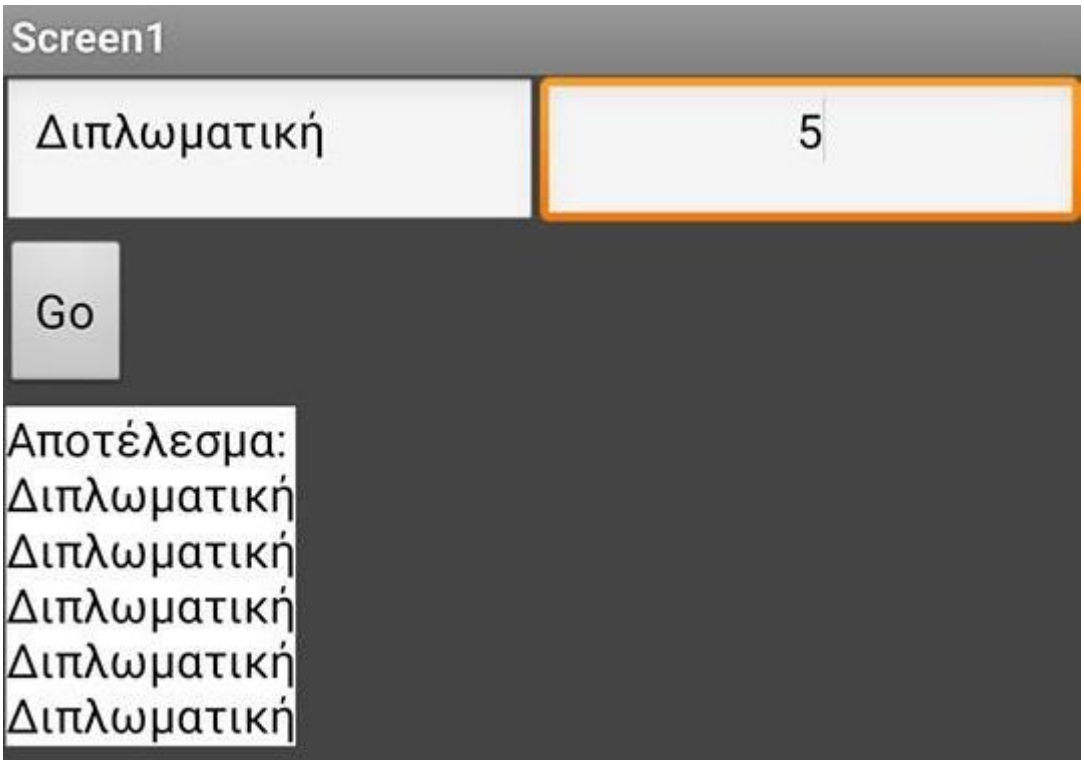

Figure 2-16. mytest3 Results

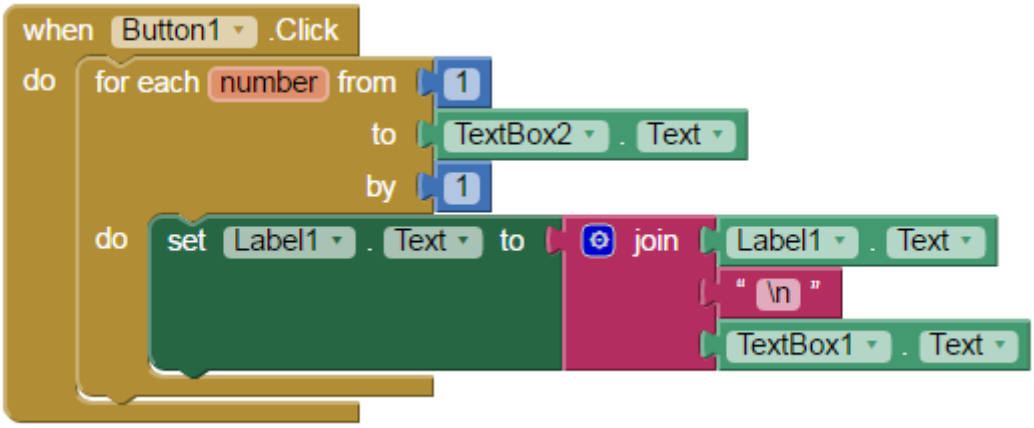

Figure 2-17. mytest3 Blocks

2.3.3 Arduino IDE and Atmel Studio

The Arduino board that uses the ATMega 2560 microcontroller, can be programmed via the official Arduino IDE, and Integrated Development Environment that offers a text editor, a compiler, a debugger, and an abstraction layer for a family of other ATMEL Microcontrollers. The environment is written in Java and based on Processing and other open-source software. The Arduino language is a set of C/C++ functions that can be called from the code. The sketch undergoes minor changes (e.g. automatic generation of function prototypes) and then is passed directly to the avr-g++ compiler. All standard C and C++ constructs are supported by avr-g++ and work in Arduino.

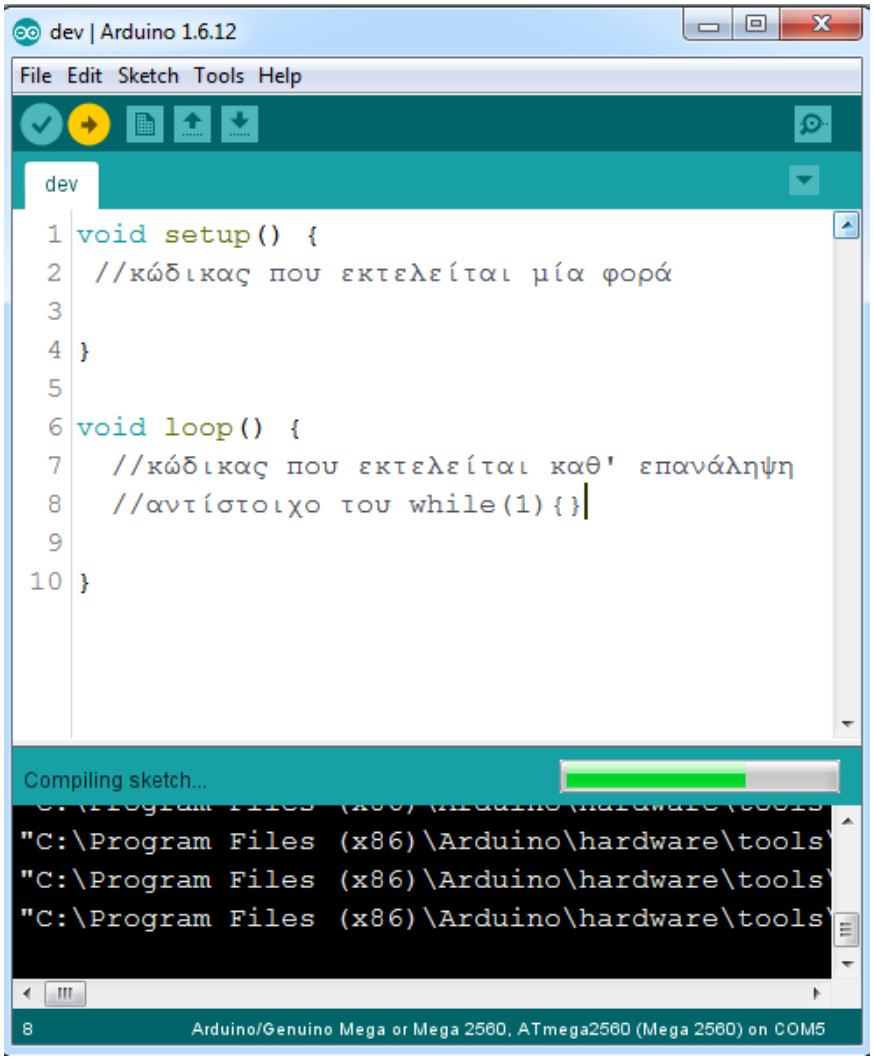

Figure 2-18. Arduino IDE

Atmel Studio is the integrated development environment (IDE) for developing and debugging Atmel ARM-based and Atmel AVR microcontroller (MCU) applications. The Atmel Studio offers a development environment to write, build and debug embedded applications written in C/C++ or assembly code. It also offers connectivity to Atmel debuggers and development kits. All C++ Arduino Projects can be imported, compiled and debugged using the Atmel Studio. (Atmel)

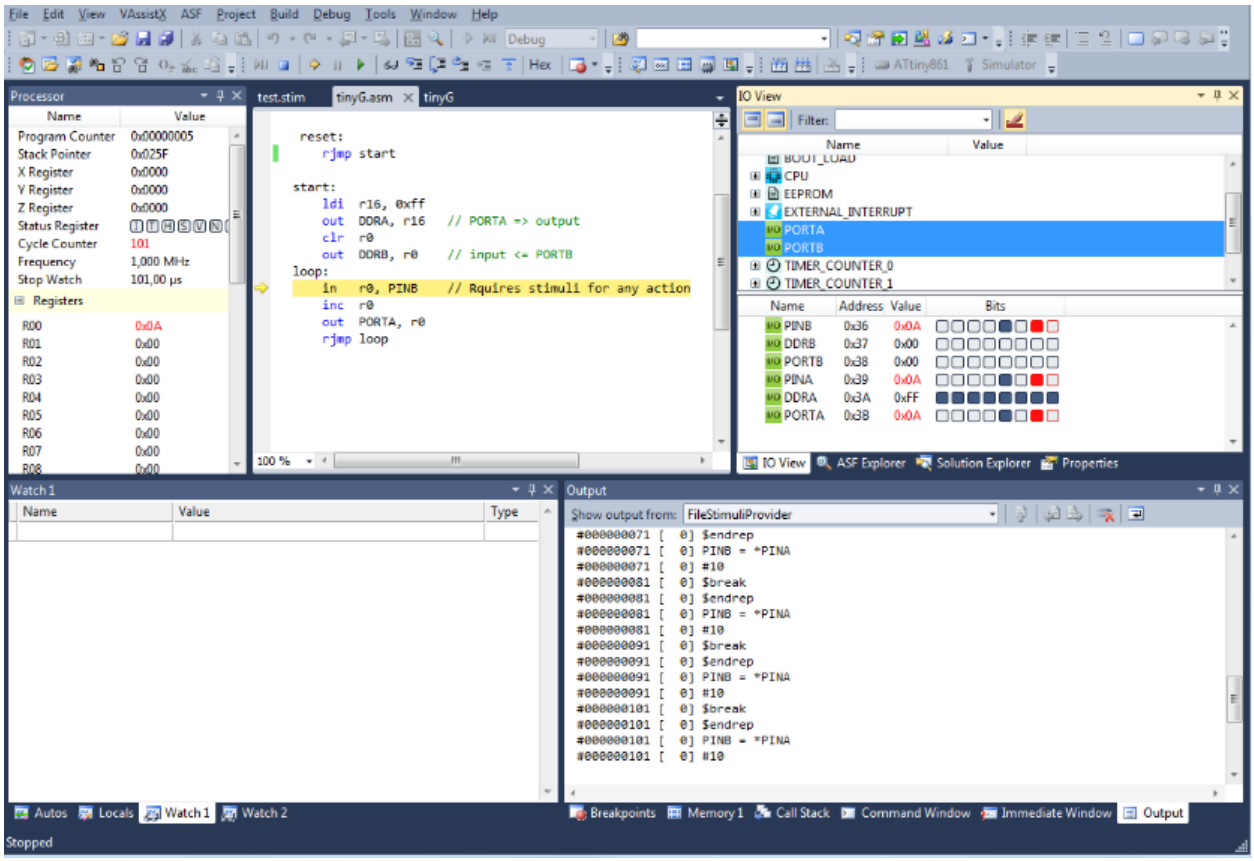

Figure 2-19. Atmel Studio IDE

2.3.4 UI Framework 7

Framework7 is an open source mobile HTML framework to develop hybrid mobile apps or web apps with Android & iOS native look. The main approach of the Framework7 is to give the user the ability to create iOS & Android apps with HTML5, CSS and JavaScript. Framework7 Material Android theme was designed according to the official Google Material design specification and guidelines. Framework7 supports an unlimited number of different isolated views, and isn't dependent on any third party libraries. (Framework7)

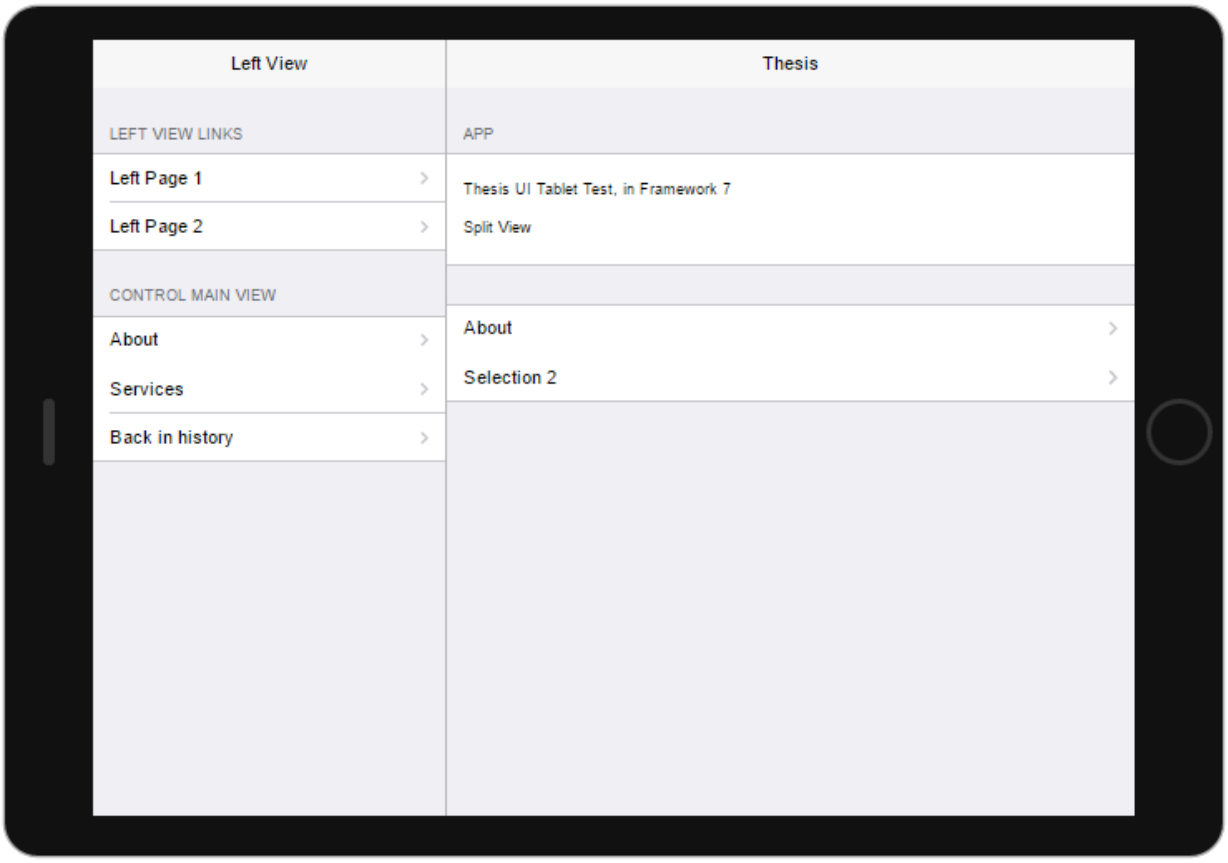

Figure 2-20. Framework 7 Split View

The classes panel-right and panel-left can be used to split the UI elements in two categories, in order to create a tablet Friendly UI, as show in Figure 2-20. Framework 7 Split View.

```
<div class="panel panel-left panel-reveal">
<div class="content-block">
<p>Left panel content</p>
</div> </div>
```

2.3.5 UI Framework JQuery Mobile

JQuery Mobile is a HTML5-based user interface system designed to make responsive mobile apps that are accessible on all smartphone, tablet and desktop devices. It is built on the rock-solid jQuery and jQuery UI foundation, and offers Ajax navigation with page transitions, touch events, and various widgets. Its lightweight code is built with progressive enhancement, and has a flexible, easily themeable design. jQuery Mobile is a project of the jQuery Foundation. It is completely open source and is one of the most popular Mobile UI/Theming frameworks.

A page in jQuery Mobile consists of an element with a data-role="page" attribute. Within the "page" container, any valid HTML markup can be used, but for typical pages in jQuery Mobile, the immediate children of a "page" are divs with data-role="header", class="ui-content", and data-role="footer". The baseline requirement for a page is only the page wrapper to support the navigation system, the rest is optional. A page can be styled as a dialog that makes the page look like it's a modal style overlay. To give a standard page the appearance of a modal dialog, add the data-rel="dialog" attribute to the link. Transitions can also be set on dialog links. (The jQuery Foundation, n.d.)

Inside the content container, any standard HTML elements can be added- headings, lists, paragraphs, etc. Custom styles can also be created by adding an additional stylesheet to the head after the jQuery Mobile stylesheet. jQuery Mobile includes a wide range of touch-friendly UI widgets: form elements, collapsibles, collapsible sets (accordions), popups, dialogs, and responsive tables. JQuery Mobile also includes a diverse set of common listviews that are coded as lists with a data-role="listview" added.

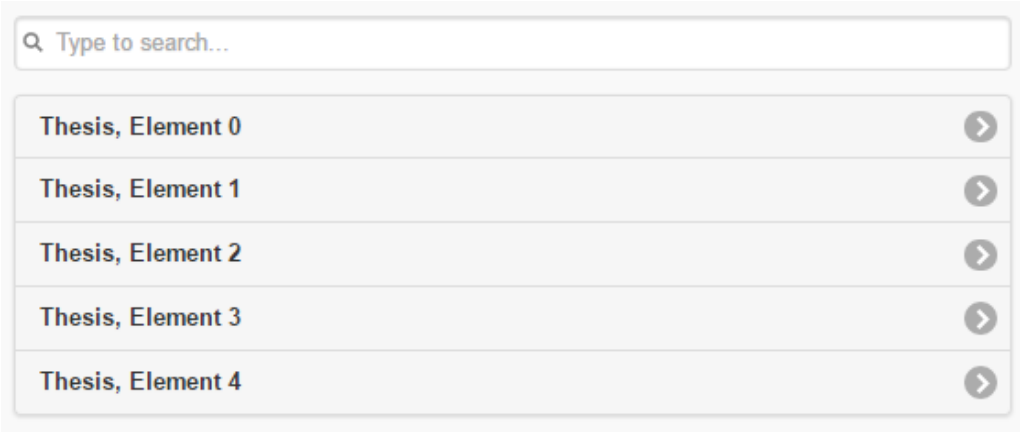

Figure 2-21. JQuery Mobile Listview

The code below creates the list of elements presented in the Figure above.

```
<ul data-role="listview" data-inset="true">
<li>Thesis, Element 1</a></li>
<li>Thesis, Element 2</a></li>
<li>Thesis, Element 3</a></li>
<li>Thesis, Element 4</a></li>
<li>Thesis, Element 5</a></li>
</ul>
```

The framework also contains a full set of form elements that are automatically enhanced into touch-friendly styled widgets. The image below shows a slider made with the new HTML5 input type of range, and miscellaneous elements.

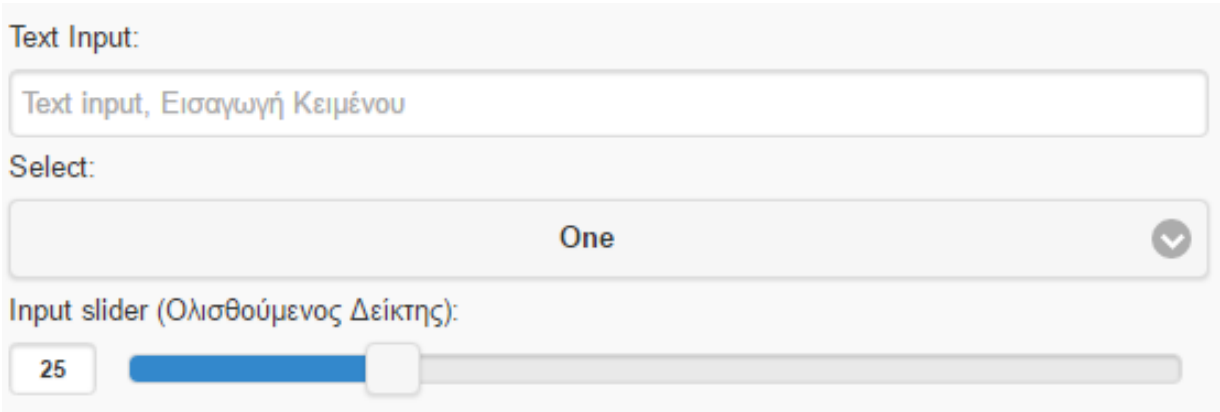

Figure 2-22. JQuery Mobile Slider, Text Input, and Button Selector

The following code creates the list of elements presented in the Figure above.**Error! Reference ource not found.**

```
<form>
    <label for="textinput-s">Text Input:</label>
    <input type="text" name="textinput-s" id="textinput-s"
placeholder="Text input, Εισαγωγή Κειμένου">
    <label for="select-native-s">Select:</label>
    <select name="select-native-s" id="select-native-s">
        <option value="small">One</option>
        <option value="medium">Two</option>
    </select>
    <label for="slider-s">Input slider (Ολισθούμενος Δείκτης):</label>
    <input type="range" name="slider-s" id="slider-s"
    value="25" min="0" max="100" data-highlight="true">
</form>
```

2.3.6 Chart JS

Charts are often used to ease understanding of large quantities of data and the relationships between parts of the data. Charts can usually be read more quickly than the raw data that they are produced from. ChartJS is a HTML5 library that uses the canvas element and can be used in order to graphically represent data. To import the chart js libary the following element must be added to the head tag of an HTML file. (ChartJS, n.d.)

```
<script src="Chart.js"></script>
```

To create a chart, we need to instantiate the Chart class. To do this, we need to pass in the node, jQuery instance, or 2d context of the canvas of where we want to draw the chart.

```
<canvas id="myChart" width="400" height="400"></canvas>
var chartInstance = new Chart(ctx, {
    type: 'line',
    data: data,
    options: {
        responsive: true
    }
});
```

The line chart usually requires an array of labels. This labels are shown on the X axis. There must be one label for each data point. More labels than datapoints are allowed, in which case the line ends at the last data point. The data for line charts is broken up into an array of datasets. Each dataset has a color for the fill, a color for the line and colors for the points and strokes of the points.

These colours are strings just like CSS. RGBA, RGB, HEX or HSL notation can be used. The label key on each dataset is optional, and can be used when generating a scale for the chart. When spanGaps is set to true, the gaps between points in sparse datasets are filled in. The data passed to the chart can be passed in two formats. The most common method is to pass the data array as an array of numbers. In this case, the data.labels array must be specified and must contain a label for each point or, in the case of labels to be displayed over multiple lines an array of labels (one for each line) i.e [["June","2015"], "July"].

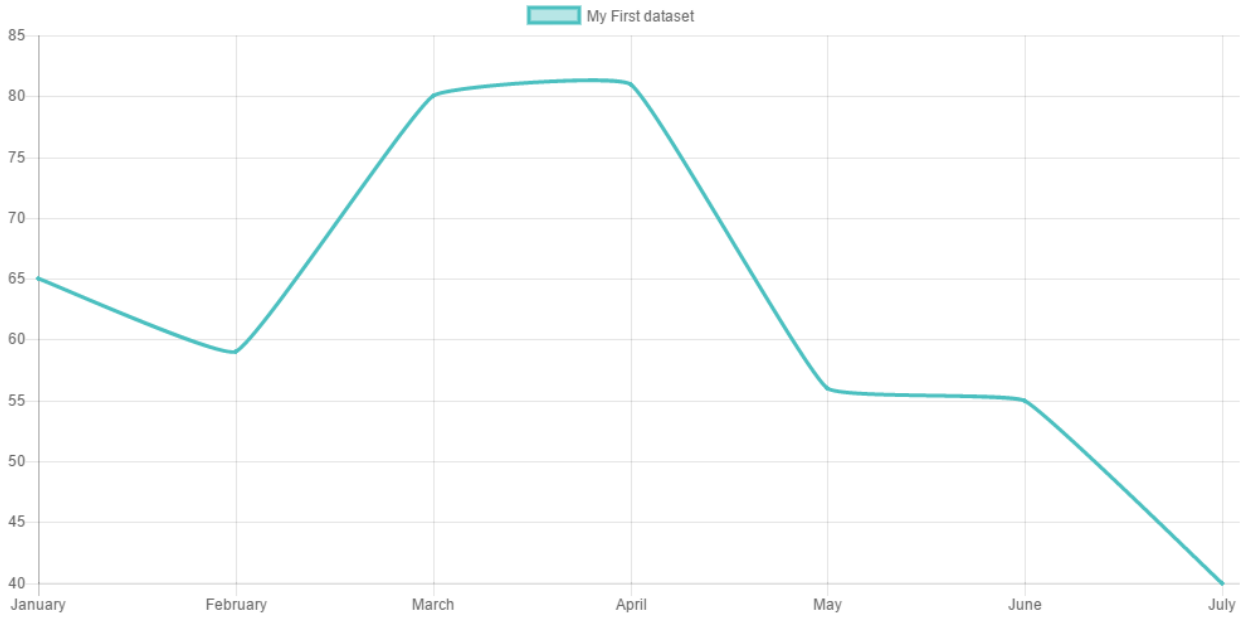

Figure 2-23. Sample Chart created with the ChartJS library

The code to create the above chart is in the Index.

2.3.7 Canvas Gauges

Canvas gauges are highly configurable UI components. There are plenty of options which could help build a unique pretty gauge for mobile apps or web pages. Configuration options for the gauge usually passed to a constructor or update functions and are a plain JavaScript object or can be specified as attributes of HTML-elements. Canvas gauges are friendly to minimalist code design, so whenever you need these gauges to use on a desktop, mobile or IoT devices with limited resources, it will provide the best options to get the minimum amount of code for the solution. Canvas gauges can be simply installed using npm package manager. Depending on the needs of the application, there is a possibility to install whole gauge library or only that part you really need for your project. If you only need the exact type of the gauge it can be installed using the appropriate npm tag. (Canvas Gauges, n.d.) Currently the following gauges are supported: linear and radial. This strategy is useful only if you need to minimize the code base and plan to use only a specific gauge type. If various gauge types are needed in the project, it is recommended to use whole gauge package. There are two ways of using gauges on the page. First one is declarative by simply defining a gauge components in HTML. Another way is to use the scripting API to inject gauges to the page. Some gauge examples with their code are presented below.

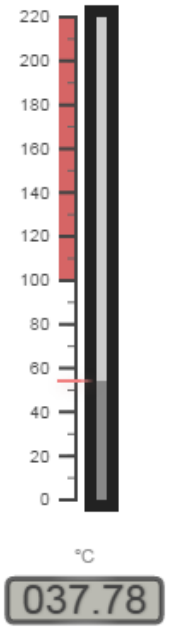

Figure 2-24. Linear Temperature Gauge

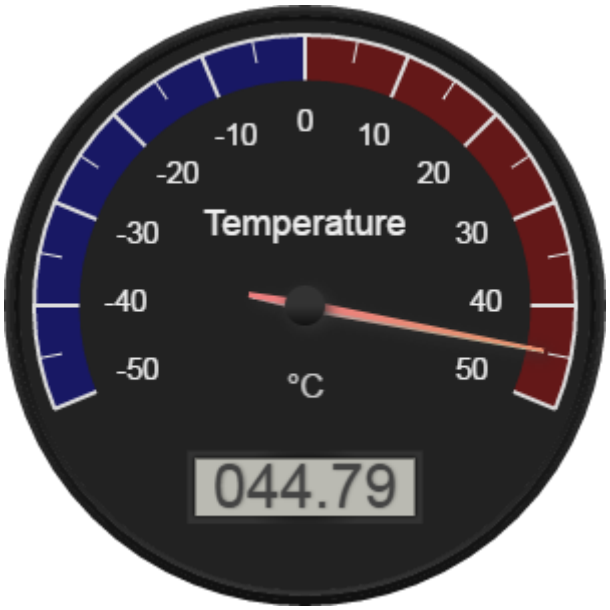

Figure 2-25. Radial Temperature Gauge

The code to create these gauges is placed in the Index.

2.3.8 CoolClock

CoolClock is a customizable javascript analog clock. CoolClock requires canvas support therefore it works best in Firefox, Safari or Chrome. It can work in IE via the use of ExplorerCanvas however in IE it refreshes slowly, doesn't render as nicely and the second hand decoration is disabled due to a rendering glitch. CoolClock does not use Flash. In order to use CoolClock it is necessary to download coolclock.js, moreskins.js and excanvas.js and put them in the same folder as the html file. (Coolckock, n.d.) In the head section of the html file, the following lines of code must be added:

```
<script type="text/javascript" src="excanvas.js"></script>

<script type="text/javascript" src="coolclock.js"></script>

<script type="text/javascript" src="moreskins.js"></script>
```

If a page doesn't use jQuery then it is necessary to add the following line of code to the body tag:

```
<body onload="CoolClock.findAndCreateClocks()">
```

Somewhere in the body of the html file add the following:

```
<canvas id="clockid"
class="CoolClock:Skin:Radius:noSeconds:GMTOffset"></canvas>
```

Omitting the fields will have as a result to load the default values. Additionally, there is the option to add a real css class to the clock canvases. In order to do this just add a space and then the class. Of course styles can be added directly if needed. The id can be anything but it should be unique of course. Below there is a CoolClock example and its code.

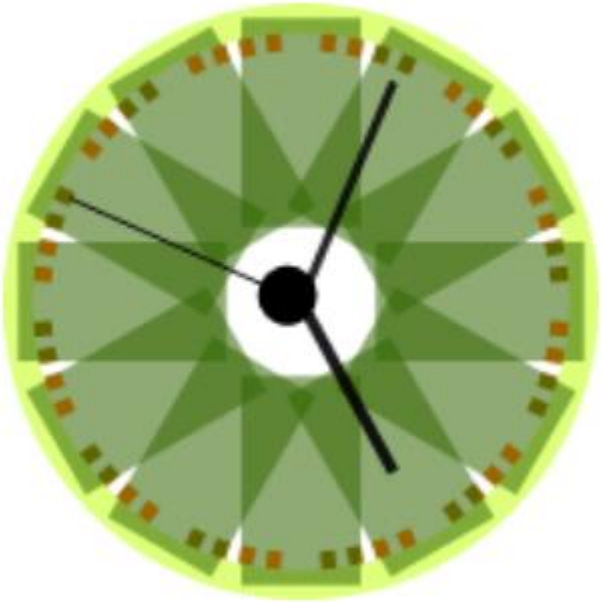

Figure 2-26. CoolClock

```
<canvas id="s03" class="CoolClock:Lev" width="170" height="170"
style="width: 170px; height: 170px;"></canvas>
```

2.3.9 JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write and for machines to parse and generate. It is based on a subset of the JavaScript Programming Language. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures: A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array, and an ordered list of values, which, in most languages, this is realized as an array, vector, list, or sequence. These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures. (JSON org, n.d.)

An object is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right brace). Each name is followed by : (colon) and the name/value pairs are separated by , (comma).

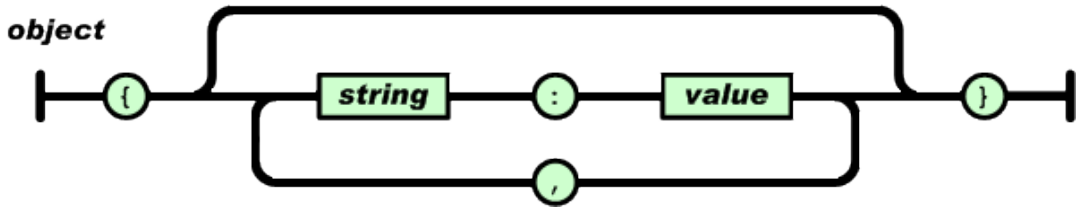

Figure 2-27. JSON objects

An array is an ordered collection of values. An array begins with [(left bracket) and ends with] (right bracket). Values are separated by , (comma).

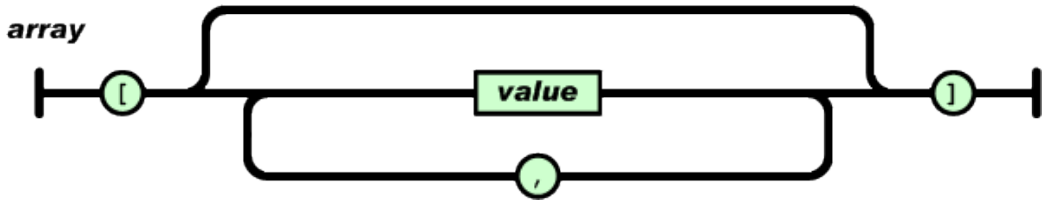

Figure 2-28. JSON Array

A value can be a string in double quotes, or a number, or true or false or null, or an object or an array. These structures can be nested.

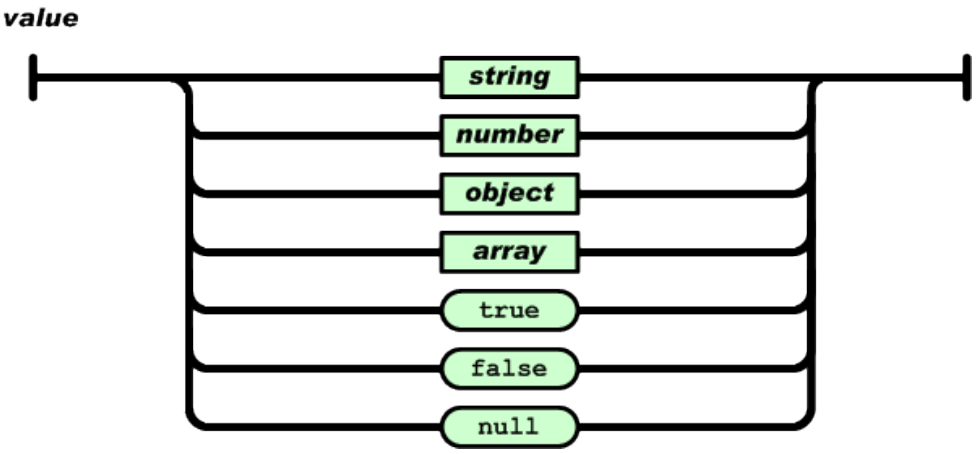

Figure 2-29. JSON value

Data presented in JSON format, are presented below

```
[
    {
        color: "red",
        value: "#f00"
    },
    {
        color: "green",
        value: "#0f0"
    },
    {
        color: "yellow",
        value: "#ff0"
    },
    {
        color: "black",
        value: "#000"
    }
]
```

2.3.10 XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The design goals of XML emphasize simplicity, generality, and usability. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services. (Goldberg, 2009)

Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.

An element in an XML file, is a logical document component that either begins with a start-tag and ends with a matching end-tag or consists only of an empty-element tag. The characters between the start-tag and end-tag, if any, are the element's content, and may contain markup, including other elements, which are called child elements. An example is <tag>reading</tag>.

The data structure of a sample XML File, is presented below.

```
<breakfast-menu>
    <food>
        <name>Waffles</name>
        <price>6,0</price>
        <calories>650</calories>
    </food>
    <food>
        <name>Biscuits</name>
        <price>8,5</price>
        <calories>900</calories>
    </food>
    <food>
        <name>Toast</name>
        <price>4,0</price>
        <calories>450</calories>
    </food>
</breakfast-menu>
```

3. Current Sensing Platform Development

3.1 Base station development

3.1.1 Arduino platform development

The current sensing platform runs on the Arduino Mega, on top of which an Ethernet Controller Shield is connected. Three sensors are connected, a current sensor, a temperature sensor and a light sensor.

The Current is sensed with the use of Allegro's ACS711ELCTR-12B-T hall effect-based linear current sensor, the output of which is connected to Arduinos Analog Input 0. The output of the LM35 temperature sensor is connected to Arduinos Analog Input 1. The output of the light sensor TEMT6000 is connected to the Arduinos Analog Input 2. All relays are powered by the on board voltage regulator of the Arduino, 5V DC. Their state is controlled via Output Pins 2-5. The output of the power supply of the LED strips, is 12V DC, 0,5A.

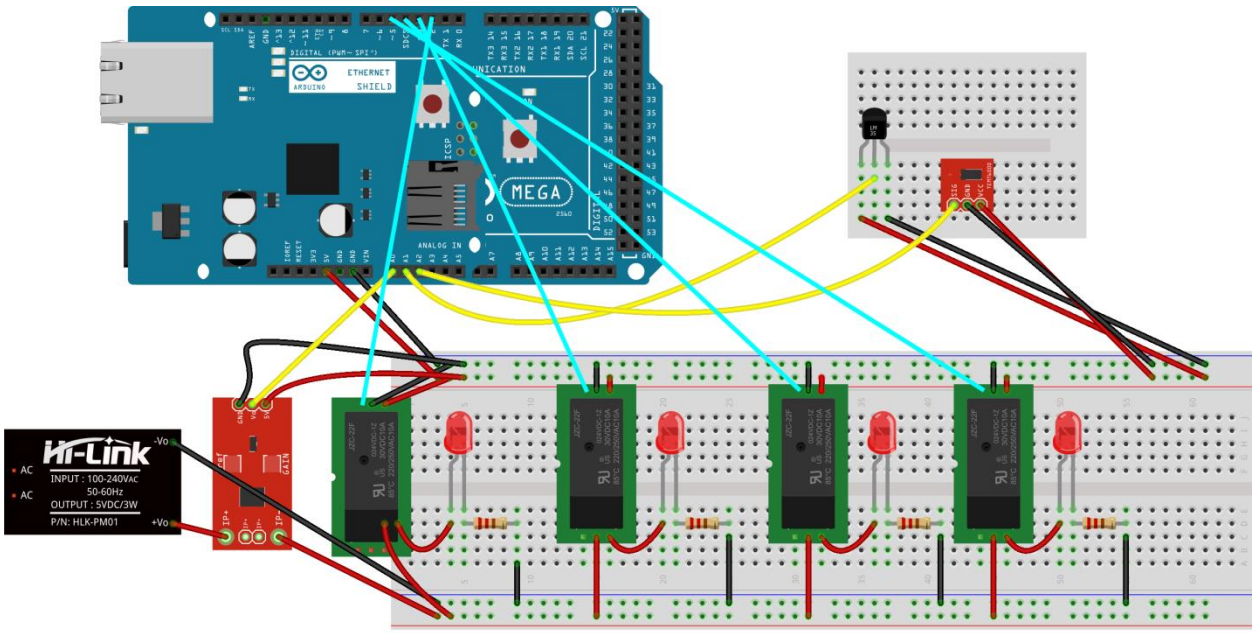

Figure 3-1. Arduino based current sensing platform

In the proposed Arduino Code, the Ethernet Shield and the Arduino board are used to create a simple Web server. Using the Ethernet library, the device is able to answer a HTTP request with the Ethernet shield. After opening a browser and navigating to the Ethernet shield's IP address, the Arduino will respond with a XML / JSON output of the readings of the connected sensors. The code which creates a basic Web Server which contains sensor readings, is presented below.

```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = {0x20, 0x20, 0x20, 0x30, 0x30, 0x30 };
IPAddress ip(192, 168, 1, 120);
EthernetServer server(80);
String readString;

void setup()
{
  pinMode(A0, INPUT);
  Ethernet.begin(mac, ip);
  server.begin();
  //Serial.begin(9600); //debug
}

void loop()
{
  EthernetClient client = server.available();
  if (client)
  {
    while (client.connected())
    {
      if (client.available())

      {
        char c = client.read();

        if (readString.length() < 100)

        {
          readString += c;
```

```
            Serial.write(c);
            if (c == '\n') {

              client.println("HTTP/1.1 200 OK");
              client.println("Server: Arduino");
              client.println("Access-Control-Allow-Origin: *");
              client.println("Connection: close");
              client.println("Content-Type: text/xml\r\n");
              client.println("<xml>");
              client.println("<sensorvalue>");
              int reading=analogRead(A0);
              client.println(reading);
              client.println("</sensorvalue>");
              client.println("</xml>");

              delay(1);

              client.stop();

              readString = "";
              delay(1);
              client.stop();

            }
          }
        }
      }
    }
}
```

The Arduino Web Server can also be programmed to output a JSON formatted file with the readings of the sensors. In this case the content type must be declared as "application/json". The following code snippet displays all the readings from the analog Pins A0-15 of the Arduino Mega, as a JSON-formatted file.

```
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: application/json");
client.println("Connection: close");
client.println("Access-Control-Allow-Origin: *");
client.println();
client.print("{\"analogReadings\":[");

for (int analogChannel = 0; analogChannel < 16; analogChannel++)
{
  if(analogChannel>0) client.print(',');
  int sensorReading = analogRead(analogChannel);
  client.print(sensorReading);
}

client.println("]}");
```

3.1.2 Arduino sensor readings, conversion and calibration

Each analog output of the connected sensors to the proposed platform, is read by Arduinos analog-to-digital converter, and converted to a digital number. The ADC has a resolution of 10bits, which equals to 1024 separate values for a signal from 0V to 5V. The output of the ADC should then be converted to a S.I. unit, using the datasheet calculation correction. The following functions, converts the analog readings from all sensors used in the proposed platform, to SI measurements. Digital filters and statistical methods of reducing errors in readings were also used.

```
[...]
int calculateAmperage() {
  long sum = 0;
  for (int i = 0; i <= 100; i++) {
    int reading = analogRead(A0);
    sum += reading;
  }
  int mo = sum / 101;
  Serial.println(mo);
  int amperagema = (mo - 501) * 13;
  //Serial.println(amperagema);
  return amperagema;
}

int calculateWattage(int amps) {
  int ipologismos = amps * 12;
  return ipologismos;
}

int calculateTemperature() {
  long sum = 0;
  for (int i = 0; i <= 10; i++) {
    int thermokrasia = (5.0 * analogRead(A1) * 100.0) / 1024;
    sum += thermokrasia;
  }
  int mo = sum / 11;
  return mo;
}
```

```
int calculateLux() {
  int illuminance = analogRead(A2) * 0.97;  //fwteinotita
  //Serial.println(illuminance);
  return illuminance;

}

long calculatemwh(int mw) {
  sumwh += mw * 5;
  Serial.println(sumwh);
  return sumwh;
}
[...]
```

3.1.3 Arduino Actuators and Relay Control

All high current devices connected to the proposed platform can be controlled manually via the Android app or a web browser. Relays are used to control the state of high current devices such as vacuum cleaners or heaters. A low current signal sent by the Arduino controls the electromagnet inside the relays, and activates or deactivates the devices. The commands to the Arduino Web Server are issued via POST or GET requests.

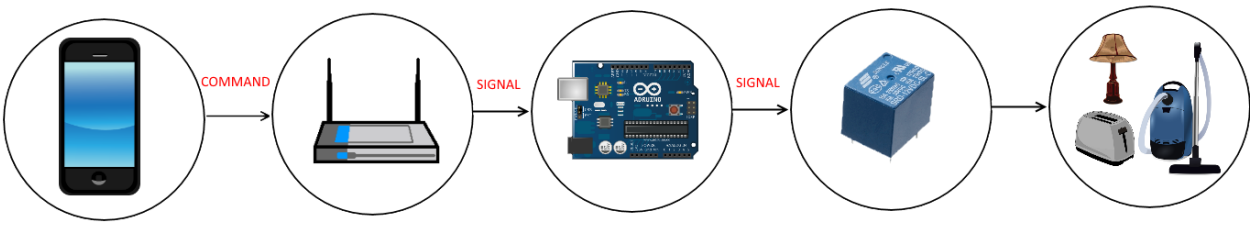

Figure 3-2. Arduino Actuator and Relay Control

Each command sent by the application or a web browser, corresponds to an Arduino action. For example the command “laon” corresponds to the activation of the first connected device, while the command “laoff” corresponds to the deactivation of the first connected device. Some commands were also built to activate or deactivate multiple devices at once.

```
[...]
if (readString.indexOf("?laon") > 0) //on device 1
            {
              digitalWrite(2, HIGH);
            }
else if (readString.indexOf("?laoff") > 0) //off device 1
            {
              digitalWrite(2, LOW);
            }
else if (readString.indexOf("?lbon") > 0) //on device 2
            {
              digitalWrite(3, HIGH);
            }
else if (readString.indexOf("?lboff") > 0) //off device 2
```

```
            {
              digitalWrite(3, LOW);
            }
else if (readString.indexOf("?lcon") > 0) //on device 3
            {
              digitalWrite(4, HIGH);
            }
else if (readString.indexOf("?lcoff") > 0) //off device 3
            {
              digitalWrite(4, LOW);
            }
else if (readString.indexOf("?ldon") > 0) //on device 4
            {
              digitalWrite(5, HIGH);
            }
else if (readString.indexOf("?ldoff") > 0) //off device 4
            {
              digitalWrite(5, LOW);
            }
else if (readString.indexOf("?leon") > 0) //on all devices
            {
              digitalWrite(2, HIGH);
              digitalWrite(3, HIGH);
              digitalWrite(4, HIGH);
              digitalWrite(5, HIGH);
              delay(10);
            }
  else if (readString.indexOf("?leoff") > 0) //off all devices
            {
              digitalWrite(2, LOW);
              digitalWrite(3, LOW);
              digitalWrite(4, LOW);
              digitalWrite(5, LOW);
              delay(10);
            }
[...]
```

3.1.4 Arduino Automations and Optimal Control Code

In order to reduce power consumption it was deemed necessary to automate the deactivation of devices under specific conditions. The automatic modes built on the platform, can be selected via the Android app. These modes are named autofmode, autowmode and autowhmode. The first one, once selected, will deactivate all light bulbs and LED strips, once the reading of the light sensor reaches a certain level. As a result, once the light inside a room is adequate, all the electric lights will automatically turn off. The second mode, will deactivate all connected high current devices, once the total consumed watts reach a certain value (e.g. 10Watts). The third automatic mode, continuously monitors power consumption, and, if selected, will disable all connected devices once the power consumed up to that point reaches a certain level, for example 10kWh.

```
[...]
bool autofmode = false;
bool autowmode = false;
bool autowhmode = false;
[...]


else if (readString.indexOf("?afm") > 0)
    {
    //Serial.println("inside ?afm");
    autofmode = true;
    }
else if (readString.indexOf("?awm") > 0)
    {
    //Serial.println("inside ?awm");
    autowmode = true;
    }
else if (readString.indexOf("?awhm") > 0)
    {
    //Serial.println("inside ?awhm");
    autowhmode = true;
    }
[...]
```

```
if (autofmode == true) {
    if (illuminance >= 35) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    autofmode = false;
    }
}
if (autowmode == true) {
    //Serial.println("inside true");
    if (mwatt >= 1500) {
    //Serial.println("inside oria");
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    autowmode = false;
    }
}
if (autowhmode == true) {
    //Serial.println("inside autowhmode");
    long metr = calculatemwh(mwatt);
    client.println(metr);
    //Serial.println(metr);
    if (metr >= 30000) {
        digitalWrite(2, LOW);
        digitalWrite(3, LOW);
        digitalWrite(4, LOW);
        digitalWrite(5, LOW);
        sumwh=0;
        autowhmode = false;
    }
}
```

The algorithm for the above logic is presented in the flowchart below:

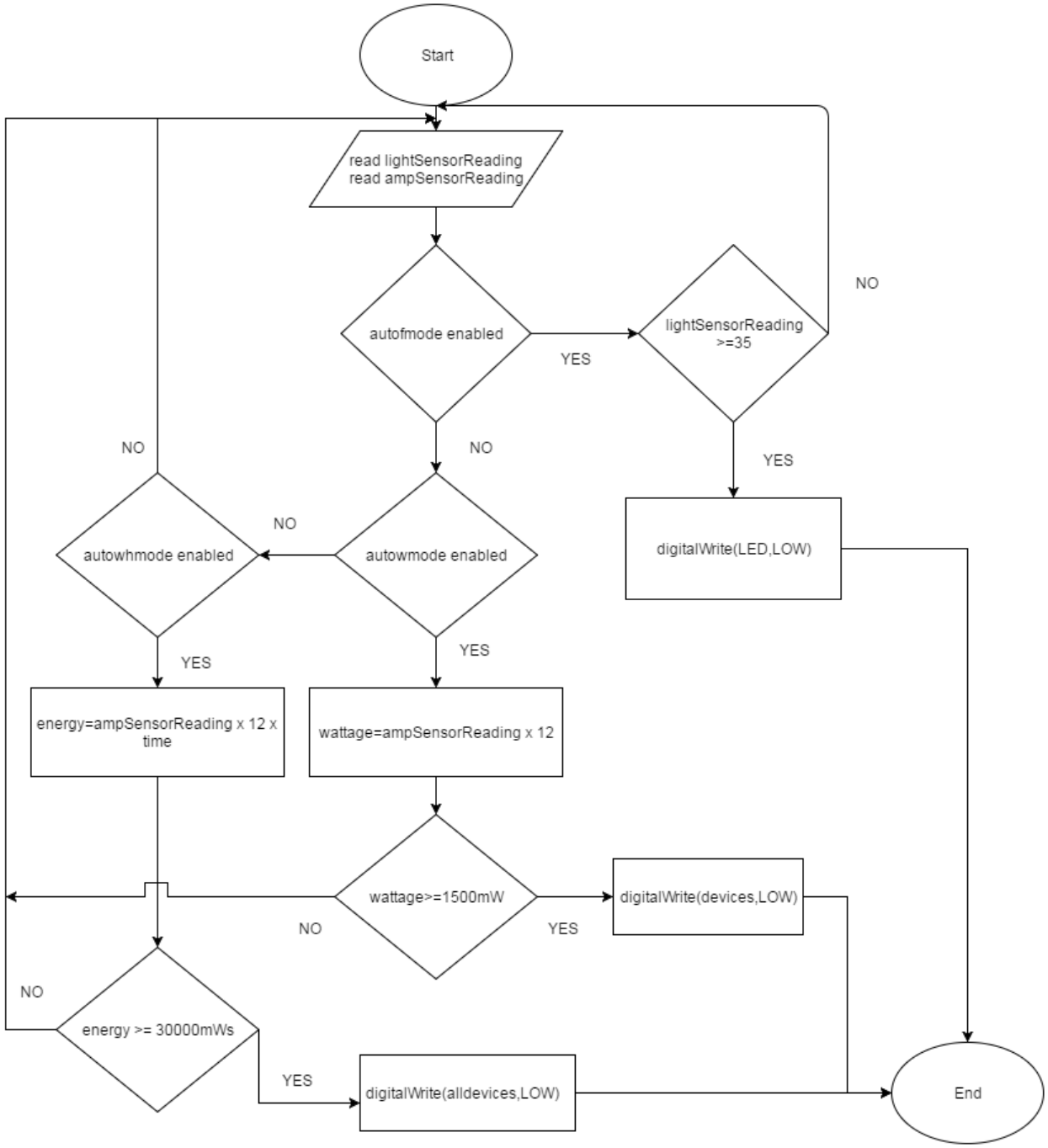

Figure 3-3. Logic control automations flowchart

3.1.5 Arduino Array Readings for Charts

The development of a charting component in the Android app, requires the Arduino code to display an Array of readings. For example the last five power consumption readings are stored in the Array, and then displayed as xml elements on the web server. At first the Array is declared and contains no readings. A variable is also declared, that contains a pointer for the first element of the Array.

```
int arrayMw[]={0,0,0,0,0};
int pointer =0;
```

The array is then populated with readings from the sensor, and each loop the value of the pointer is increased by one.

```
arrayMw[pointer]=mwatt;
pointer++;
```

Once the Array contains readings in all elements, its contents are transferred to the [i-1] position.

```
if (pointer==5){
    //metafora [i] se [i-1]
    arrayMw[0]=arrayMw[1];
    arrayMw[1]=arrayMw[2];
    arrayMw[2]=arrayMw[3];
    arrayMw[3]=arrayMw[4];
    //gia na einai pio katanoito ston meso xristi, ennoeitai ginetai
kai me for loop
    pointer=4;
}
```

Last but not least, all the elements of the Array are displayed on the xml / json file:

```
client.println("<m0>");
client.println(arrayMw[0]);
client.println("</m0>");
client.println("<m1>");
client.println(arrayMw[1]);
```

```
client.println("</m1>");
client.println("<m2>");
client.println(arrayMw[2]);
client.println("</m2>");
client.println("<m3>");
client.println(arrayMw[3]);
client.println("</m3>");
client.println("<m4>");
client.println(arrayMw[4]);
client.println("</m4>");
```

3.2 App Development

3.2.1 UI/UX Development

The building process of the application, required the UI/UX Elements to be built, using a Framework. The main menu of the application was built with the use of the JQuery Mobile Framework, which offers a lot of theming options on elements such as lists and buttons. The following code creates the main Listview of elements, with links to the app submenus (Temperature, Illuminance, Electric Current, Power, Plot and About).

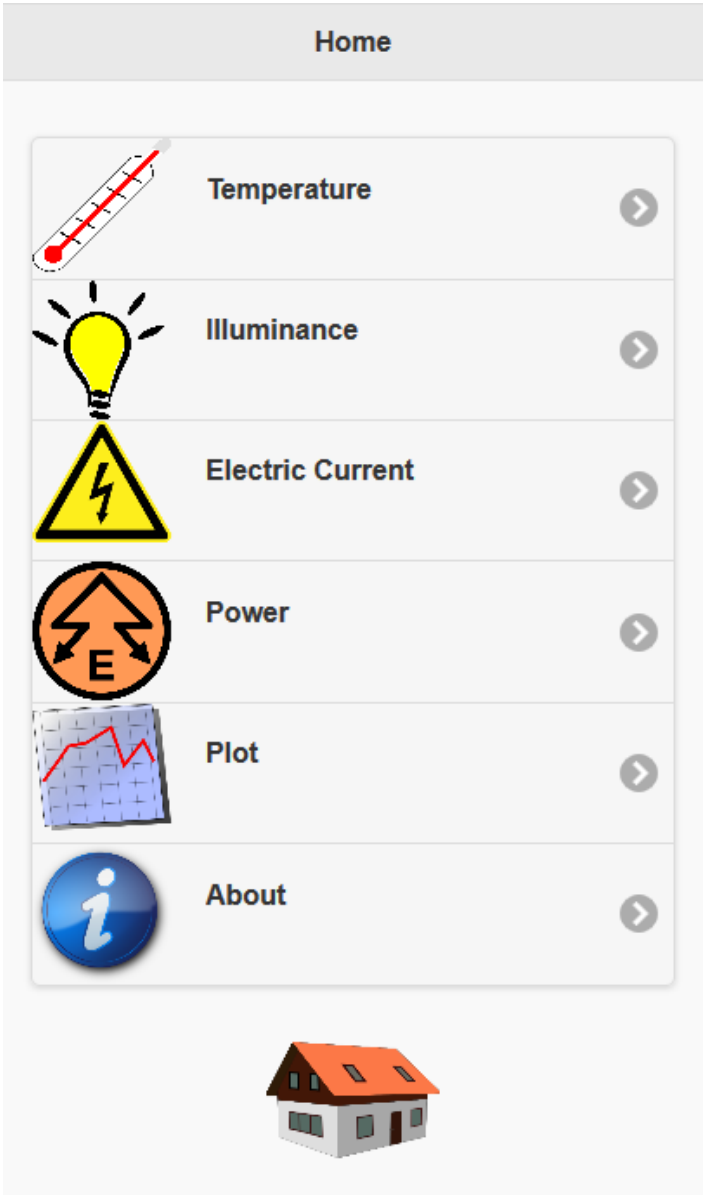

Figure 3-4. Jquery UI, main app menu

```
<!DOCTYPE html>
<html>
<head>

<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>

</head>
<body>
<div data-role="page">

    <div data-role="header" data-theme="a">
        <h1>Home</h1>
    </div>

    <div role="main" class="ui-content">

 <ul data-role="listview" data-inset="true">
    <li>
    <a href="temp.html" data-ajax="false">
        <img src="temp.png">
        <h2>Temperature</h2>
    </a>
    </li>
    <li><a href="fwteinotita.html" data-ajax="false">
        <img src="fws.png">
        <h2>Illuminance</h2>
    </a>
    </li>
    <li><a href="current.html" data-ajax="false">
        <img src="current.png">
        <h2>Electric Current</h2>
    </a>
    </li>
```

```
    <li><a href="power.html" data-ajax="false">
        <img src="power.png">
        <h2>Power</h2>
    </a>
    </li>
    <li><a href="plot.html" data-ajax="false">
        <img src="plot.png">
        <h2>Plot</h2>
    </a>
    </li>
    <li><a href="about.html" data-ajax="false">
        <img src="about.png">
        <h2>About</h2>
    </a>
    </li>
</ul>
    </div>
    <center>
      <img src="smarthouse.png" width="100">
    </center>
</div>
</body>
</html>
```

In order to create the button elements, the JQuery Mobile Framework was also used. Splitting the screen content into groups allows to group the buttons together.

Figure 3-5. JQuery Mobile Button Elements

```
<div class="ui-grid-c" >
      <div class="ui-block-a" ><button class="ui-btn" id="b0">
      Light 1</button></div>
      <div class="ui-block-b"><button class="ui-btn" id="b2">
      Light 2</button></div>
      <div class="ui-block-c"><button class="ui-btn" id="b4">
      Light 3</button></div>
      <div class="ui-block-d"><button class="ui-btn" id="b6">
      Light 4</button></div>
</div><!-- /grid-b -->


<div class="ui-grid-c">
      <div class="ui-block-a" ><button class="ui-btn ui-btn-b" id="b1">
      Light 1</button></div>
      <div class="ui-block-b"><button class="ui-btn ui-btn-b" id="b3">
      Light 2</button></div>
      <div class="ui-block-c"><button class="ui-btn ui-btn-b" id="b5">
      Light 3</button></div>
      <div class="ui-block-d"><button class="ui-btn ui-btn-b" id="b7">
      Light 4</button></div>
</div><!-- /grid-b -->

<button class="ui-btn" id="b8">Turn On All the Lights</button>

<button class="ui-btn ui-btn-b" id="b9">
                                    Turn Off All the Lights</button>
```

The code for the button UI which were developed for the proposed application, is shown above. Code for controlling the usage of its button was also developed. Each button has its own unique identifier (for example id=”b1”) which allows to be programmatically be controlled using a js script. The script for controlling each button is presented below.

```
<script>

$(document).ready(function(){
    $("#b0").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
    });
    $("#b1").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
    });
    $("#b2").click(function(){
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
    });
    $("#b3").click(function(){
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
    });
    $("#b4").click(function(){
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
    });
    $("#b5").click(function(){
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
    });
    $("#b6").click(function(){
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b7").click(function(){
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
```

```
    $("#b8").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b9").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b10").click(function(){
        $.get("http://192.168.1.120/?afm", function(data, status){
        });
    });
});

</script>
```

3.2.2 Chart Software Development

The graphical representation of power consumption data in the Android app, was deemed necessary. Such a representation will let the user compare his current power consumption with past measurements. The ultimate goal of plotting such data, is to minimize energy consumption.

The usage of a charting library –ChartJS–, allows a great many types of charts and options.

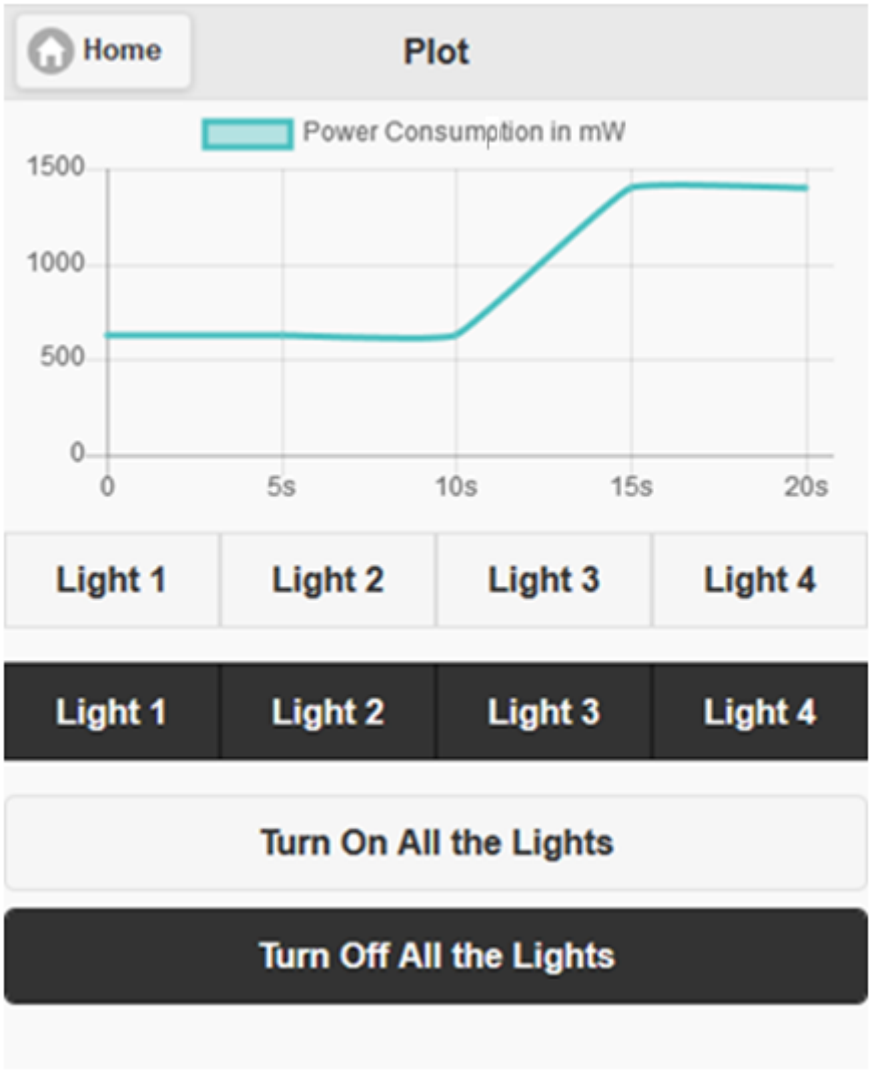

Figure 3-6. ChartJS, plot instance

Plotting Arduino data, requires to fetch them properly from the XML / JSON on the WebServer. The following code parses the XML values from the HTML file on the WebServer using the getElementsByTagName() method, and then stores its values on five variables.

```
<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var q=xmlDoc.getElementsByTagName("m0")[0].childNodes[0].nodeValue;
var w=xmlDoc.getElementsByTagName("m1")[0].childNodes[0].nodeValue;
var e=xmlDoc.getElementsByTagName("m2")[0].childNodes[0].nodeValue;
var r=xmlDoc.getElementsByTagName("m3")[0].childNodes[0].nodeValue;
var t=xmlDoc.getElementsByTagName("m4")[0].childNodes[0].nodeValue;
</script>
```

The plotting API of the ChartJS library offers many customization options. The script that is executed in order to plot the data from the XML or the JSON file is given below.

```
<script>
var ctx = document.getElementById("myChart");
var myChart = new Chart(ctx, {
    type: 'line',
    data: {
        labels: ["0", "5s", "10s", "15s", "20s"],
        datasets: [{
                    label: "Power Consumption in mW",
            fill: false,
            lineTension: 0.1,
            backgroundColor: "rgba(75,192,192,0.4)",
            borderColor: "rgba(75,192,192,1)",
            borderCapStyle: 'butt',
            borderDash: [],
```

```
            borderDashOffset: 0.0,
            borderJoinStyle: 'miter',
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data: [q, w, e, r, t],
            spanGaps: false,
        }]
    },
    options: {
        scales: {
            yAxes: [{
                ticks: {
                    beginAtZero:true
                }
            }]
        }
    }
});
</script>
```

Last but not least, the positioning of the chart can be selected by placing the div element on the body of the HTML document.

```
<div style="width:100%;">
    <canvas id="myChart"></canvas>
</div>
```

3.2.3 Gauges Software Development in UI

The readings of the sensors connected to the Arduino Development Board, can be displayed on UI elements such as gauges on the Android App. Canvas Gauges were selected, because they were specifically built for Internet of Things applications, and are highly configurable. In order to display the reading from the temperature sensor, a linear gauge object was built. The parameters minValue and maxValue were set to display the usable range of the sensor (0-60 degrees). Setting the highlights parameter, as a rgba colour value, allows the scale to show different colors for each temperature range. The scale colors blue, green and red represent cold temperatures, normal temperatures and hot temperatures. Last but not least, setting the boolean parameter valueBox to true, allows the gauge to display the exact sensor reading inside a box at the bottom of the scale.

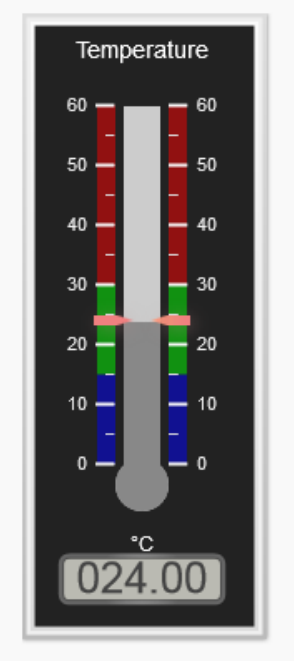

Figure 3-7. Temperature Gauge

```
<script>
new LinearGauge({
    renderTo: 'gauge1',
    width: 150,
    height: 380,
    units: '°C',
    title: "Temperature",
```

```
    value: 0,
    minValue: 0,
    maxValue: 60,
    majorTicks: ['0','10','20','30','40','50','60'],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [ { from: 0, to: 15, color: 'rgba(0,0,255,0.5)' },
                  { from: 15, to: 30, color: 'rgba(0,255,0,0.5)' },
                  { from: 30, to: 60, color: 'rgba(255,0,0,0.5)' }  ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();
if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=z;

</script>
```

The illuminance reading from the light sensor is displayed on a Radial Gauge object. It is configured to display values from 0 to 100 lux. Parameters, such as different scale colors for dark or light were set. The complete code for creating the RadialGauge object that displays the light sensor readings, is presented below.

Figure 3-8. Illuminance Gauge

```
<script>
new RadialGauge({
    renderTo: 'gauge1',
    width: 300,
    height: 300,
    units: 'LUX',
    title: "Illuminance",
    minValue: 0,
    maxValue: 100,
    majorTicks: [
        '0','10','20','30','40','50','60','70','80','90','100'
    ],
    minorTicks: 2,
    ticksAngle: 270,
    startAngle: 45,
```

```
    strokeTicks: true,
    highlights  : [
        { from : 0,  to : 30, color : 'rgba(150, 150, 150, 0.5)' },
        { from : 30, to : 100, color : 'rgba(255, 255, 0, 0.5)' }
    ],
    valueInt: 1,
    valueDec: 0,
    colorPlate: "#fff",
    colorMajorTicks: "#686868",
    colorMinorTicks: "#686868",
    colorTitle: "#000",
    colorUnits: "#000",
    colorNumbers: "#686868",
    valueBox: true,
    colorValueText: "#000",
    colorValueBoxRect: "#fff",
    colorValueBoxRectEnd: "#fff",
    colorValueBoxBackground: "#fff",
    colorValueBoxShadow: false,
    colorValueTextShadow: false,
    colorNeedleShadowUp: true,
    colorNeedleShadowDown: false,
    colorNeedle: "rgba(200, 50, 50, .75)",
    colorNeedleEnd: "rgba(200, 50, 50, .75)",
    colorNeedleCircleOuter: "rgba(200, 200, 200, 1)",
    colorNeedleCircleOuterEnd: "rgba(200, 200, 200, 1)",
    borderShadowWidth: 0,
    borders: true,
    borderInnerWidth: 0,
    borderMiddleWidth: 0,
    borderOuterWidth: 5,
    colorBorderOuter: "#fafafa",
    colorBorderOuterEnd: "#cdcdcd",
    needleType: "arrow",
    needleWidth: 2,
    needleCircleSize: 7,
    needleCircleOuter: true,
    needleCircleInner: false,
```

```
    animationDuration: 1500,
    animationRule: "dequint",
    fontNumbers: "Verdana",
    fontTitle: "Verdana",
    fontUnits: "Verdana",
    fontValue: "Led",
    fontValueStyle: 'italic',
    fontNumbersSize: 20,
    fontNumbersStyle: 'italic',
    fontNumbersWeight: 'bold',
    fontTitleSize: 24,
    fontUnitsSize: 22,
    fontValueSize: 50,
    animatedValue: true
}).draw();

if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=q;

</script>
```

The current Amperage reading is displayed on a RadialGauge Object. The parameters maxValue and majorTicks can be changed to display higher current dc or ac loads. The units of the scale (mA) can also be changed, if deemed necessary, to Amps. The valueBox parameter was also set to true, in order to display the exact amperage value read from the current sensor connected to the Arduino.

Figure 3-9. Electric Current Gauge

```
<script>
new RadialGauge({
    renderTo: 'gauge1',
    width: 300,
    height: 300,
    units: 'mA',
    title: "Current",
    value: 0,
    minValue: 0,
    maxValue: 400,
    majorTicks: [
        '0','50','100','150','200','250','300','350','400'
    ],
    minorTicks: 2,
    strokeTicks: false,
```

```
    highlights: [
        { from: 0, to: 150, color: 'rgba(0,255,0,.15)' },
        { from: 150, to: 300, color: 'rgba(255,255,0,.15)' },
        { from: 300, to: 400, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();


if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=x;

</script>
```

The current power consumption and the total power consumption are displayed using two RadialGauge objects, gauge2 and gauge3. The parameter units of the first one is set to mW, although it can easily be configured to display the value in Watts. The total power consumption units parameter is set to mWs (milliwatts x seconds) for demo purposes. The application code, also includes modes for setting this parameter to watt-hours (watts x hours) and killowatt-hours (kwatt x hours).

Figure 3-10. Power Gauge

Figure 3-11. Energy Consumption Gauge

```
<script>

new RadialGauge({
    renderTo: 'gauge2',
    width: 235,
    height: 235,
    units: 'mW',
    title: "Power",
    value: 0,
    minValue: 0,
    maxValue: 4000,
    majorTicks: [
        '0','500','1000','1500','2000','2500','3000','3500','4000'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 1500, color: 'rgba(0,255,0,.15)' },
        { from: 1500, to: 3000, color: 'rgba(255,255,0,.15)' },
        { from: 3000, to: 4000, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();

new RadialGauge({
    renderTo: 'gauge3',
    width: 235,
    height: 235,
```

```
    units: 'mWs',
    title: "Energy Consumption",
    value: 0,
    minValue: 0,
    maxValue: 40000,
    majorTicks: [
        '0','10000','20000','30000','40000'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 20000, color: 'rgba(0,255,0,.15)' },
        { from: 20000, to: 30000, color: 'rgba(255,255,0,.15)' },
        { from: 30000, to: 40000, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();


if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}

if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
```

```
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}

var gauge = document.gauges.get('gauge2');
gauge.value=y;

var gauge = document.gauges.get('gauge3');
gauge.value=z;

</script>
```

3.2.4 Android app development

The conversion of the HTML5 elements to an mobile application can be achieved with the WebView class, an extension of Android's View class that allows to display local or remote HTML documents as a part of the activity layout. A WebView must first be added to the activity layout.

```
<?xml version="1.0" encoding="utf-8"?>
<WebView  xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/webview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
/>
```

Afterwords, any local html5 files, placed under the /assets/ folder can be accessed. Loading a local HTML file from the android app can be achieved with the following code snippet.

```
WebView myWebView = (WebView) findViewById(R.id.webview);
WebSettings webSettings = myWebView.getSettings();
webSettings.setJavaScriptEnabled(true);
myWebView.loadUrl("file:///android_asset/main.html");
```

The app should also be able to access a network, in order to receive sensor data, and send trigger commands to the Arduino base station. As a result, the Internet permission must be added to the manifest file of the Android app.

```
<manifest ... >
    <uses-permission android:name="android.permission.INTERNET" />
    ...
</manifest>
```

When the user clicks on the app that contains a link to a HTML5 page in the WebView, the default behavior is for Android to launch an application that handles URLs. Usually, the default web browser opens and loads the destination URL. However, it is possible to override this behavior for the WebView, so links open within it. It is then possible to allow the user to navigate backward and forward through the page history that's maintained by the WebView.

To open links clicked by the user, a WebViewClient must be provided for the WebView, using setWebViewClient():

```
WebView myWebView = (WebView) findViewById(R.id.webview);
myWebView.setWebViewClient(new WebViewClient());
```

When the WebView overrides URL loading, it automatically accumulates a history of visited web pages. It is possible to navigate backward and forward through the history with goBack() and goForward(). The back button is configured below to call the method goBack() if pressed.

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
// Check if the key event was the Back button and if there's history
    if ((keyCode == KeyEvent.KEYCODE_BACK) && myWebView.canGoBack()) {
        myWebView.goBack();
        return true;
    }

    return super.onKeyDown(keyCode, event);
}
```

3.2.5 Voice Recognition Development

The whole system can be controlled with voice commands. As we mentioned the MIT App inventor was used for this purpose. The application's start screen contains a "Voice Command" button. Pressing this button and issuing a correct command, sends a signal (getRequest()) to the system in order either to turn on or off a device. Many voice actions are incorporated in the app. The user is able to turn devices on or off using a wide variety of commands, such as "Turn on", "Turn off", "Yes", "No", "Anapse", "Svise", "Άναψε", "Σβήσε" and more.

Three instances of the application are presented below. The first one contains the start screen of the application. In the second picture the voice command "Turn on" is used in order to turn on the device, while in the third one the voice command "Turn off" is used to turn it off.

Figure 3-12. Application Start Screen

Figure 3-13. Activation Voice Command

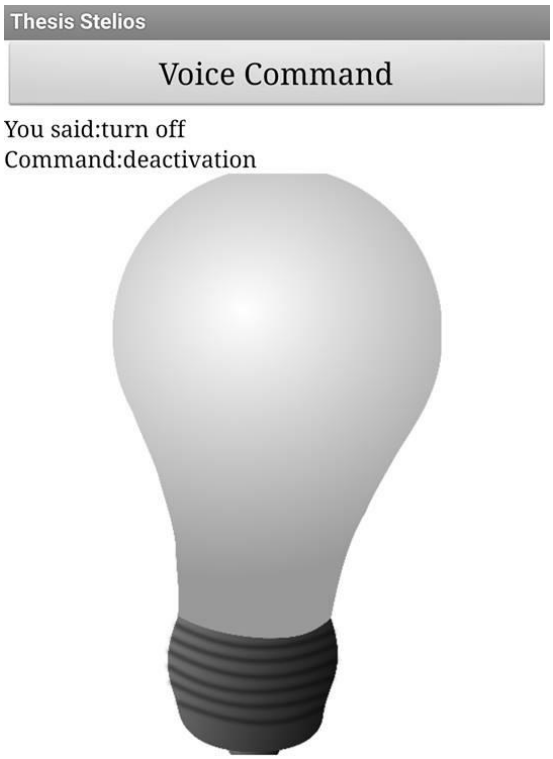

Figure 3-14. Deactivation Voice Command

In the next figure there are presented all the blocks and their code that were used in order to create the application.

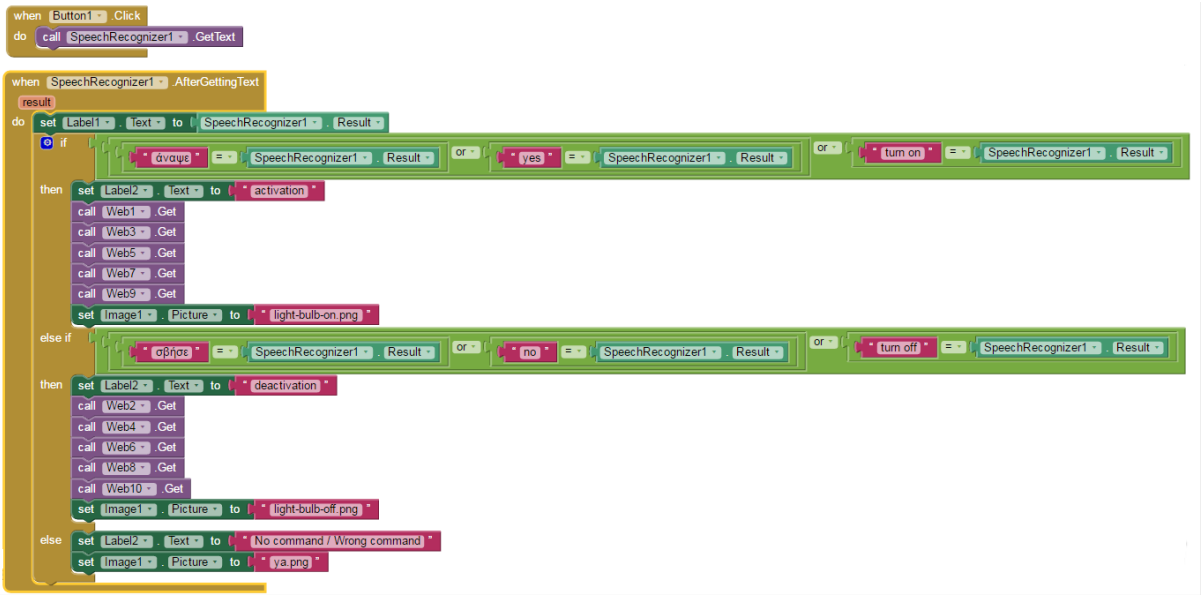

Figure 3-15. Application Blocks and Code

4. Fuzzy Logic Control

4.1 Optimal Control with the use of a Fuzzy Control System Implemented in the Matlab Fuzzy Logic Toolbox

In order to maximize the efficiency of the heating elements of a house, and minimize the power consumption an optimal control algorithm was implemented using fuzzy logic. Fuzzification is the process of associating crisp, or numerical, input values with the linguistic terms of the corresponding input linguistic variables.

Designing and testing the optimal control algorithms was achieved with the use of the Matlab software and its included fuzzy logic toolkit.

The first designed fuzzy controller associates the temperature readings from two thermometers with the linguistic terms “cold”, “normal”, and “hot" for the current temperature linguistic variables. Depending on the membership functions for the linguistic terms, the temperature values might correspond to one or more of the linguistic terms. The Fuzzy Process is presented in the block diagram below.

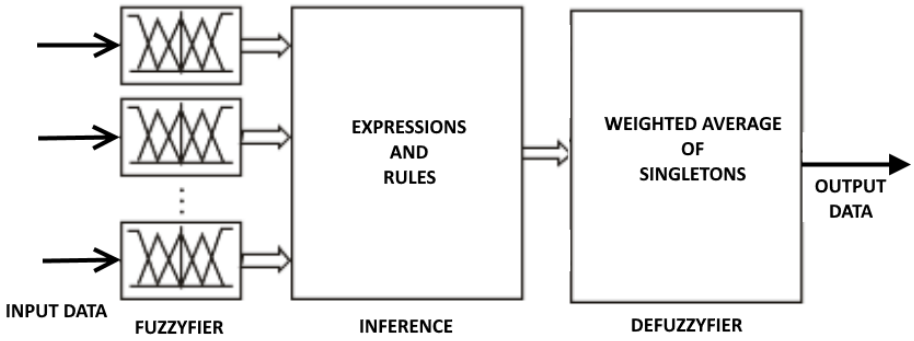

Figure 4-1. Fuzzy Process

The optimal control algorithm requires three triangular membership functions, the two thermometer inputs, and an output power MF.

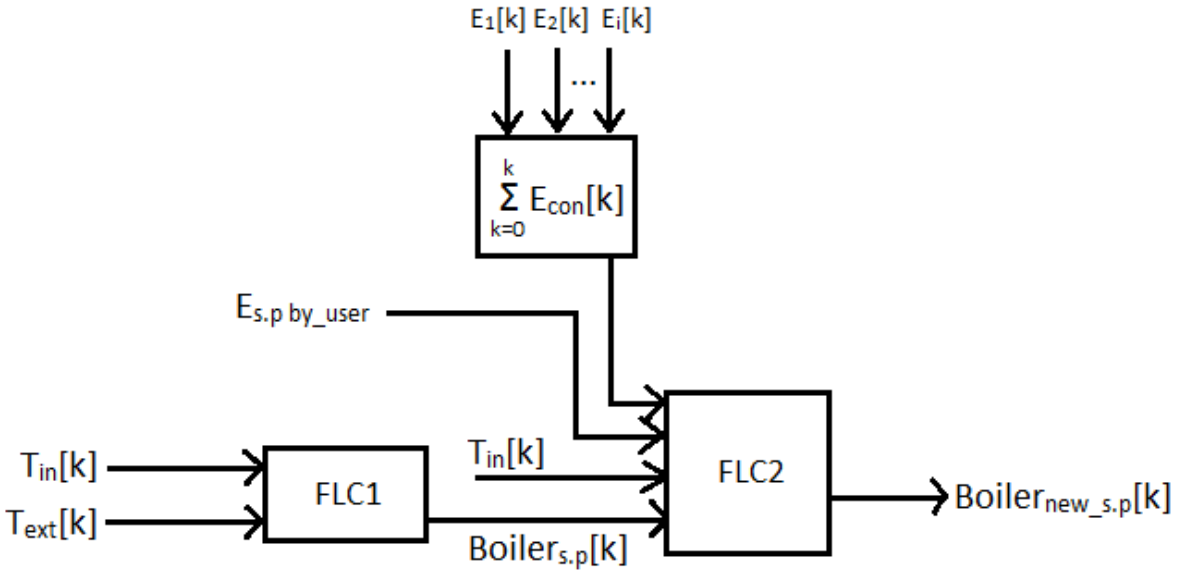

Figure 4-2. The proposed Fuzzy control system of home boiler where E_1, E_2, … , E_i is the Energy Consuption of the devices in kW

[k] notes the current measurement

FIS VARIABLES	Membership Functions (triangular)		
Internal Temp	-12 to 12, ISCOLD	0 to 24, ISNORMAL	12 to 24, ISHOT
External Temp	-40 to 0, ISCOLD	-20 to 20, ISNORMAL	O to 40, ISHOT
Output Power ($Boiler_{s.p}$ as a PWM signal)	-128 to 128, LOWP	0 to 255, MEDIUMP	128 to 383, HIGHP

Table 4-1. FIS Variables and their Membership Functions for the FLC1

The membership functions for the Internal and the External Temperature are presented below:

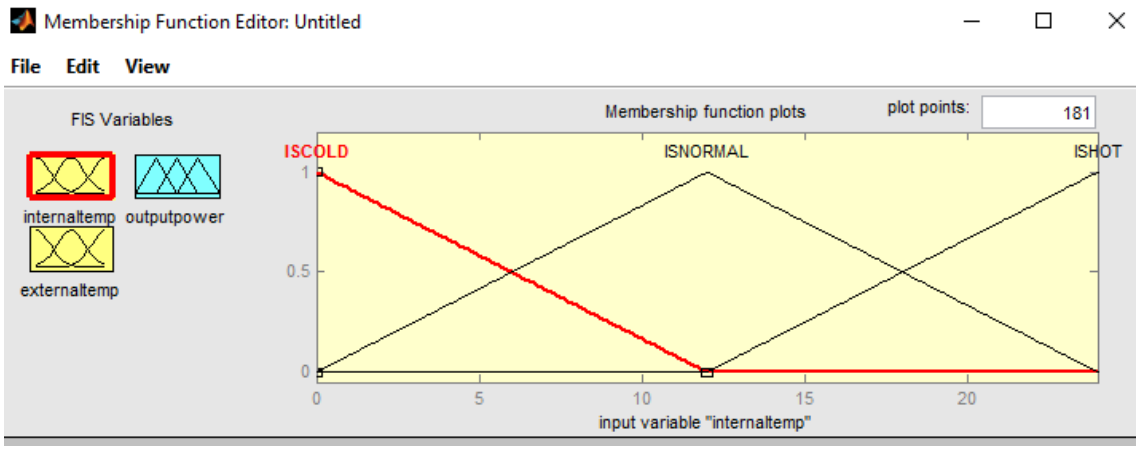

Figure 4-3. Internal Temperature Membership Function for the FLC1

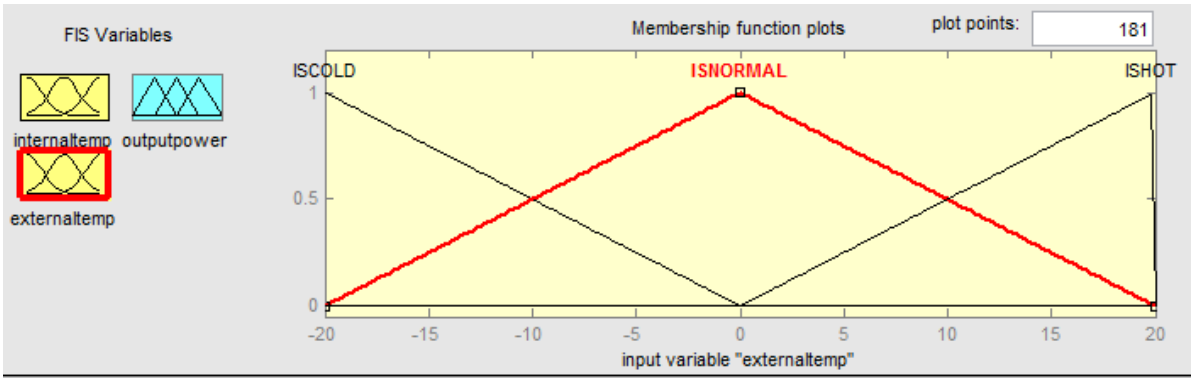

Figure 4-4. External Temperature Membership Function for the FLC1

The Membership Function of the Output Power is presented below:

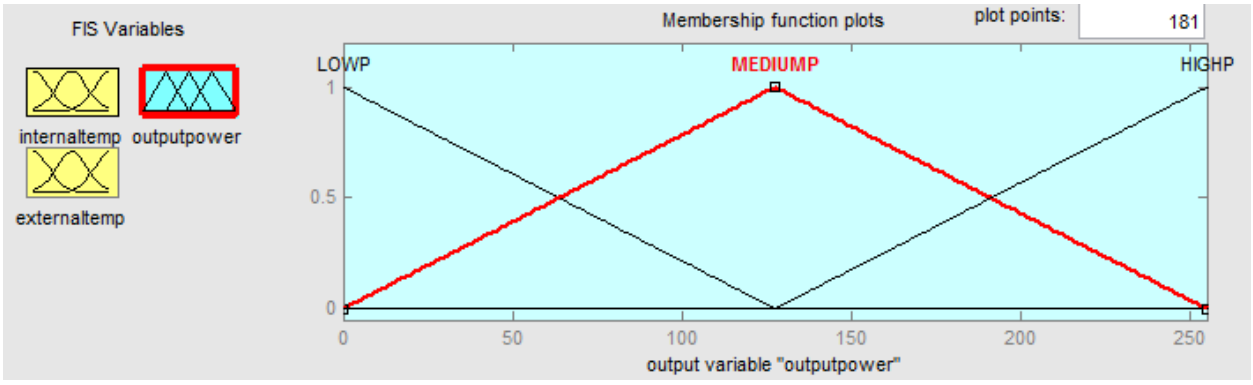

Figure 4-5. Output Power PWM Function (a PWM signal controls the duty cycle of the boiler) for the FLC1

The rules of the designed system are presented below:

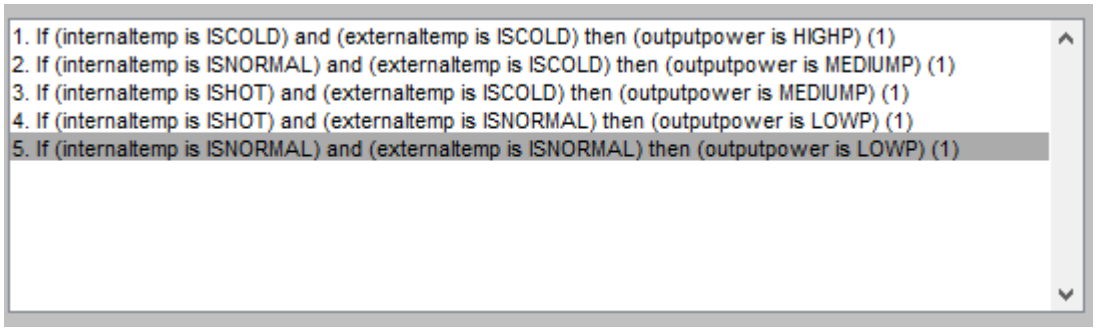

Figure 4-6. Fuzzy Logic Implementation Rules for the FLC1

A symbolic representation of the rules implemented:

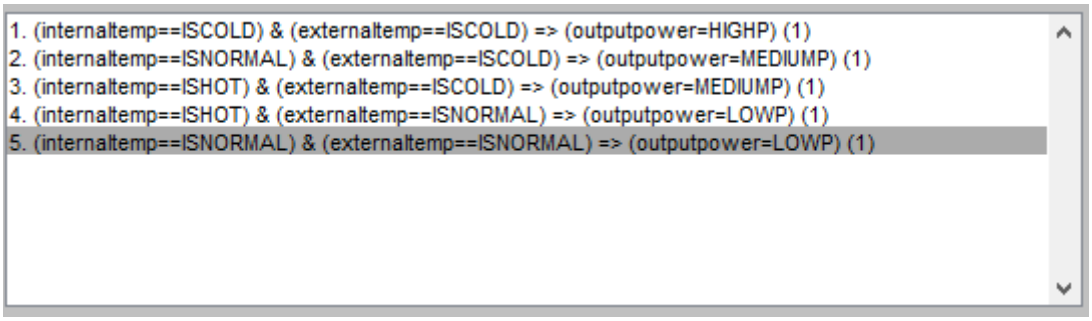

Figure 4-7. Fuzzy Logic Implementation Rules for the FLC1

For example the rule IF (internaltemp is ISCOLD) and (extrnaltemp is ISCOLD) then (outputpower is HIGHP), This rule uses the truth value of the "internaltemp" and “externaltemp” inputs, which is some truth value of "cold", to generate a result in the fuzzy set for the "outputpower" output, which is some value of "highp". This result is used with the results of other rules to finally generate the crisp composite output. Obviously, the greater the truth value of "cold", the higher the truth value of "high".

The output variable can then be used to control the duty cycle of a device –for example a heater, or a LED- via a signal. The Five rules are simulated below.

If the internal temperature is 6 degrees and the external temperature is -10 degrees, the pwm duty cycle of the output will be 127.

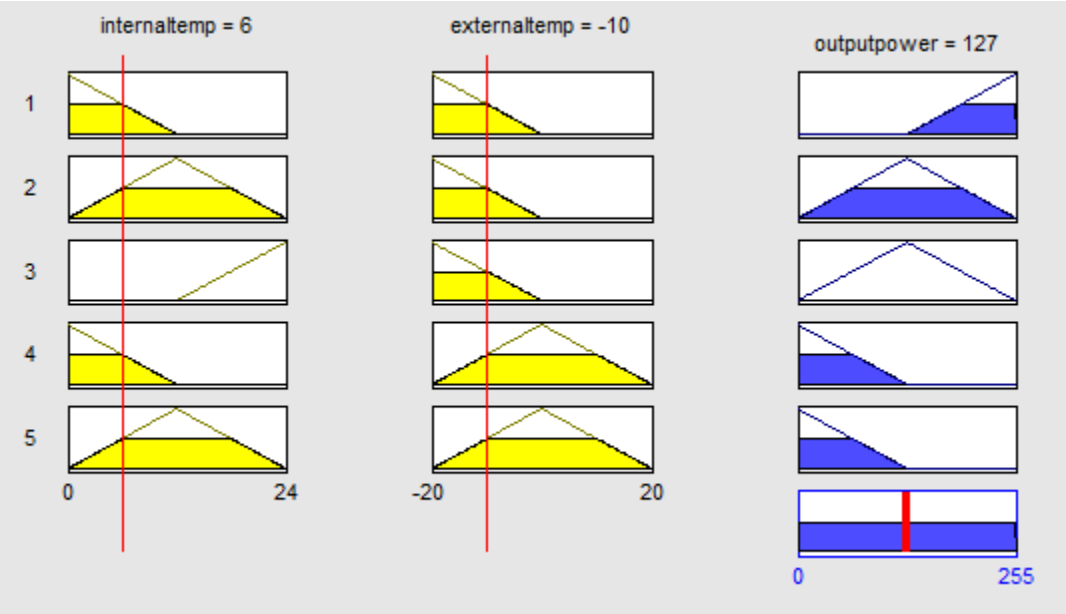

Figure 4-8. Fuzzy Logic Rule Simulation for the FLC1

If the internal temperature is 2 degrees and the external temperature is -17 degrees, the pwm duty cycle of the output will increase to 173, as intended:

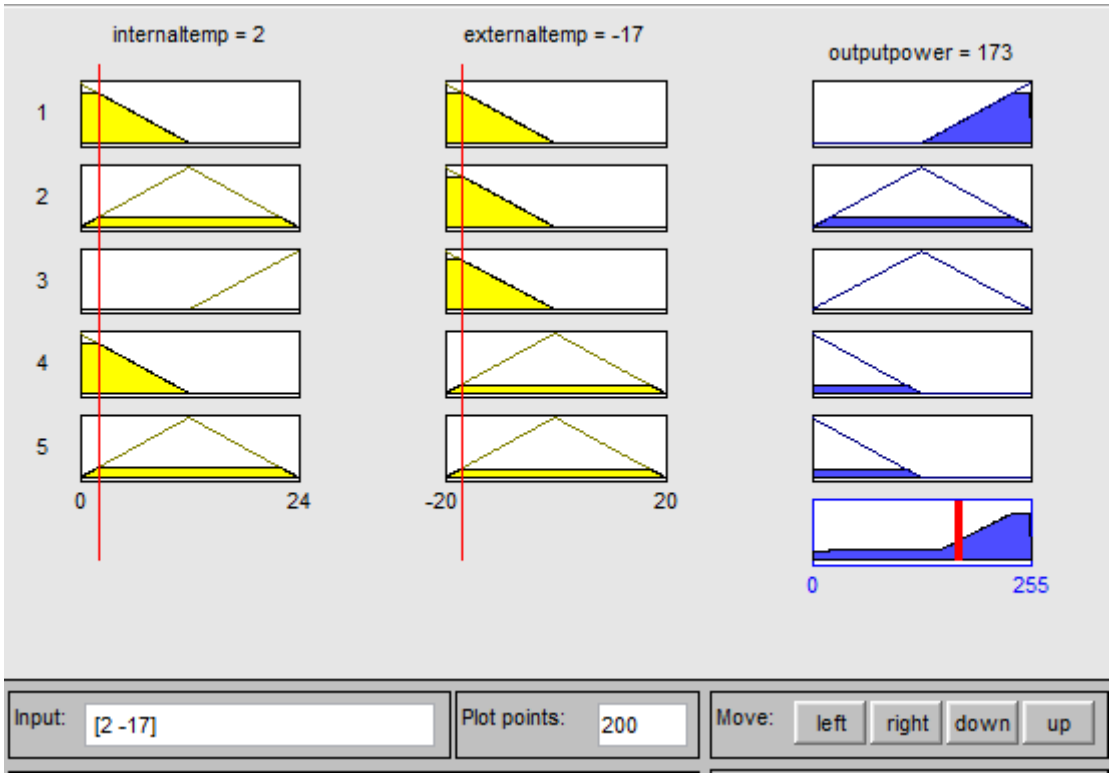

Figure 4-9. Fuzzy Logic Rule Simulation, Lower Temperatures, Higher Output for the FLC1

The second controller checks the current temperature and the total power consumption and controls the power level of a boiler (IsHigh, IsMedium, IsLow, IsOff).

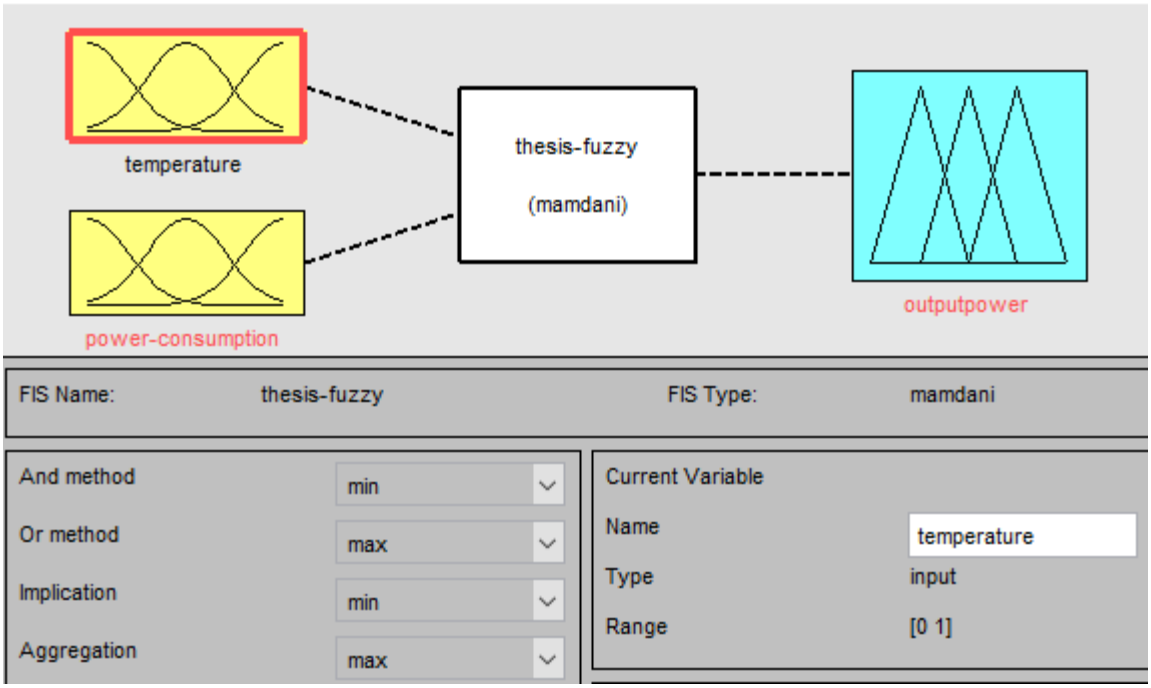

Figure 4-10. Temperature and Power Consumption control output power Levels Using Fuzzy Logic for the FLC1

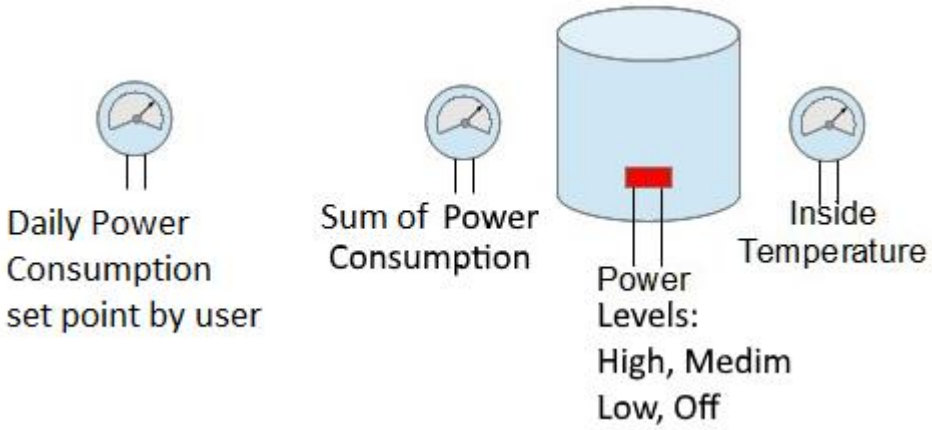

Figure 4-11. Power Consumption and Temperature data to control the power level of a boiler using fuzzy logic (inputs and output for the FLC2)

The optimal control algorithm requires three triangular membership functions: an internal temperature input, a power consumption input, and an output power output:

FIS VARIABLES	Membership Functions (triangular)		
Internal Temp	-8 to 8, ISCOLD	0 to 16, ISNORMAL	8 to 24, ISHOT
Power Consumption	-100 to 100, ISLOW	0 to 200, ISAVERAGE	100 to 300, ISHIGH
Output Power	-128 to 128, LOWP	0 to 255, MEDIUMP	128 to 384, HIGHP

Table 4-2. FIS Variables and their Membership Functions for the FLC2

The Membership Functions of the Power Consumption, the Temperature and the Output Power are presented below.

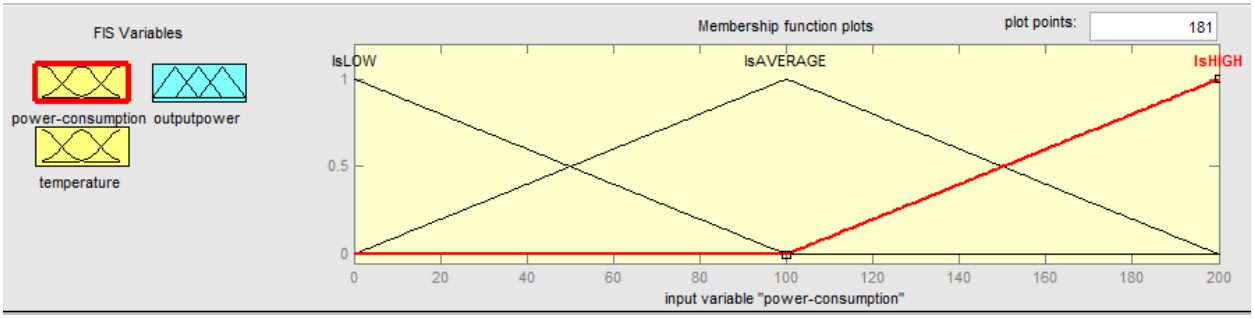

Figure 4-12. Power Consumption Membership Function for the FLC2

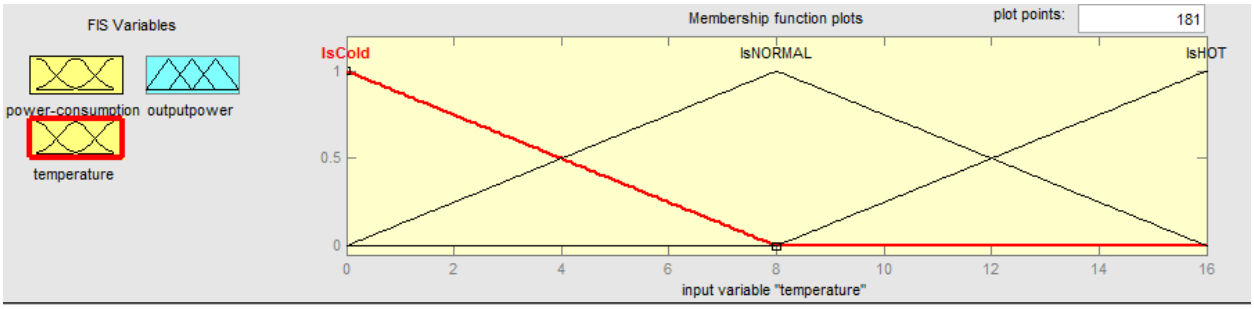

Figure 4-13. Temperature Membership Function for the FLC2

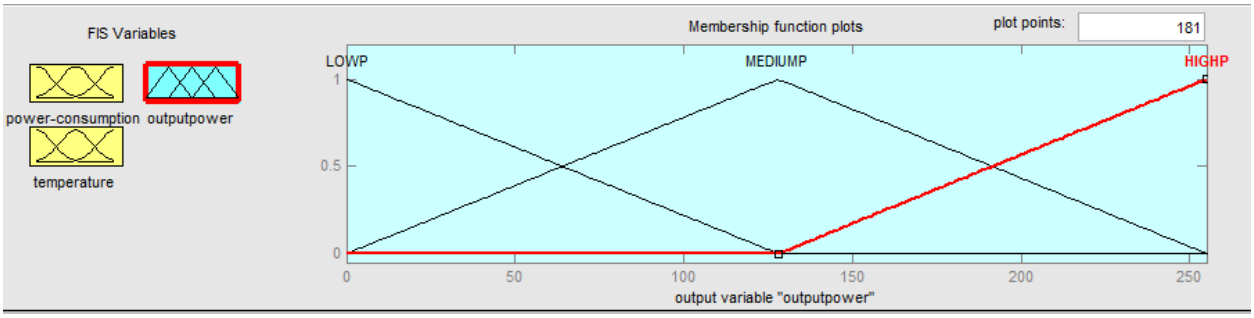

Figure 4.14. Output Power Membership Function for the FLC2

The rules of the system are definded below:

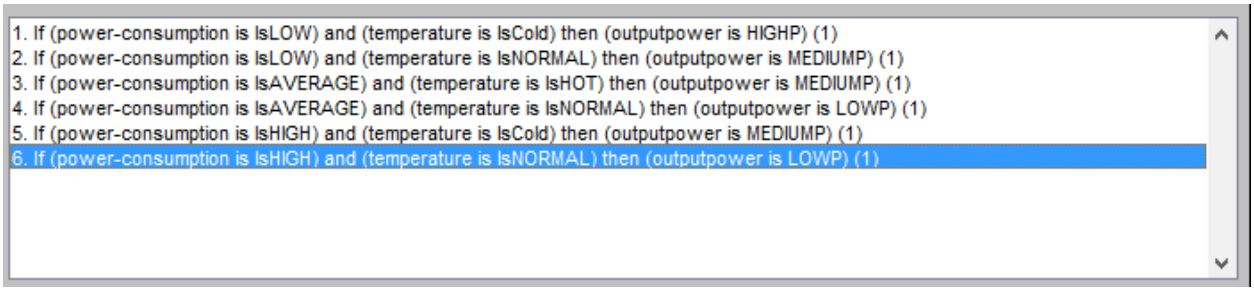

Figure 4-15. The Fuzzy Logic Rules

A simulation of the above control system is presented below:

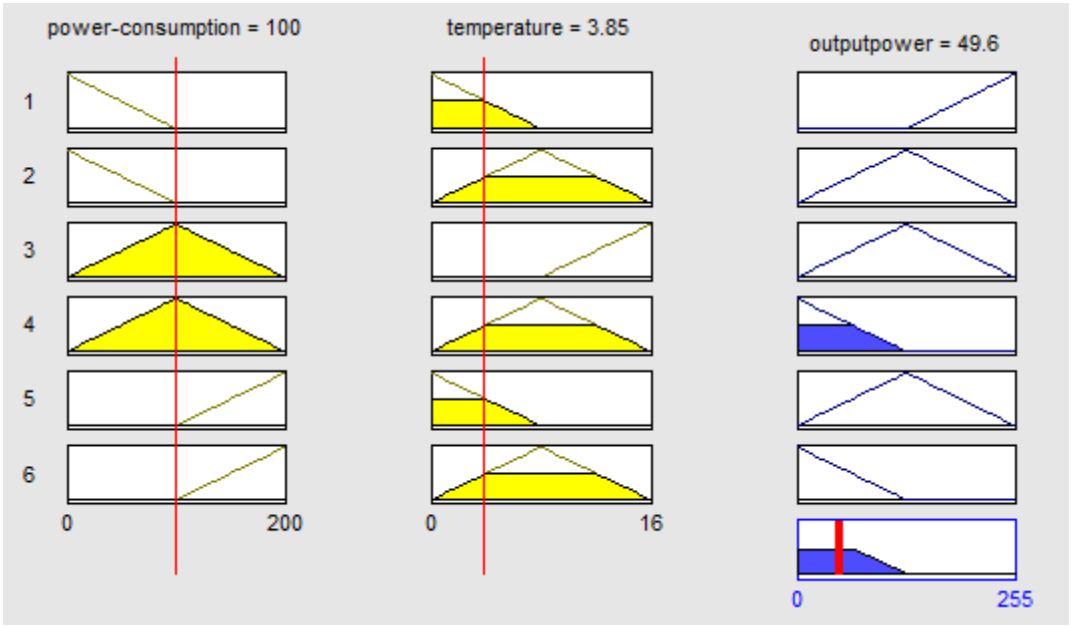

Figure 4-16. Fuzzy Logic System Simulation for the FLC2

4.2 Implementation of the Fuzzy Logic Algorithm on the Arduino

The implementation of the Fuzzy Logic Algorithm on the ATMega Microcontroller, requires the use of the Arduino IDE το compile and upload the sketch. The Fuzzy Logic Code also requires the Fuzzy.h library to be imported and included in the /libraries directory. The selected compiler included in the Arduino IDE is the AVR gcc (Arduino LLC, n.d.) (Atmel ATMega2560 Datasheet), and its default optimization level is –Os.

As already described, the first one compares temperature data from two different sensors in order to control the heat level of a boiler:

Required header files, -such as the Fuzzy.h file- are included in the c code below:

```
#include <Fuzzy.h>
[…]
```

The membership functions of three sets are declared in the Arduino Code. The selected library only allows triangular membership Functions. Three Membership Functions are Declared, InTemp, ExTemp, and Pwm Output. The value 0 describes the peak of the triangular Function -12 to 12, etc. The values of the MFs are then associated to linguistic terms.

```
MF3 InTemp(0,12,24); //Inside Temperatures, triangular MFS
MF3 ExTemp(-20,0,20); //External Temperatures, triangular MFS
Output(255, 128, 0); //Output that controls a device

#define IsCOLD IsA
#define IsNORMAL IsB
#define IsHOT IsC
#define HIGHP IsA
#define MEDIUMP IsB
#define LOWP IsC


 […]
```

The temperature parameters are passed to the class MF3 which is declared inside the library:

```
class Mf3
```

```
{
 private:
   int x1; int x2; int x3;          //Peaks : A,B,C
   int inp;
 public:
 Mf3(int A,int B,int C);
   Fv IsA();
   Fv IsB();
   Fv IsC();
   void setInp(int x);
};
```

The initialization of the inference system requires the number of rules followed by their name. The selected library uses by default the Mandami Inference Method.

```
Inference InfSystem(5,rule1,rule2,rule3,rule4,rule5);
```

The IF-THEN rules of the controller are then defined into separate void functions.

```
 […]
void rule1()
    {
    FzIf(InTemp.IsCOLD() && ExTemp.IsCOLD());
    FzThen(Pwm,HIGHP);
    }

void rule2()
    {
    FzIf(InTemp.IsNORMAL() && ExTemp.IsCOLD());
    FzThen(Pwm,MEDIUMP);
    }


void rule3()
    {
    FzIf(InTemp.IsHOT()&&ExTemp.IsCOLD());
    FzThen(Pwm,MEDIUMP);
    }
```

```
void rule4()
    {
    FzIf(InTemp.IsHOT()&&ExTemp.IsNORMAL());
    FzThen(Pwm,LOWP);
    }


void rule5()
    {
    FzIf(InTemp.IsNORMAL()&&ExTemp.IsNORMAL());
    FzThen(Pwm,LOWP);
    }
 […]
```

The output readings of the LM35 temperatures sensors are then passed as parameters to the methods:

[…]

```
InTemp.setInp(temp);
ExTemp.setInp(temp2);
InfSystem.compute();
int output = Pwm.getResult();
```

[…]

The InfSystem.compute() function calculates the crisp composite output of the system, which is then stored into the output variable using the .getResult() class.

The second implementation takes into consideration and combines temperature data with power consumption data to control the heat level of the boiler. This Fuzzy Logic Code also requires the Fuzzy.h library to be imported and included.

```
#include <Fuzzy.h>
[…]
```

The three triangular membership functions are then declared. The library accepts the peaks of the triangular sets. The InTemperature Membership Function and the PowerC MF are used as

inputs, while the Heater MF is used as an output. The linguistic terms IsCold, IsNormal, IsHot, IsLow, isAverage and isHigh are associated with the MFs.

```
MF3 InTemperature(0,8,16); //Internal Temperature Levels
MF3 PowerC(0,100,200); //Power Consumption, watt-hours levels
Out heater(255,128,0);

#define IsCOLD IsA
#define IsNORMAL IsB
#define IsHOT IsC
#define IsLOW IsA
#define IsAVERAGE IsB
#define IsHIGH IsC
[…]
```

The initialization of the inference system:

```
Inference
InfSystem2(6,rulenew1,rulenew2,rulenew3,rulenew4,rulenew5,rulenew6);
[…]
```

The defined rules of the controller are required to be declared into separate void functions.

For example, the function rulenew1(), takes into account the levels of IsCold and isLow, define the level of the IsHigh output.

```
void rulenew1()
    {
    FzIf(InTemp.IsCOLD && PowerC.isLOW);
    FzThen(heater.IsHigh);
    }

void rulenew2()
    {
    FzIf(InTemp.IsNORMAL && PowerC.isLOW);
    FzThen(heater,IsMEDIUM);
```

```
    }

void rulenew3()
    {
    FzIf(InTemp.IsHOT && PowerC.isAVERAGE);
    FzThen(heater,IsMEDIUM);
    }

void rulenew4()
    {
    FzIf(InTemp.IsNORMAL && PowerC.isAVERAGE);
    FzThen(heater,IsLOW);
    }

void rulenew5()
    {
    FzIf(InTemp.IsCOLD && PowerC.isHIGH);
    FzThen(heater,IsMEDIUM);
    }

void rulenew6()
    {
    FzIf(InTemp.IsNORMAL && PowerC.isHIGH);
    FzThen(heater,IsLOW);
    }

InTemperature.setInp(temp);
InPowerC.setInp(sumwh);
InfSystem2.compute();
int outputsig = heater.getResult();
```

The InfSystem.compute() function calculates the crisp composite output of the system, which is then stored into the output variable using the .getResult() class.

5. Industrial Design and Installation of the Platform

The main components of the platform can be installed inside the distribution board (fuse box) of a house. Each currents sensor will be installed in a separate power line inside the distribution board in order to take current measurements for each part of the house separately. The relay modules can also be installed inside the distribution board. Both sensors and relays, if deemed necessary can alternatively be installed inside wall sockets. Such a device is presented below:

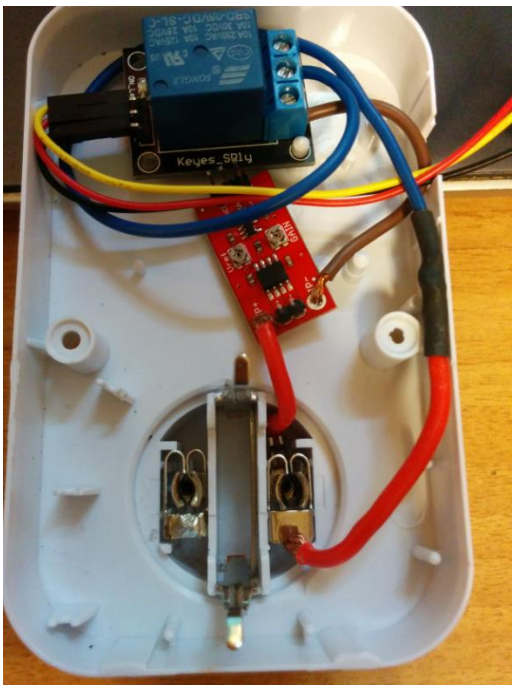

Figure 5-1. Relay Module (blue) and Current Sensor (red) mounted inside a socket

A smaller project box can also be used to minimize the space required inside the distribution board. The confined space of a smaller project box –instead of the 225x175x80mm that was used- has the negative side-effect of increasing the operating temperature of the electronic components such as the microcontroller board. As a result adequate active cooling is required.

An 8cm cooling fan was experimentally installed in a smaller project box and found to provide adequate cooling to the platform, keeping operating temperatures of the microcontroller below 40 degrees.

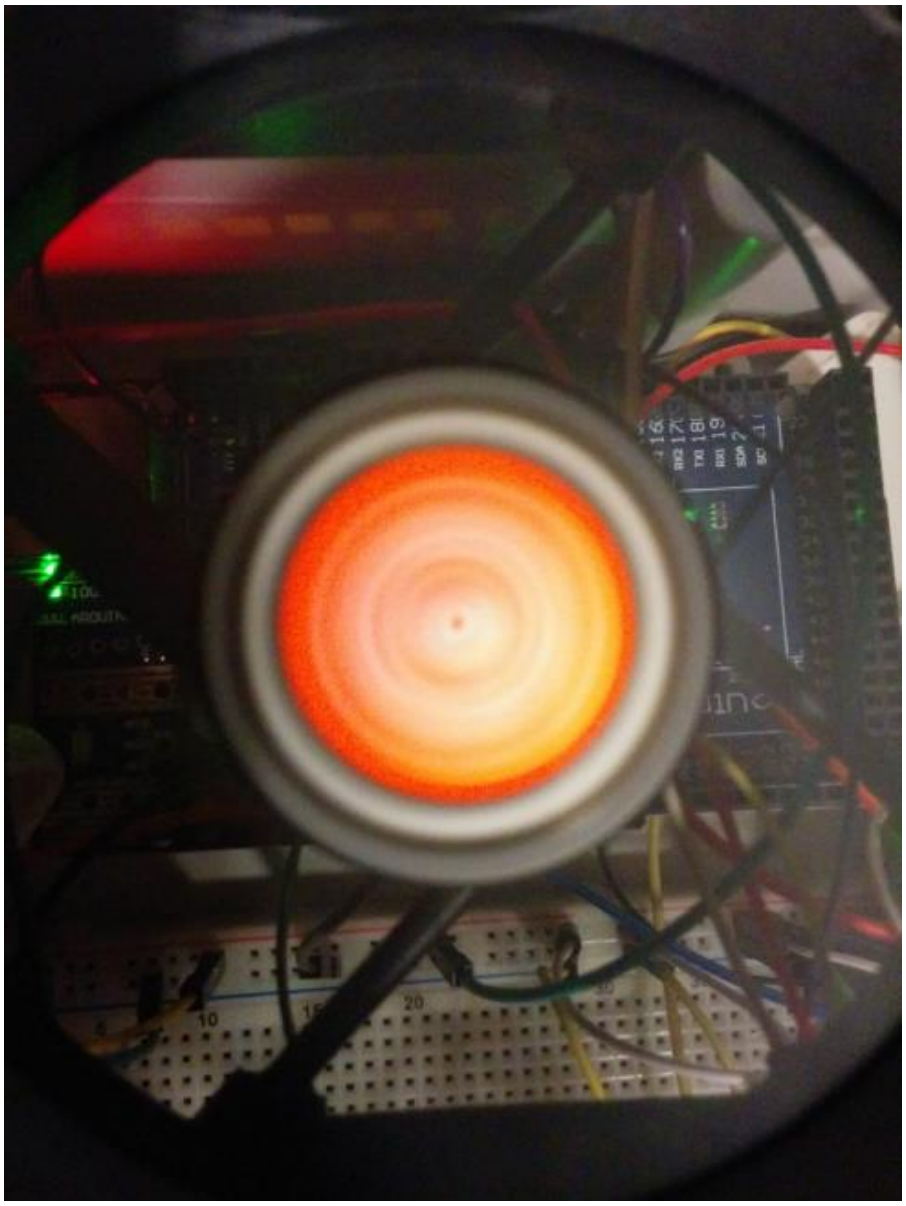

Figure 5-2. Experimental cooling fan installation on the platform

The system can easily be extended with a plethora of sensors, and/or more actuators. Additional sensors can be powered via the +3,3V, +5V or +12V lines that are already available. A voltage divider circuit or a separate power supply could also be required by some sensors. A total of sixteen analog sensors can be connected to the system, currently only three analog pins are occupied. A conversion function code, to convert the value of the ADC reading to a S.I. value must also be written for each new connected sensor. The S.I. value is then added to the produced XML/JSON file that is uploaded onto the webserver. For example:

```
client.println("<newsensor>");
int reading = calculateNewSensorReading();
client.println(reading);
client.println("</newsensor>");
[...]
calculateNewSensorReading()
{
[...]
}
```

The S.I. value can then be parsed by the android app to display the value on a gauge:

```
var
newValueData=xmlDoc.getElementsByTagName("newsensor")[0].childNodes[0]
.nodeValue;
[...]
new RadialGauge({
    renderTo: 'NewGauge',
[...]
var gauge = document.gauges.get('NewGauge');
gauge.value= newValueData;
[...]
```

More Actuators can also be added to extend the functionality of the project. Motorized window shades can also be controlled, using four relays in a H-bridge configuration.

Only ten (4 relays, 1 Serial TX Pin, 1 Serial RX Pin, 4 SPI Pins) of the 54 digital pins are currently occupied. The required code changes for additional relays in the base station is presented below.

```
if (readString.indexOf("?newbtncommand") > 0)
      {
      digitalWrite(newPin, HIGH);
      }
```

Code changes for additional buttons in the Android app, are presented below:

```
<button class="ui-btn" id="newbutton">New Button</button>
```

```
$("#newbutton").click(function(){
      $.get("http://192.168.1.120/?newbtncommand", function(data,
            status){
      });
});
```

Additional changes are required to the cabling -the jumper cables should be replaced with either solder connections, or easy to use connector clips-. The sensors should also be included in separate weather shielded modules.

The cost to acquire an industrial current sensing solution with similar IoT functionalities such as networking and remote control, is around 7000€. (Veris Industries, 2016). Our platform – although not able to measure more than 16 channels at once-, costs a fraction of that amount, around ~100€. Extending our platform, compared to an industrial solution, is also far less pricey. The complete installation guide of the system, is presented in the block diagram below:

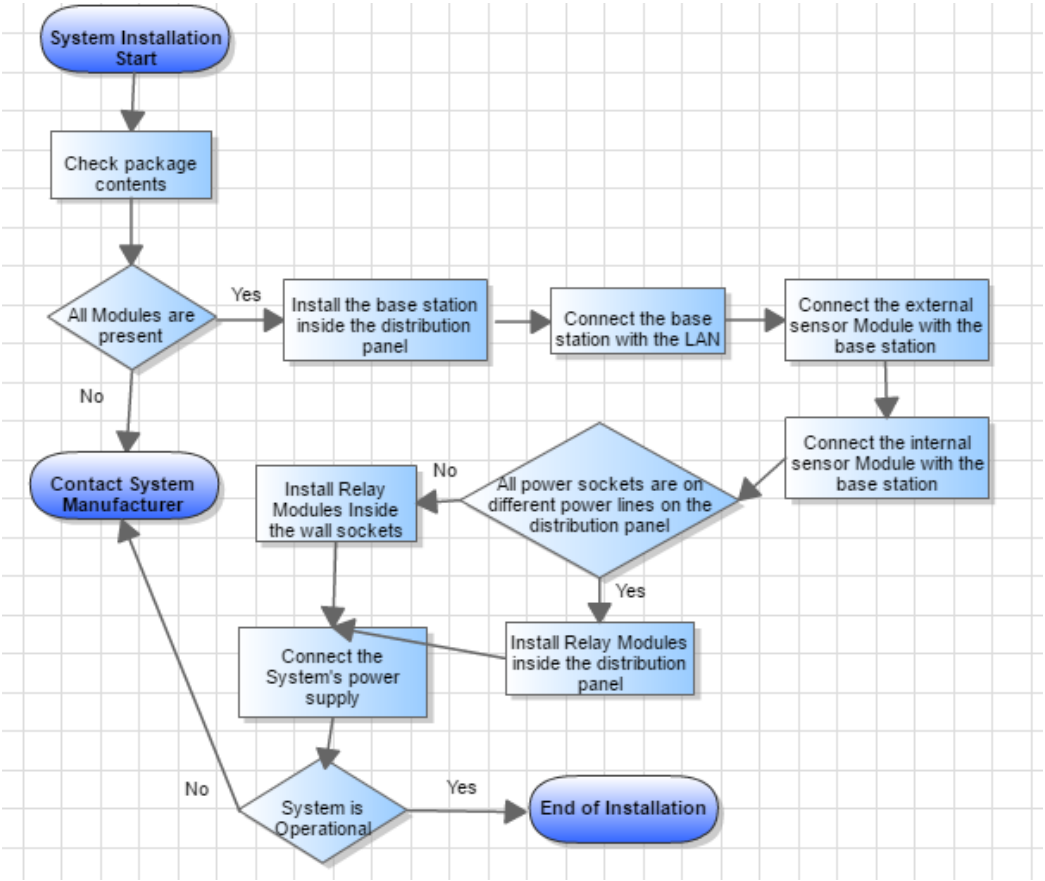

Figure 5-3. System Installation Block Diagram

6. Performance evaluation

The performance of the platform was evaluated in many aspects. First, the accuracy of the current readings were tested against a multimeter reading, and found to be within ±5%. The accuracy of the temperature sensor was tested against the temp probe of the multimeter, and its readings were found to be within ±3%. The illuminance readings were tested against other light sensors, and their calibrated values were within ±5%.

Moreover, the fastest refresh rate of the platform that didn't cause stability issues was found to be one second, an adequate value for real time monitoring. Choosing a crystal above 16Mhz might have reduced further this delay. The platform was found stable for continuous usage, while its power consumption is minimal. The total cost of the platform was ~100 €, meeting the goal for an extremely low cost solution. The bill of materials is presented below:

S/N	Quantity	Product Type	Description	Price
1	1x	Arduino Mega 2560	Development Board	19,9 €
2	1x	Arduino Ethernet Shield W5100	Network Controller	14 €
3	1x	LM35D	Temperature Sensor	1,6 €
4	1x	TEMT6000	Light Sensor	4 €
5	4x	Relay Modules	Relay Actuator	8 €
6	2x	Pin Headers	M/M Headers	0,5 €
7	2x	Jumper Cables	M/M and F/F Cables	8 €
8	1x	Power Supply	PSU 12V 2A	10 €
9	1x	Microbot MR003-009.1	Linear Current Sensor	6 €
10	2x	Breadboards	Prototyping Board	7,1 €
11	1x	Project Box 225x175x80mm	Enclosure Case	6 €
12	1x	Toggle Switch	System Activator	3,5 €
13	1x	LED Strips	LED Lights	20 €
			SUM:	108,6 €

Table 6-1. Bill of Materials: Arduino Mega, Ethernet Shield, LM35D, TEMT6000, Relay Modules, Pin Headers, Jumper Cables, Power Supply, Current Sensor, Breadboards, Box, Switch, LED Strips

The complete base station was packaged inside a project box for continuous usage. The Android app communicates with the base station within 200ms of each request under normal network load.

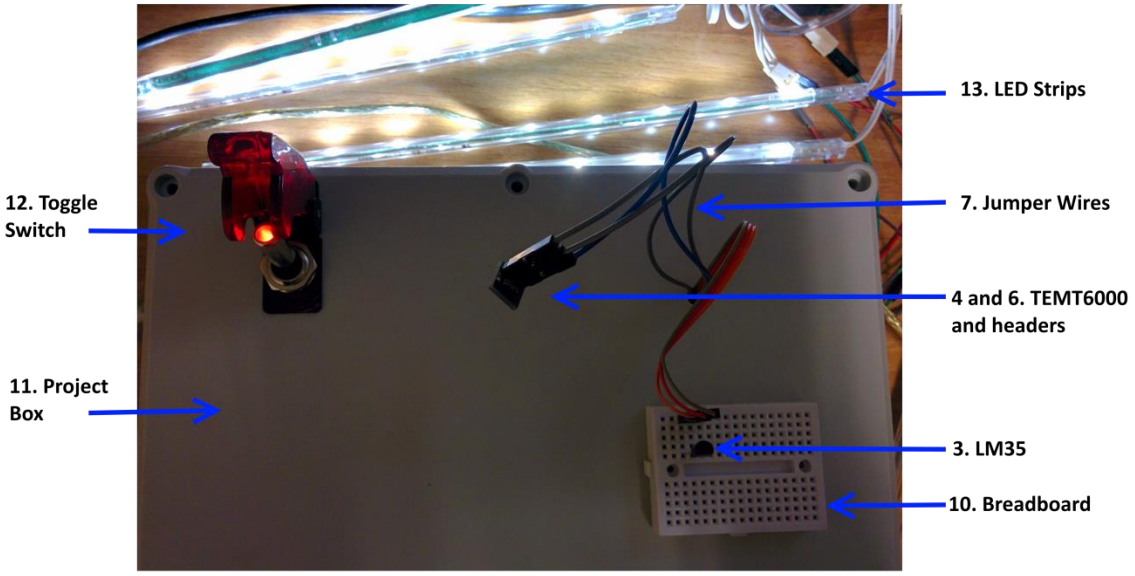

Figure 6-1. Current Sensing platform containing the following materials: Toggle Switch, Project Box, LED Strips, Jumper Wires, TEMT6000 with Headers, LM35, Breadboard

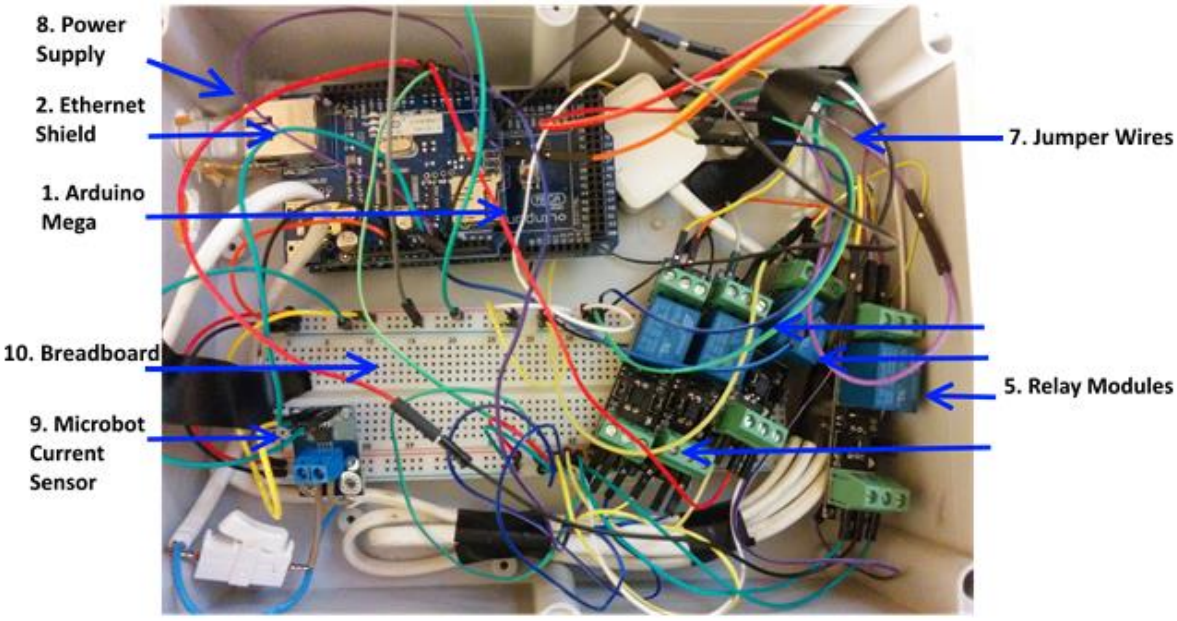

Figure 6-2. Current Sensing platform containing the following materials: Power Supply, Ethernet Shield, Arduino Mega, Breadboard, Microbot Current Sensor, Jumper Wires, Relay Modules

The complete platform accompanied with the android app are presented in the Figure below.

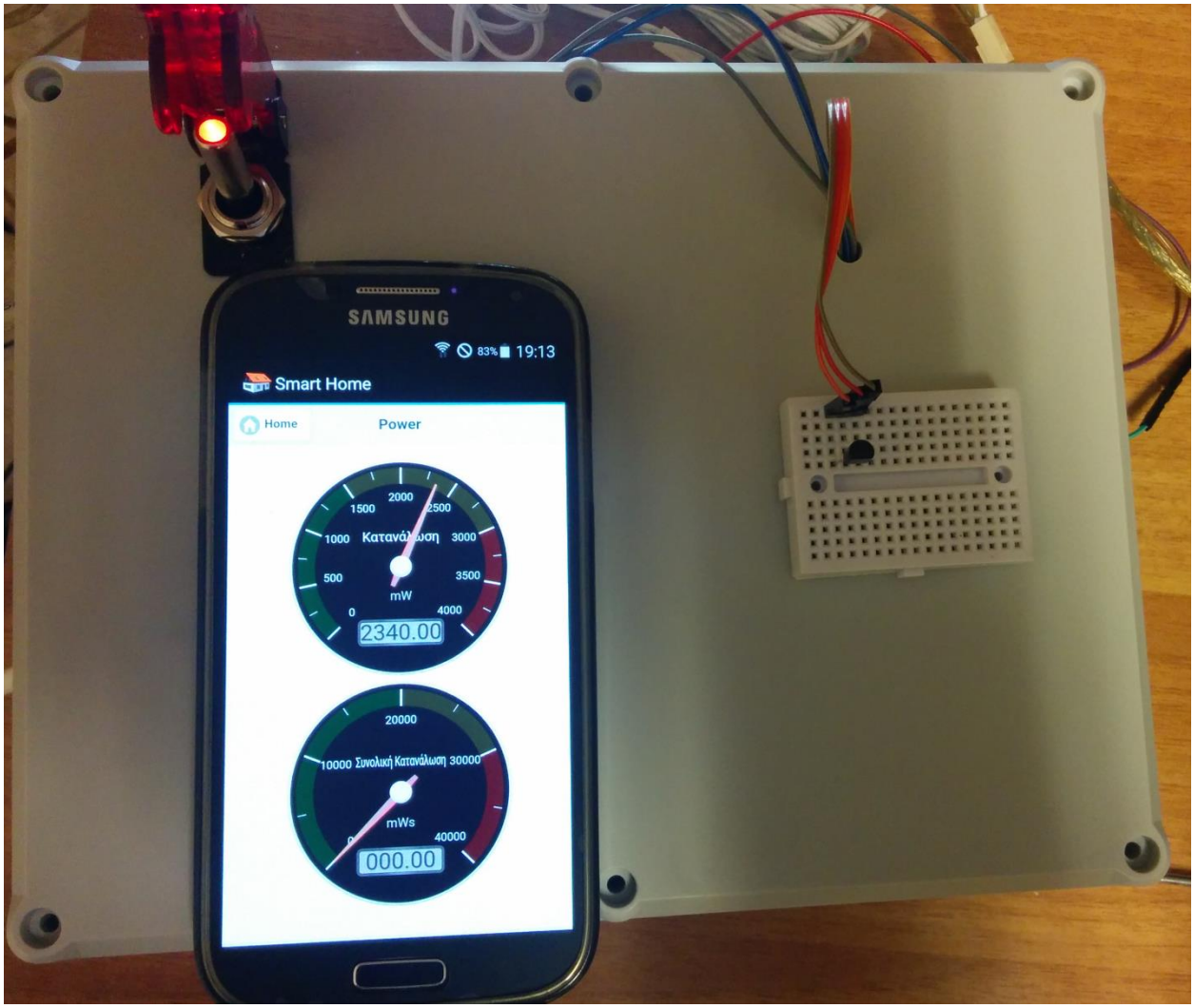

Figure 6-3. Current Sensing Platform

7. Conclusions and Future Work

7.1 Conclusions

In the present study, we propose and create solutions for the following three main issues:

1) How to connect devices in home environment using networking technologies.

2) How to use technologies for communication between home devices using Arduino.

3) How to control and manage home appliances under user's comfort and finance constraints using AI.

4) How to implement and design an industrial prototype.

A home automation system for comfort ability, remote control, and optimal resource utilization is implemented using fuzzy control as artificial intelligence for energy efficiency. Smart home scaled model (of scale 1:100 or less) is equipped with special wiring and network connections, so to enable occupants to remotely control and program an array of automated domestic devices, by issuing simple commands. This system is also consisting of a convenient and easy to understand user interface. From this work it is clear that fuzzy control is emerging as a very useful and applicable technology for Home automation. We can conclude that fuzzy logic is very suitable for heterogeneous home appliances, because fuzzy logic is very robust in headlining computing problems where we have many rules and devices to be addressed. A complete current sensing platform is developed at a very low cost. The developed platform consists of sensing elements and actuators: current sensors, temperature sensors, light sensors and relays, all connected to a microcontroller-based development board. An Ethernet controller is also programmed to offer HTTP connectivity in order to access base station data in real time via the Internet or any local network. An Android app is also developed to access and display sensor data and control the actuators. The developed app can present data graphically in charts, or in specifically designed IoT gauges. The platform is also programmed to support many energy saving automations based on sensor readings: It can automatically turn off all the lights connected on the platform if the luminance inside a room is adequate based on the reading of the sensor, it can automatically disable devices based on their current power consumption, or calculate the total power consumed in a household at a

period of time and automatically disable all devices once the consumption reaches a certain level (for example 40kwh). Last but not least, a complete set of voice actions is developed, using the Speech to Text Android API, in order to remotely control the platform via voice commands.

The development of a low cost platform deemed a big challenge due to the limited computational power and the low ADC resolution of a low cost microcontroller board, as well as the inaccurate readings of low cost sensors. Making a wise choice of components, technologies and methods is necessary for obtaining adequate results.

A significant part of the Thesis is to design and implement an API in order to enable the exchange of data between the Arduino and the Android app. Sampling and filtering sensor data is also a challenge. Due to this fact, statistical methods were used in the sampling algorithms, in order to minimize inaccurate results.

The Android app has been optimized according to the best practices to maintain system stability, reduce power consumption and provide fluid performance and user experience. The Arduino code is also optimized to make the most out of the limited computational power of the ATMega microcontroller.

Familiarizing with the use of IoT libraries for plotting and presenting data, as well as the Speech to Text library for the voice commands in the Android app, also required a significant amount of effort.

The complete platform is built using open source technologies, open source software and hardware, and can be easily customized to fulfill not only a current sensing platform setup, but also a much wider set of applications.

We can conclude from the discussed research work that the use of neural-networks significantly increases the automation of home appliances as well as a very user-friendly solution. We further think that such a user interface is very helpful; especially when it comes to people with disabilities, because there is no need for direct interaction with the system and also it is automatically configurable.

7.2 Future Work

Future relative studies between different algorithms, architectures and serves for developing more cutting edge smart home technologies. Implementation of research projects employing combination of tools and techniques, such as: multi-agent system, action prediction, artificial neural network, and reinforcement learning (Reaz, 2013).

It is well known that predicting future inhabitant's device interaction is needed for home automation. Since the prediction algorithms do not always provide 100% accuracy it should not be used alone for home automation. To avoid wrong prediction we can combine both prediction algorithms and reasoning techniques such as reinforcement-learning, this way we can make sure that unnecessary prediction outcomes are avoided.

While this thesis has demonstrated that it is possible to develop a complete current sensing platform for energy and security management in homes, with a bill of materials around 100 €, many opportunities for extending the features and expanding the scope of this work, remain. This section presents some of these directions.

Extending the platform for use in a more current sensitive environment would require the use of a more expensive development board, containing a microcontroller with a higher resolution ADC. Such an extension would offer better resolution in sensor readings, while at the same time the application would benefit from the upgraded computational power.

The system can also be modified to cooperate with renewable energy sources. The application could be extended to display the system's efficiency, the charging state of the batteries, and optimize the states (charging/discharging) of the system based on user data.

Security vulnerabilities have recently been discovered in many commercial IoT appliances. It is deemed necessary to safeguard all IoT systems from malicious attacks. Security implications of the usage of the current system should be examined.

Last but not least, the platform can also be expanded to gather useful statistical power usage data from a wide variety of users. Such an expansion requires all legal and ethical implications to be examined.

8. Bibliography

Albino Szesz JuniorI, M. M. (n.d.). *Scielo*. Ανάκτηση από Embedded system in Arduino platform with Fuzzy control: http://www.scielo.br/pdf/cr/2016nahead/1678-4596-cr-0103_8478cr20141808.pdf

Allegro Microsystems LLC. (n.d.). *Allergo's Official Page.* Ανάκτηση 09 25, 2016, από ACS711: Hall Effect Linear Current Sensor Datasheet: http://www.allegromicro.com/~/media/Files/Datasheets/ACS711-Datasheet.ashx

Arduino LLC. (n.d.). *Arduino Official Page*. Ανάκτηση Οκτώβριος 10, 2016, από Επίσημη Ιστοσελίδα της Arduino LLC: https://www.arduino.cc/en/Guide/Introduction

Atmel ATMega2560 Datasheet. (n.d.). *Atmel Official Page.* Ανάκτηση 10 05, 2016, από ATMega 2560 Datasheet: http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf

Atmel. (n.d.). *Atmel Official Page.* Ανάκτηση 10 15, 2016, από Atmel Studio 7.0 Release Notes: http://www.atmel.com/tools/atmelstudio.aspx?tab=documents

Canvas Gauges. (n.d.). *Canvas Gauges Official Page*. Ανάκτηση 09 15, 2016, από Canvas Gauges User Guide: https://canvas-gauges.com/documentation/user-guide/

ChartJS. (n.d.). *ChartJS Official Page*. Ανάκτηση 10 05, 2016, από ChartJS Documentation and API: http://www.chartjs.org/docs/

Coolckock. (n.d.). *Coolclock Official Page*. Ανάκτηση 10 01, 2016, από CoolClock - The Javascript Analog Clock: http://randomibis.com/coolclock/

D, N. (1994). *A Fuzzy Perceptron as a Generic Model for Neuro-Fuzzy Approaches.*

F, H. H. (2005). *A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments.*

Framework7. (n.d.). *Framework 7 Official Page.* Ανάκτηση 09 10, 2016, από Framework 7 Documentation: https://framework7.io/docs/

Goldberg, K. H. (2009). *XML.* Berkeley, CA: Peachpit.

Google. (n.d.). *Android Developer Official Page*. Ανάκτηση 10 17, 2016, από Android Studio: https://developer.android.com/studio/index.html

Hagras H, C. M. (2002). *A Fuzzy Incremental Synchronous Learning Technique for Embedded-Agents Learning and Control in Intelligent Inhabited Environments.*

JSON org. (n.d.). *JSON Organization Official Page*. Ανάκτηση 09 02, 2016, από JSON Documentation: http://www.json.org/

Massachusetts Institute of Technology M.I.T. (n.d.). *App Inventor Official Page*. Ανάκτηση 10 17, 2016, από MIT App Inventor: http://appinventor.mit.edu/explore/content/what-app-inventor.html

Mendel J, W. L. (1992). *Generating Fuzzy Rules by Learning through Examples.*

Microbot. (n.d.). *Microbot Official Page.* Ανάκτηση 09 23, 2016, από Microbot MR003-009.1 Datasheet: http://www.microbot.it/documents/mr003-009_datasheet.pdf

Muna, M. (n.d.). *ResearchGate*. Ανάκτηση από Simulation of Fuzzy Logic Control for DC Servo Motor using Arduino based on Matlab/Simulink: https://www.researchgate.net/publication/270884168_Simulation_of_Fuzzy_Logic_Control_for_DC_Servo_Motor_using_Arduino_based_on_MatlabSimulink

National Instruments, N.I. (n.d.). *National Instruments Official Page*. Ανάκτηση 09 23, 2016, από Sensor Terminology: http://www.ni.com/white-paper/14860/en/

Qadeer, S. K. (December 2012). *Application of AI in Home Automation.*

Reaz. (2013). *ARTIFICIAL INTELLIGENCE TECHNIQUES FOR ADVANCED*. Ανάκτηση από http://acta.fih.upt.ro/pdf/2013-2/ACTA-2013-2-07.pdf

S. Sharanya, S. J. (n.d.). *ResearchGate*. Ανάκτηση από Comfort Sensor Using Fuzzy Logic and Arduino: https://www.researchgate.net/publication/309091841_Comfort_Sensor_Using_Fuzzy_Logic_and_Arduino

Songle Relay.,ltd. (n.d.). *Official Songle Relay Page.* Ανάκτηση 09 29, 2016, από Songle Relay Datasheet: http://www.songle.com/pdf/20085271543431001.pdf

Sparkfun. (n.d.). *Sparkfun Official Page*. Ανάκτηση 10 02, 2016, από Arduino Ethernet Shield: https://www.sparkfun.com/products/11166

Texas Instruments T.I. (n.d.). *Texas Instruments Official Page.* Ανάκτηση 10 02, 2016, από LM35D Datasheet: http://www.ti.com/lit/ds/symlink/lm35.pdf

The jQuery Foundation. (n.d.). *jQuery Mobile Official Page*. Ανάκτηση 09 03, 2016, από jQuery Mobile 1.4 API Documentation: http://api.jquerymobile.com/

Veris Industries. (2016, 10 21). *Veris Official Website*. Ανάκτηση από Power / Energy Monitoring Systems: http://veris.com/Category/Power-fslEnergy-spcMonitoring/High-spcDensity/Branch-spcCircuit-spcMonitors/Solid-core.aspx

Vishay LLC. (n.d.). *Vishay Official Page.* Ανάκτηση 17 10, 2016, από TEMT6000x01 Datasheet: www.vishay.com/docs/81579/temt6000.pdf

Wiznet5100 Datasheet. (n.d.). *Wiznet Official Page.* Ανάκτηση 09 25, 2016, από Wiznet 5100 Datasheet: http://www.wiznet.co.kr/wp-content/uploads/wiznethome/Chip/W5100/Document/W5100_Datasheet_v1.2.6.pdf

9. Index

9.1 Screenshots

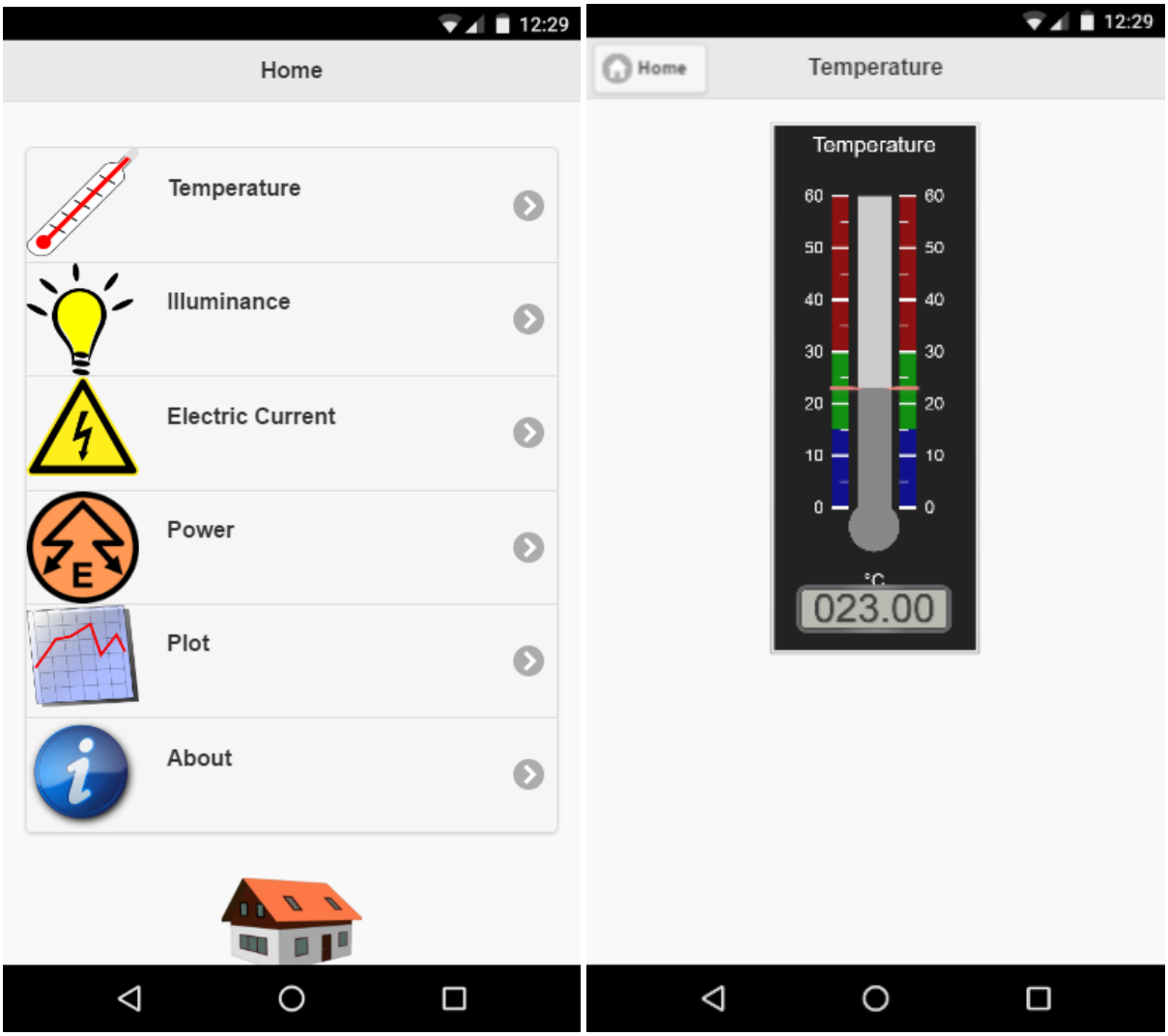

Figure 9-1. Screenshots of Main Menu and Temperature Submenu

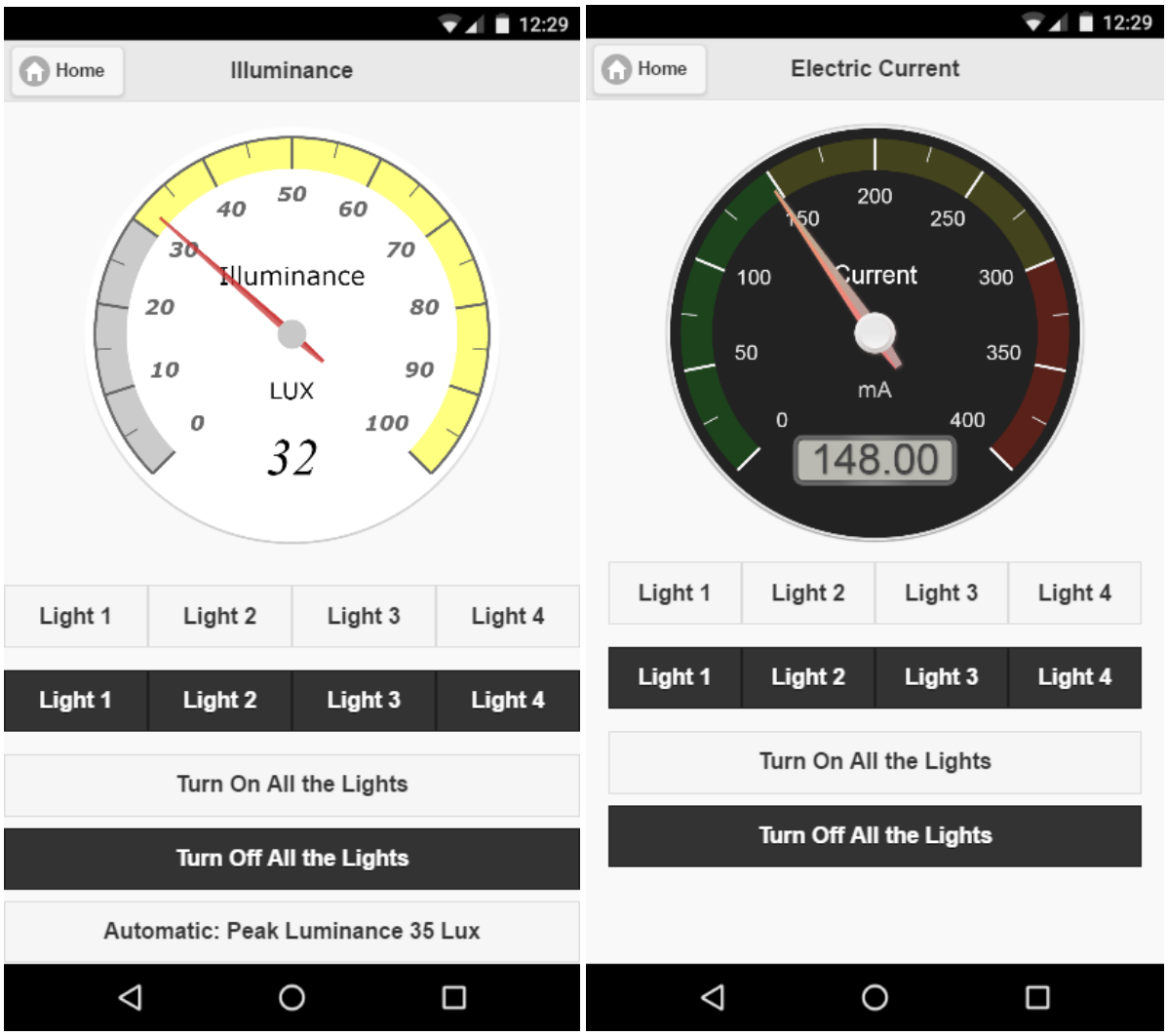

Figure 9-2. Illuminance and Electric Current Submenus

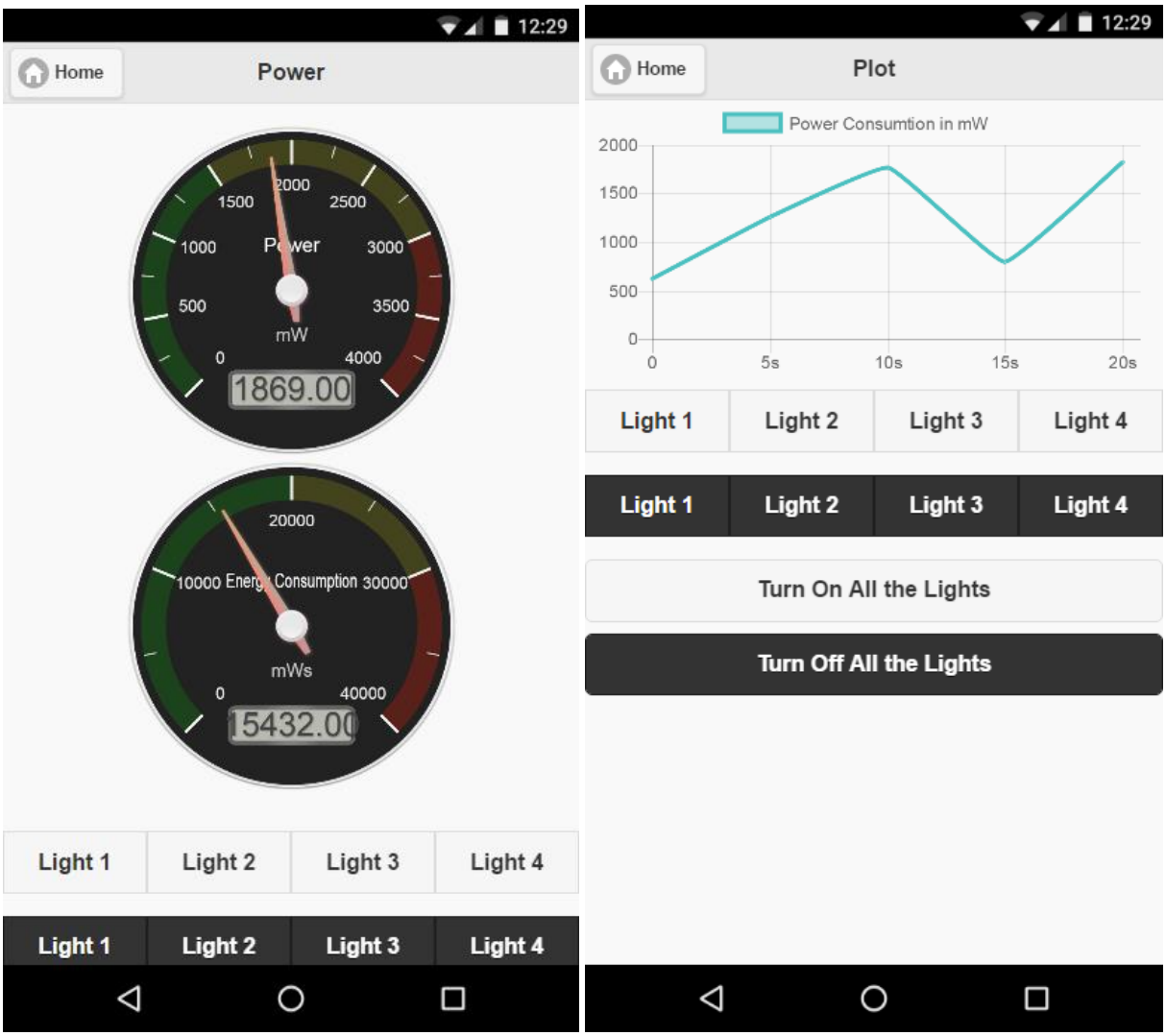

Figure 9-3. Power and Plot Submenus

Figure 9-4. About Submenu

9.2 HTML5 Code

9.2.1 ChartJS Code

```
var data = {
    labels: ["January", "February", "March", "April", "May", "June",
"July"],
    datasets: [
        {
            label: "My First dataset",
            fill: false,
            lineTension: 0.1,
            backgroundColor: "rgba(75,192,192,0.4)",
            borderColor: "rgba(75,192,192,1)",
            borderCapStyle: 'butt',
            borderDash: [],
            borderDashOffset: 0.0,
            borderJoinStyle: 'miter',
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data: [65, 59, 80, 81, 56, 55, 40],
            spanGaps: false,
        }
    ]
};
```

9.2.2 Linear Temperature Gauge Code

```
var gauge = new LinearGauge({
    renderTo: 'canvas-id',
    width: 120,
    height: 400,
```

```
    units: "°C",
    minValue: 0,
    maxValue: 220,
    majorTicks: [
        "0",
        "20",
        "40",
        "60",
        "80",
        "100",
        "120",
        "140",
        "160",
        "180",
        "200",
        "220"
    ],
    minorTicks: 2,
    strokeTicks: true,
    highlights: [
        {
            "from": 100,
            "to": 220,
            "color": "rgba(200, 50, 50, .75)"
        }
    ],
    colorPlate: "#fff",
    borderShadowWidth: 0,
    borders: false,
    needleType: "arrow",
    needleWidth: 2,
    animationDuration: 1500,
    animationRule: "linear",
    tickSide: "left",
    numberSide: "left",
    needleSide: "left",
    barStrokeWidth: 7,
    barBeginCircle: false,
```

```
    value: 75
}).draw();
```

9.2.3 Radial Temperature Gauge Code

```
var gauge = new RadialGauge({
    renderTo: 'canvas-id',
    width: 300,
    height: 300,
    units: "°C",
    title: "Temperature",
    minValue: -50,
    maxValue: 50,
    majorTicks: [
        -50,
        -40,
        -30,
        -20,
        -10,
        0,
        10,
        20,
        30,
        40,
        50
    ],
    minorTicks: 2,
    strokeTicks: true,
    highlights: [
        {
            "from": -50,
            "to": 0,
            "color": "rgba(0,0, 255, .3)"
        },
        {
            "from": 0,
            "to": 50,
            "color": "rgba(255, 0, 0, .3)"
```

```
        }
    ],
    ticksAngle: 225,
    startAngle: 67.5,
    colorMajorTicks: "#ddd",
    colorMinorTicks: "#ddd",
    colorTitle: "#eee",
    colorUnits: "#ccc",
    colorNumbers: "#eee",
    colorPlate: "#222",
    borderShadowWidth: 0,
    borders: true,
    needleType: "arrow",
    needleWidth: 2,
    needleCircleSize: 7,
    needleCircleOuter: true,
    needleCircleInner: false,
    animationDuration: 1500,
    animationRule: "linear",
    colorBorderOuter: "#333",
    colorBorderOuterEnd: "#111",
    colorBorderMiddle: "#222",
    colorBorderMiddleEnd: "#111",
    colorBorderInner: "#111",
    colorBorderInnerEnd: "#333",
    colorNeedleShadowDown: "#333",
    colorNeedleCircleOuter: "#333",
    colorNeedleCircleOuterEnd: "#111",
    colorNeedleCircleInner: "#111",
    colorNeedleCircleInnerEnd: "#222",
    valueBoxBorderRadius: 0,
    colorValueBoxRect: "#222",
    colorValueBoxRectEnd: "#333"
}).draw();
```

9.2.4 Main Menu Code

```
<!DOCTYPE html>
```

```
<html>
<head>

<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>

</head>
<body>

<div data-role="page">

    <div data-role="header" data-theme="a">
        <h1>Home</h1>
    </div><!-- /header -->

    <div role="main" class="ui-content">


<ul data-role="listview" data-inset="true">
    <li>
    <a href="temp.html" data-ajax="false">
        <img src="temp.png">
        <h2>Temperature</h2>
    </a>
    </li>
    <li><a href="fwteinotita.html" data-ajax="false">
        <img src="fws.png">
        <h2>Illuminance</h2>
    </a>
    </li>
    <li><a href="current.html" data-ajax="false">
        <img src="current.png">
        <h2>Electric Current</h2>
    </a>
```

```
    </li>
    <li><a href="power.html" data-ajax="false">
        <img src="power.png">
        <h2>Power</h2>
    </a>
    </li>
    <li><a href="plot.html" data-ajax="false">
        <img src="plot.png">
        <h2>Plot</h2>
    </a>
    </li>
    <li><a href="about.html" data-ajax="false">
        <img src="about.png">
        <h2>About</h2>
    </a>
    </li>
</ul>

    </div><!-- /content -->

      <center>
            <img src="smarthouse.png" width="100">
    </center>

</div><!-- /page -->

</body>
</html>
```

9.2.5 Temperature Submenu Code

```
<!DOCTYPE html>
<html>
<head>

<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="gauge.min.js"></script>
<script>
 function autoRefresh()
{
    window.location = window.location.href;
}
 setInterval('autoRefresh()', 5000);
</script>
</head>
<body>

<div data-role="page">

    <div data-role="header" data-theme="a">
    <a href="main.html" data-ajax="false" data-icon="home">Home</a>
    <h1>Temperature</h1>
    </div><!-- /header -->

    <div role="main" class="ui-content">

    <center>
           <canvas id="gauge1"></canvas>
    </center>
    </div><!-- /content -->

</div><!-- /page -->
```

```
<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var z=xmlDoc.getElementsByTagName("temp")[0].childNodes[0].nodeValue;
</script>
<script>
new LinearGauge({
    renderTo: 'gauge1',
    width: 150,
    height: 380,
    units: '°C',
    title: "Temperature",
    value: 0,
    minValue: 0,
    maxValue: 60,
    majorTicks: [
        '0','10','20','30','40','50','60'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 15, color: 'rgba(0,0,255,0.5)' },
        { from: 15, to: 30, color: 'rgba(0,255,0,0.5)' },
        { from: 30, to: 60, color: 'rgba(255,0,0,0.5)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
```

```
    animationDuration: 300
}).draw();

if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=z;

</script>

</body>
</html>
```

9.2.6 Illuminance Submenu Code

```
<!DOCTYPE html>
<html>
<head>
<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="gauge.min.js"></script>
<script>
$(document).ready(function(){
    $("#b0").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
    });
    $("#b1").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
    });
    $("#b2").click(function(){
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
    });
    $("#b3").click(function(){
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
    });
    $("#b4").click(function(){
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
    });
    $("#b5").click(function(){
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
    });
```

```
    $("#b6").click(function(){
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b7").click(function(){
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b8").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b9").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b10").click(function(){
        $.get("http://192.168.1.120/?afm", function(data, status){
        });
    });
});

</script>
```

```
<script>
function autoRefresh()
{
    window.location = window.location.href;
}
 setInterval('autoRefresh()', 5000);
</script>

</head>
<body>

<div data-role="page">

    <div data-role="header" data-theme="a">
    <a href="main.html" data-ajax="false" data-icon="home">Home</a>
        <h1>Illuminance</h1>
    </div><!-- /header -->

    <div role="main" class="ui-content">

    <center>
    <canvas id="gauge1"></canvas>
    </center>
    </div><!-- /content -->

<div class="ui-grid-c" >
    <div class="ui-block-a" ><button class="ui-btn" id="b0">
      Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn" id="b2">
      Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn" id="b4">
      Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn" id="b6">
      Light 4</button></div>
 </div><!-- /grid-b -->
```

```
<div class="ui-grid-c">
    <div class="ui-block-a" ><button class="ui-btn
     ui-btn-b" id="b1">Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn
     ui-btn-b" id="b3">Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn
     ui-btn-b" id="b5">Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn
     ui-btn-b" id="b7">Light 4</button></div>
</div><!-- /grid-b -->

<button class="ui-btn" id="b8">Turn On All the Lights</button>
<button class="ui-btn ui-btn-b" id="b9">
Turn Off All the Lights</button>
<button class="ui-btn" id="b10">
Automatic: Peak Luminance 35 Lux</button>
</div><!-- /page -->
<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var
q=xmlDoc.getElementsByTagName("illuminance")[0].childNodes[0].nodeValu
e;
</script>
<script>
new RadialGauge({
    renderTo: 'gauge1',
    width: 300,
    height: 300,
    units: 'LUX',
    title: "Illuminance",
    minValue: 0,
    maxValue: 100,
    majorTicks:
['0','10','20','30','40','50','60','70','80','90','100'],
    minorTicks: 2,
```

```
    ticksAngle: 270,
    startAngle: 45,
    strokeTicks: true,
    highlights  : [
        { from : 0,  to : 30, color : 'rgba(150, 150, 150, 0.5)' },
        { from : 30, to : 100, color : 'rgba(255, 255, 0, 0.5)' }
    ],
    valueInt: 1,
    valueDec: 0,
    colorPlate: "#fff",
    colorMajorTicks: "#686868",
    colorMinorTicks: "#686868",
    colorTitle: "#000",
    colorUnits: "#000",
    colorNumbers: "#686868",
    valueBox: true,
    colorValueText: "#000",
    colorValueBoxRect: "#fff",
    colorValueBoxRectEnd: "#fff",
    colorValueBoxBackground: "#fff",
    colorValueBoxShadow: false,
    colorValueTextShadow: false,
    colorNeedleShadowUp: true,
    colorNeedleShadowDown: false,
    colorNeedle: "rgba(200, 50, 50, .75)",
    colorNeedleEnd: "rgba(200, 50, 50, .75)",
    colorNeedleCircleOuter: "rgba(200, 200, 200, 1)",
    colorNeedleCircleOuterEnd: "rgba(200, 200, 200, 1)",
    borderShadowWidth: 0,
    borders: true,
    borderInnerWidth: 0,
    borderMiddleWidth: 0,
    borderOuterWidth: 5,
    colorBorderOuter: "#fafafa",
    colorBorderOuterEnd: "#cdcdcd",
    needleType: "arrow",
    needleWidth: 2,
    needleCircleSize: 7,
```

```
    needleCircleOuter: true,
    needleCircleInner: false,
    animationDuration: 1500,
    animationRule: "dequint",
    fontNumbers: "Verdana",
    fontTitle: "Verdana",
    fontUnits: "Verdana",
    fontValue: "Led",
    fontValueStyle: 'italic',
    fontNumbersSize: 20,
    fontNumbersStyle: 'italic',
    fontNumbersWeight: 'bold',
    fontTitleSize: 24,
    fontUnitsSize: 22,
    fontValueSize: 50,
    animatedValue: true
}).draw();

if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=q;

</script>

</body>
</html>
```

9.2.7 Electric Current Submenu Code

```
<!DOCTYPE html>
<html>
<head>
<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="gauge.min.js"></script>
<script>
$(document).ready(function(){
    $("#b0").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
    });
    $("#b1").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
    });
    $("#b2").click(function(){
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
    });
    $("#b3").click(function(){
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
    });
    $("#b4").click(function(){
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
    });
    $("#b5").click(function(){
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
    });
```

```
    $("#b6").click(function(){
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b7").click(function(){
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b8").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b9").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
});
</script>
<script>
 function autoRefresh()
{
    window.location = window.location.href;}

 setInterval('autoRefresh()', 5000);
</script>
</head>
```

```
<body>

<div data-role="page">

    <div data-role="header" data-theme="a">
    <a href="main.html" data-ajax="false" data-icon="home">Home</a>
        <h1>Electric Current</h1>
    </div><!-- /header -->

    <div role="main" class="ui-content">

    <center>
    <canvas id="gauge1"></canvas>

    </center>
    <div class="ui-grid-c" >
    <div class="ui-block-a" ><button class="ui-btn" id="b0">
     Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn" id="b2">
     Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn" id="b4">
     Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn" id="b6">
     Light 4</button></div>
    </div><!-- /grid-b -->

    <div class="ui-grid-c">
    <div class="ui-block-a" ><button class="ui-btn
     ui-btn-b" id="b1">Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn
     ui-btn-b" id="b3">Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn
     ui-btn-b" id="b5">Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn
     ui-btn-b" id="b7">Light 4</button></div>
    </div><!-- /grid-b -->
```

```
      <button class="ui-btn" id="b8">Turn On All the Lights</button>
      <button class="ui-btn ui-btn-b" id="b9">Turn Off All the
      Lights</button>
    </div><!-- /content -->

</div><!-- /page -->
<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var
x=xmlDoc.getElementsByTagName("milliamps")[0].childNodes[0].nodeValue;
</script>
<script>
new RadialGauge({
    renderTo: 'gauge1',
    width: 300,
    height: 300,
    units: 'mA',
    title: "Current",
    value: 0,
    minValue: 0,
    maxValue: 400,
    majorTicks: [
        '0','50','100','150','200','250','300','350','400'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 150, color: 'rgba(0,255,0,.15)' },
        { from: 150, to: 300, color: 'rgba(255,255,0,.15)' },
        { from: 300, to: 400, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
```

```
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();


if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
var gauge = document.gauges.get('gauge1');
gauge.value=x;

</script>

</body>
</html>
```

9.2.8 Power Submenu Code

```
<!DOCTYPE html>
<html>
<head>
<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="gauge.min.js"></script>
<script>
$(document).ready(function(){
    $("#b0").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
    });
    $("#b1").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
    });
    $("#b2").click(function(){
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
    });
    $("#b3").click(function(){
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
    });
    $("#b4").click(function(){
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
    });
    $("#b5").click(function(){
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
    });
```

```
    $("#b6").click(function(){
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b7").click(function(){
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b8").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b9").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b10").click(function(){
        $.get("http://192.168.1.120/?awm", function(data, status){
        });
    });
    $("#b11").click(function(){
        $.get("http://192.168.1.120/?awhm", function(data, status){
        });
    });
});
```

```
</script>

<script>

 function autoRefresh()
{
    window.location = window.location.href;
}

 setInterval('autoRefresh()', 5000);
</script>
</head>
<body>

<div data-role="page">

    <div data-role="header" data-theme="a">
    <a href="main.html" data-ajax="false" data-icon="home">Home</a>
        <h1>Power</h1>
    </div><!-- /header -->

    <div role="main" class="ui-content">

    <center>
           <canvas id="gauge2"></canvas>
           <canvas id="gauge3"></canvas>
    </center>
    </div><!-- /content -->
    <div class="ui-grid-c" >
    <div class="ui-block-a" ><button class="ui-btn" id="b0">
     Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn" id="b2">
     Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn" id="b4">
     Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn" id="b6">
     Light 4</button></div>
    </div><!-- /grid-b -->
```

```
    <div class="ui-grid-c">
    <div class="ui-block-a" ><button class="ui-btn
     ui-btn-b" id="b1">Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn
     ui-btn-b" id="b3">Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn
     ui-btn-b" id="b5">Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn
     ui-btn-b" id="b7">Light 4</button></div>
    </div><!-- /grid-b -->

<button class="ui-btn ui-corner-all" id="b8">
Turn On All the Lights</button>
<button class="ui-btn ui-btn-b ui-corner-all" id="b9">
Turn Off All the Lights</button>
<button class="ui-btn ui-btn-a ui-corner-all" id="b10" >
Automatic 1: Peak Output Power 1500mW</button>
<button class="ui-btn ui-btn-b ui-corner-all" id="b11">
Automatic 2: Total Power Consumption 30.000mWs</button>

</div><!-- /page -->
<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var
y=xmlDoc.getElementsByTagName("milliwatts")[0].childNodes[0].nodeValue
;
var
z=xmlDoc.getElementsByTagName("milliwattssec")[0].childNodes[0].nodeVa
lue;
</script>
<script>
```

```
new RadialGauge({
    renderTo: 'gauge2',
    width: 235,
    height: 235,
    units: 'mW',
    title: "Power",
    value: 0,
    minValue: 0,
    maxValue: 4000,
    majorTicks: [
        '0','500','1000','1500','2000','2500','3000','3500','4000'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 1500, color: 'rgba(0,255,0,.15)' },
        { from: 1500, to: 3000, color: 'rgba(255,255,0,.15)' },
        { from: 3000, to: 4000, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();

new RadialGauge({
    renderTo: 'gauge3',
    width: 235,
    height: 235,
    units: 'mWs',
    title: "Energy Consumption",
```

```
    value: 0,
    minValue: 0,
    maxValue: 40000,
    majorTicks: [
        '0','10000','20000','30000','40000'
    ],
    minorTicks: 2,
    strokeTicks: false,
    highlights: [
        { from: 0, to: 20000, color: 'rgba(0,255,0,.15)' },
        { from: 20000, to: 30000, color: 'rgba(255,255,0,.15)' },
        { from: 30000, to: 40000, color: 'rgba(255,30,0,.25)' }
    ],
    colorPlate: '#222',
    colorMajorTicks: '#f5f5f5',
    colorMinorTicks: '#ddd',
    colorTitle: '#fff',
    colorUnits: '#ccc',
    colorNumbers: '#eee',
    colorNeedle: 'rgba(240, 128, 128, 1)',
    colorNeedleEnd: 'rgba(255, 160, 122, .9)',
    valueBox: true,
    animationRule: 'bounce',
    animationDuration: 300
}).draw();
if (!window.addEventListener) {
    window.addEventListener = function(evt, listener) {
        window.attachEvent('on' + evt, listener);
    };
}
if (!Array.prototype.forEach) {
    Array.prototype.forEach = function(cb) {
        var i = 0, s = this.length;
        for (; i < s; i++) {
            cb && cb(this[i], i, this);
        }
    }
}
```

```
var gauge = document.gauges.get('gauge2');
gauge.value=y;
var gauge = document.gauges.get('gauge3');
gauge.value=z;

</script>

</body>
</html>
```

9.2.9 Plot Submenu Code

```
<!doctype html>
<html>

<head>

<title>Thesis</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="gauge.min.js"></script>
<script src="Chart.bundle.js"></script>
<script src="Chart.bundle.min.js"></script>
<script src="Chart.js"></script>
<script src="Chart.min.js"></script>
<script src="jquery.min.js"></script>

<script>
$(document).ready(function(){
    $("#b0").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
    });
    $("#b1").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
    });
    $("#b2").click(function(){
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
    });
    $("#b3").click(function(){
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
    });
```

```
    $("#b4").click(function(){
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
    });
    $("#b5").click(function(){
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
    });
    $("#b6").click(function(){
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b7").click(function(){
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
    $("#b8").click(function(){
        $.get("http://192.168.1.120/?laon", function(data, status){
        });
        $.get("http://192.168.1.120/?lbon", function(data, status){
        });
        $.get("http://192.168.1.120/?lcon", function(data, status){
        });
        $.get("http://192.168.1.120/?ldon", function(data, status){
        });
    });
    $("#b9").click(function(){
        $.get("http://192.168.1.120/?laoff", function(data, status){
        });
        $.get("http://192.168.1.120/?lboff", function(data, status){
        });
        $.get("http://192.168.1.120/?lcoff", function(data, status){
        });
        $.get("http://192.168.1.120/?ldoff", function(data, status){
        });
    });
});
```

```
</script>
<script>
 function autoRefresh()
{
    window.location = window.location.href;
}
 setInterval('autoRefresh()', 5000);
</script>

<body>
<div data-role="header" data-theme="a">
<a href="main.html" data-icon="home">Home</a>
        <h1>Plot</h1>
    </div><!-- /header -->
    <div style="width:100%;">
        <canvas id="myChart"></canvas>
    </div>

    <div class="ui-grid-c" >
    <div class="ui-block-a" ><button class="ui-btn" id="b0">
     Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn" id="b2">
     Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn" id="b4">
     Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn" id="b6">
     Light 4</button></div>
    </div><!-- /grid-b -->

    <div class="ui-grid-c">
    <div class="ui-block-a" ><button class="ui-btn
     ui-btn-b" id="b1">Light 1</button></div>
    <div class="ui-block-b"><button class="ui-btn
     ui-btn-b" id="b3">Light 2</button></div>
    <div class="ui-block-c"><button class="ui-btn
     ui-btn-b" id="b5">Light 3</button></div>
    <div class="ui-block-d"><button class="ui-btn
     ui-btn-b" id="b7">Light 4</button></div>
```

```
    </div><!-- /grid-b -->

<button class="ui-btn ui-corner-all" id="b8">
Turn On All the Lights</button>
<button class="ui-btn ui-btn-b ui-corner-all" id="b9">
Turn Off All the Lights</button>

<script>
var xmlhttp, xmlDoc;
xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "http://192.168.1.120", false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;
var q=xmlDoc.getElementsByTagName("m0")[0].childNodes[0].nodeValue;
var w=xmlDoc.getElementsByTagName("m1")[0].childNodes[0].nodeValue;
var e=xmlDoc.getElementsByTagName("m2")[0].childNodes[0].nodeValue;
var r=xmlDoc.getElementsByTagName("m3")[0].childNodes[0].nodeValue;
var t=xmlDoc.getElementsByTagName("m4")[0].childNodes[0].nodeValue;
</script>
 <script>
var ctx = document.getElementById("myChart");
var myChart = new Chart(ctx, {
    type: 'line',
    data: {
        labels: ["0", "5s", "10s", "15s", "20s"],
        datasets: [{
                    label: "Power Consumtion in mW",
            fill: false,
            lineTension: 0.1,
            backgroundColor: "rgba(75,192,192,0.4)",
            borderColor: "rgba(75,192,192,1)",
            borderCapStyle: 'butt',
            borderDash: [],
            borderDashOffset: 0.0,
            borderJoinStyle: 'miter',
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
```

```
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data: [q, w, e, r, t],
            spanGaps: false,
        }]
    },
    options: {
        scales: {
            yAxes: [{
                ticks: {
                    beginAtZero:true
                }
            }]
        }
    }
});
</script>
</body>

</html>
```

9.2.10 About Submenu Code

```
<!DOCTYPE html>
<html>
<head>
<title>My app</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="jquery.mobile-1.4.5.min.css" />
<script src="jquery-1.11.1.min.js"></script>
<script src="jquery.mobile-1.4.5.min.js"></script>
<script src="coolclock.js"></script>
<script src="moreskins.js"></script>
<script src="excanvas.js"></script>
</head>
<body>

<div data-role="page">
<div data-role="header" data-theme="a">
<a href="main.html" data-ajax="false" data-icon="home">Home</a>
        <h1>About</h1>
</div><!-- /header -->
<div role="main" class="ui-content">
<p>Foremost, I would like to express my sincere gratitude to my
advisor Prof. Mr Kalaitzakis Konstantinos for the continuous support
of my research, his excellent cooperation and for all of the
opportunities I is given to conduct my thesis.</p>
<p>Besides my advisor, I would like to thank the rest of my thesis
committee: Prof. Stavrakakis Georgios, and Dr. Sergaki Eleftheria.</p>
<p>Last but not least, I would like to thank my parents and my friends
for supporting me throughout my studies.</p>
<center>
    <canvas id="clk2" style="display:block;"
    class="CoolClock:Sun"></canvas>
</center>
</div><!-- /content -->
</div><!-- /page -->

</body>
</html>
```

9.3 Arduino Code

```
#include <SPI.h>
#include <Ethernet.h>

bool autofmode = false;
bool autowmode = false;
bool autowhmode = false;
int arrayMw[]={0,0,0,0,0};
int pointer =0;

long starttime;
long sumwh = 0;

byte mac[] = {0x56, 0x49, 0x23, 0x20, 0x25, 0x26 };
IPAddress ip(192, 168, 1, 120);


EthernetServer server(80);
String readString;

void setup()
{
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);

  Ethernet.begin(mac, ip);

  server.begin();
  Serial.begin(9600);

  initializeLeds();
}
```

```
void loop()
{
  EthernetClient client = server.available();
  if (client)

  {

    while (client.connected())
    {
      if (client.available())

      {
        char c = client.read();

        if (readString.length() < 100)

        {
          readString += c;
          Serial.write(c);
          if (c == '\n') {
            client.println("HTTP/1.1 200 OK");
            client.println("Server: Arduino");
            client.println("Access-Control-Allow-Origin: *");
            client.println("Connection: close");
            client.println("Content-Type: text/xml\r\n");
            client.println("<xml>");
            client.println("<milliamps>");
            int metrisi = calculateAmperage();
            if (metrisi <= 15) {
              metrisi = 0;
            }
            client.println(metrisi);
            client.println("</milliamps>");
            client.println("<milliwatts>");
            int mwatt = calculateWattage(metrisi);
            client.println(mwatt);
```

```
        if (autowmode == true) {
           Serial.println("inside true");
           if (mwatt >= 1500) {
             Serial.println("inside oria");
             digitalWrite(2, LOW);
             digitalWrite(3, LOW);
             digitalWrite(4, LOW);
             digitalWrite(5, LOW);
             autowmode = false;
           }
         }
         client.println("</milliwatts>");
         if (pointer==5){
             arrayMw[0]=arrayMw[1];
             arrayMw[1]=arrayMw[2];
             arrayMw[2]=arrayMw[3];
             arrayMw[3]=arrayMw[4];
//gia na einai pio katanoito ston meso xristi, ennoeitai ginetai
kai me for
           pointer=4;
         }
         arrayMw[pointer]=mwatt;
         pointer++;
         client.println("<m0>");
         client.println(arrayMw[0]);
         client.println("</m0>");
         client.println("<m1>");
         client.println(arrayMw[1]);
         client.println("</m1>");
         client.println("<m2>");
         client.println(arrayMw[2]);
         client.println("</m2>");
         client.println("<m3>");
         client.println(arrayMw[3]);
         client.println("</m3>");
         client.println("<m4>");
         client.println(arrayMw[4]);
         client.println("</m4>");
```

```
client.println("<milliwattssec>");
if (autowhmode == true) {
  Serial.println("inside autowhmode");
  long metr = calculatemwh(mwatt);
  client.println(metr);
  Serial.println(metr);
  if (metr >= 30000) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    sumwh=0;
    autowhmode = false;
  }
}
else {
  client.println("0");
}
client.println("</milliwattssec>");
client.println("<temp>");
int temp = calculateTemperature();
client.println(temp);
client.println("</temp>");
client.println("<illuminance>");
int illuminance = calculateLux();
client.println(illuminance);
if (autofmode == true) {
  if (illuminance >= 35) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    sumwh = 0;
    autofmode = false;
  }
}
client.println("</illuminance>");
client.println("</xml>");
```

```
 delay(1);
 //stopping client
 client.stop();

 if (readString.indexOf("?laon") > 0) //checks for on
 {
   digitalWrite(2, HIGH);
 }
 else if (readString.indexOf("?laoff") > 0)
//checks for off
 {
   digitalWrite(2, LOW);
 }
 else if (readString.indexOf("?lbon") > 0)
 {
   digitalWrite(3, HIGH);
 }
 else if (readString.indexOf("?lboff") > 0)
 {
   digitalWrite(3, LOW);
 }
 else if (readString.indexOf("?lcon") > 0)
 {
   digitalWrite(4, HIGH);
 }
 else if (readString.indexOf("?lcoff") > 0)
 {
   digitalWrite(4, LOW);
 }
 else if (readString.indexOf("?ldon") > 0)
 {
   digitalWrite(5, HIGH);
 }
 else if (readString.indexOf("?ldoff") > 0)
 {
   digitalWrite(5, LOW);
 }
```

```
else if (readString.indexOf("?leon") > 0)
{
  digitalWrite(2, HIGH);
  digitalWrite(3, HIGH);
  digitalWrite(4, HIGH);
  digitalWrite(5, HIGH);
  delay(10);
}
else if (readString.indexOf("?leoff") > 0)
{
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  delay(10);
}
else if (readString.indexOf("?afm") > 0)
{
  autofmode = true;
}
else if (readString.indexOf("?awm") > 0)
{
  Serial.println("inside ?awm");
  autowmode = true;
}
else if (readString.indexOf("?awhm") > 0)
{
  Serial.println("inside ?awhm");
  autowhmode = true;

}

readString = "";

delay(1);
client.stop();
```

```
          }
        }
      }
    }
  }
}

int calculateAmperage() {
  long sum = 0;
  for (int i = 0; i <= 100; i++) {
    int reading = analogRead(A0);
    sum += reading;
  }
  int mo = sum / 101;
  Serial.println(mo);
  int amperagema = (mo - 501) * 13;
  //Serial.println(amperagema);
  return amperagema;
}

int calculateWattage(int amps) {
  int ipologismos = amps * 12;
  return ipologismos;
}

int calculateTemperature() {
  long sum = 0;
  for (int i = 0; i <= 10; i++) {
    int thermokrasia = (5.0 * analogRead(A1) * 100.0) / 1024;
//thermokrasia se kelsiou
    sum += thermokrasia;
  }
  int mo = sum / 11;
  return mo;
}
```

```
int calculateLux() {
  int illuminance = analogRead(A2) * 0.9765625;  //fwteinotita se lux
  //Serial.println(illuminance);
  return illuminance;

}

void initializeLeds() {
  for (int i = 2; i <= 5; i++) {
    digitalWrite(i, HIGH);
  }
}

long calculatemwh(int mw) {
  sumwh += mw * 5;
  Serial.println(sumwh);
  return sumwh;
}
```