Wireless Web-based Control and Monitoring System for the Security of a Remote House

Charikleia Vlachou

Master of Science in Network Computing

T.E.I. OF LARISSA

STAFFORDSHIRE UNIVERSITY

TECHNOLOGICAL EDUCATIONAL INSTITUTE OF LARISSA

June 2014

TECHNOLOGICAL EDUCATIONAL INSTITUTE OF LARISSA

Wireless Web-based Control and Monitoring System for the Security of a Remote House

Charikleia Vlachou

Master of Science in Network Computing, 2014

Thesis Summary

The aim of the thesis is the designing and implementation of a remote house security and alarming system using ZigBee wireless sensors during the absence of the owner. The control and monitoring of the system will be accomplished via a web interface and furthermore will be implemented a Windows Phone application for the same reason. This system targets to inform the user for the condition of the remote house. Any change on the system triggers an alert and inform the owner by an e-mail or an SMS. The system can support detection of smoke, temperature variation, any intrusion on the house from i.e. windows, doors etc. Also a wireless camera will be used for the surveillance of the house. In the case of intrusion detection, the camera will be programmed to capture images of the intruder and send them to the owner, the neighbors (neighborhood alert) and the related authority.

Keywords: home security, remote monitoring, remote control, ZigBee, wireless sensors, intrusion alarms, windows phone application, XBee, arduino, responsive, bootstrap.

Acknowledgements

First of all, I would like to thank you my advisor for the ideas and the guidance to specify the exact theme that I had to deal with. I am grateful to my friends who supported me in any respect during the completion of the project. Last but not least, I thank my family for all their psychological support and understanding throughout all this period of my studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Nowadays with the increasing of the theft more and more people buy home security systems for their own safety and the security of their home against intruders, fire, gas leakage etc. Existing systems for security are passive and costly. The proposed security system aims to provide security in such a way that user can monitor his/her home and at the same time can control any bad situation occurring at home when they absent.

There are a number of studies which says that Wireless sensor networks are a cost effective solution for collecting, receiving and transmitting data. The proposed system will be consisting of wireless devices to reduce the energy consumption and increase the safety in the case of power failure. Also the devices will communicate with each other with ZigBee standard which offers reliable, cost-effective, low-power wireless communication. It is foreseen that ZigBee technology will be embedded in a wide range of products and applications across consumer, commercial, industrial and government markets worldwide.

The security system supports camera surveillance. It worth's mentioned that in the case of intruder detection the security system informs not only the owner but also the neighbors and the nearest polish station by sending a captured image of the intruder.

The proposed system offers a solution with the state of the art in the wireless networks and communication technologies using ZigBee technology for the wireless network, and

control and monitoring of the remote house through the responsive web site or through a Windows Phone Application that will be implemented.

1.2 Aims of the Thesis

1. The aim of this project is the designing and implementation of a remote house security and alarming system using Zigbee wireless sensors during the absence of the owner.
2. The control and monitoring of the system will be accomplished via
    a. a user friendly web site,
    b. a Windows Phone application for the same reason.
3. The web site will be responsive to facilitate users to access the services if the usage of the application is not possible and that because the windows phone application that is implemented does not run to android smart phones or other operating systems.
4. The system targets to inform the user for the condition of the remote house. There will be sensors that send on a predefined time data to the server. For the same reason there will be implemented alerts that will be activated in any alteration of the system.
5. When an alert occurs then an e-mail will be sent from the server, a push notification and a SMS (depending on the options of the user) to inform the owner for the situation.
6. The system can support detection of smoke, temperature variations, humidity, motion detection, any intrusion on the house from i.e. windows, doors etc.
7. A wireless camera will be used for the surveillance of the house. In the case of intrusion detection, the camera will be programmed to capture images of the intruder and send them to the owner, the neighbors (neighborhood alert) and the related authority.

1.3 Research Methodology

1.3.1 Literature Review

The literature related to this Thesis includes knowledge about zigbee technology, arduino and electronics basic knowledge for the creation of the prototype. The main source of information comes from the official web pages of manufacturers such as zigbee Allianz Digi, Arduino and Sparkfun that provide hardware specifications, datasheets, tutorials, a number of presentations about its devices, forums and communities.

For information about the implementation of the web site and the windows phone app the main source of information will be the MSDN which includes how-to and reference documentations, sample codes, technical articles etc. As secondary sources are the w3scools for the designing of the web site and bootstrap official page to make it responsive.

Additionally, searching on the internet will help to find many other papers from several universities, conferences and journals.

1.3.2 Analysis and Investigation

1.3.2.1 Prototyping

A prototype was constructed for the need of the thesis. It could be divided in two main parts the Home which is the wireless sensor network and the Server side which is responsible for maintaining the received data, control and monitoring. The prototype will be analyzed in chapter 3.

1.3.2.2 System Evaluation

The system was tested according to several criteria. The most important is to retrieve correct data and get the corresponding alerts. For testing the sensors, were created the appropriate conditions e.g. smoke production with a cigarette to trigger the smoke alert or increase the temperature with a lighter.

1.3 Novel Features of the Thesis

The novel aspects of this thesis are:

- The creation of a prototype with low-cost and low power consumption as an alternative way to the already existed expensive security systems.
- The idea of “Neighborhood Alert” to inform the neighbors, who are registered on the system, about the intruder by sending them a captured image and notifications. In the case of emergency, if the owner will not react in the first minutes of the alert then the system informs with email or sms all the users with role “admin”.

1.4 Outline of the Thesis

The thesis is organized as follows: Chapter 2 provides all the necessary background on zigbee technology, electronics, XBee radios and Arduino microcontroller boards. Chapter 3 is analyzed the proposed system. Which types of sensors were used, the configuration of the XBees and the communication with arduino, and furthermore about the design and development of the web site and windows phone application. Chapter 4 is the last part where, an overall description about the aims of the thesis, the evaluation and proposals for future work were mentioned.

Chapter 2

Literature Review

2.1 Introduction

Since the start of the third Millennium, wireless sensor networks (WSNs) generated an increasing interest from industrial and research perspectives (Buraty 2009). Nowadays the Wireless Sensor networks (WSNs) become an increasingly popular technology for low-cost and low power consumption applications. Their importance has been enforced after the delivery of the IEEE 802.15.4 standard for the physical and MAC layers and the release of Zigbee standard for the network and application layers (Baronty).

A wireless sensor network can be described as a set of individual devices, denoted as nodes, that are able to interact with the environment by sensing or controlling physical parameters. Wireless communication enables the cooperation of the nodes to fulfill bigger tasks that single nodes could not (Hogrefe, 2011). The collected data is forwarded, possibly via multiple hops, to a sink (sometimes denoted as controller or monitor) that can use it locally or is connected to other networks (e.g., the Internet) through a gateway. The nodes can be stationary or moving. They can be aware of their location or not. They can be homogeneous or not (Burrati, 2009).

Among the applications of wireless sensor networks are the following:

- Industrial control and monitoring
- Lighting control, HVAC, machinery
- Environmental monitoring
- Fire, pollution, wild life, agriculture, etc.
- Health monitoring

- Traffic monitoring and control
- Building structures monitoring
- Security and military sensing

The main features of WSNs include: scalability with respect to the number of nodes in the network, self-organization, self-healing, energy efficiency, a sufficient degree of connectivity among nodes, low-complexity, low cost and size of nodes. Those protocol architectures and technical solutions providing such features can be considered as a potential framework for the creation of these networks, but, unfortunately, the definition of such a protocol architecture and technical solution is not simple, and the research still needs to work on it (Buratti, 2009, Sohrabi, 2000).

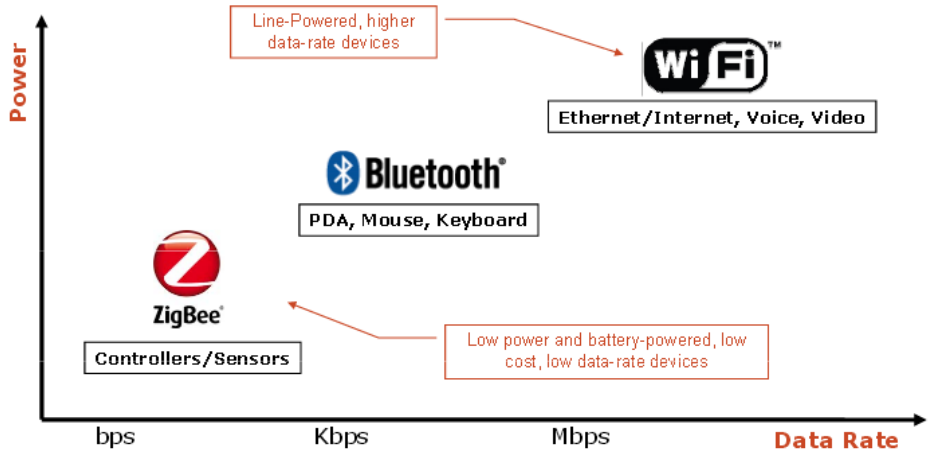

Figure 2.1: Comparison diagram of Zigbee, Bluetooth, WiFi technologies.

The ZigBee protocol is a very popular way of creating radio sensor networks for a number of reasons according to Faludi (2011). Wireless networks and connected devices in general tend to be used in situations where power is hard to come by and must be conserved. Many times the communications these networks send are small in nature, compared to systems that transfer huge files such as videos. Often, each device in the network transmits or receives unique information, for this reason a robust system of individual addressing is extremely helpful. Security and design flexibility are frequently indispensable.

In comparison with other technologies the Zigbee technology is the best solution. Bluetooth commonly used for small, low power and short-range mobile personal area networks such as printers, fax, modems, and laptops. On the other hand Zigbee provides wide range transmission with very low power consumption. Bluetooth allows up to 8 slave

nodes in a basic master-slave piconet set-up while the zigbee allows up to 254 nodes. On the following table are summarized the characteristics of Bluetooth, Zigbee and WiFi technologies.

Technology Protocol	**Bluetooth IEEE 802.15.1**	**ZigBee IEEE 802.15.4**	**WiFi IEEE 802.11**
Network Definition	WPAN	Low- rate WPAN	WLAN
Topology	Ad hoc (with dynamically allocated central element) very small networks	Ad hoc, peer-to-peer ,star, mesh	Infrastructure (ad hoc also possible)
Data Rates	1Mbps (v. 1.2) 3 Mbps (v. 2.0)	250Kbps	11 Mbps (802.11b) 54 Mbps (802.11g)
Frequency	2,4GHz, 5 to 7 GHz	868, 915 MHz, 2,4 GHz	2,4 and 5,8 GHz
Range (Max)	5 to 10 m	100m	100m
Battery life (days)	1-7	100-1000+	hours
Application	Inter-devise wireless connectivity, e.g. handsets, audio, peripherals, video, phones, serial cable replacements	Wireless sensor networks, mesh networks, monitor and control, building automation	Wireless LAN, broadband internet.
Power consumption	medium	Very low	High
Security	64 and 128 bit encryption	128 AES	SSID

Table 2.1: Characteristics of Bluetooth, Zigbee and WiFi

2.2 Zigbee

The ZigBee Alliance is an association of companies working together to enable reliable, cost-effective, low-power, wirelessly networked, monitoring and control products based on an open global standard.

ZigBee is a standard communications protocol published by ZigBee Alliance for low-power, remote monitoring, control and sensor network applications. The standard was created to address the need for a cost-effective, standards-based wireless networking solution that supports low data-rates, low-power consumption, security, and reliability.
Among its features are:

- Self-healing mesh networking. ZigBee is a decentralized network topology very similar to the Internet. It allows nodes to find new routes throughout the network if one route fails.

- Devises that fully support zigbee technology can communicate with each other regardless of the different manufacturer.
- Low power consumption.
- Easy and inexpensive to deploy.
- Layered approach to communications design and security.

2.2.1 Application profiles

The ZigBee Alliance provides a number of solutions in order to help the consumers control their world by take the advantages of zigbee. This means that provides greener solutions because of its low-power wireless standards that can be easily installed and run on harvested energy or batteries for years. The application profiles are:

- Zigbee building, home, industrial automation.
- Zigbee remote control
- Zigbee health care
- Smarter shopping
- Lighting control
- Sensor networks

2.2.2 Zigbee protocol Stack

Zigbee alliance defines 2 layers the Network layer (NWK) and Application layer (APL) that are built on top of the IEEE 802.15.4 PHY and MAC layers of OSI. Each layer performs specific services. The network layer (NWK) is in charge of organizing and providing routing over a multihop network while the Application Layer (APL) intends to provide a framework for distributed application development and communication.

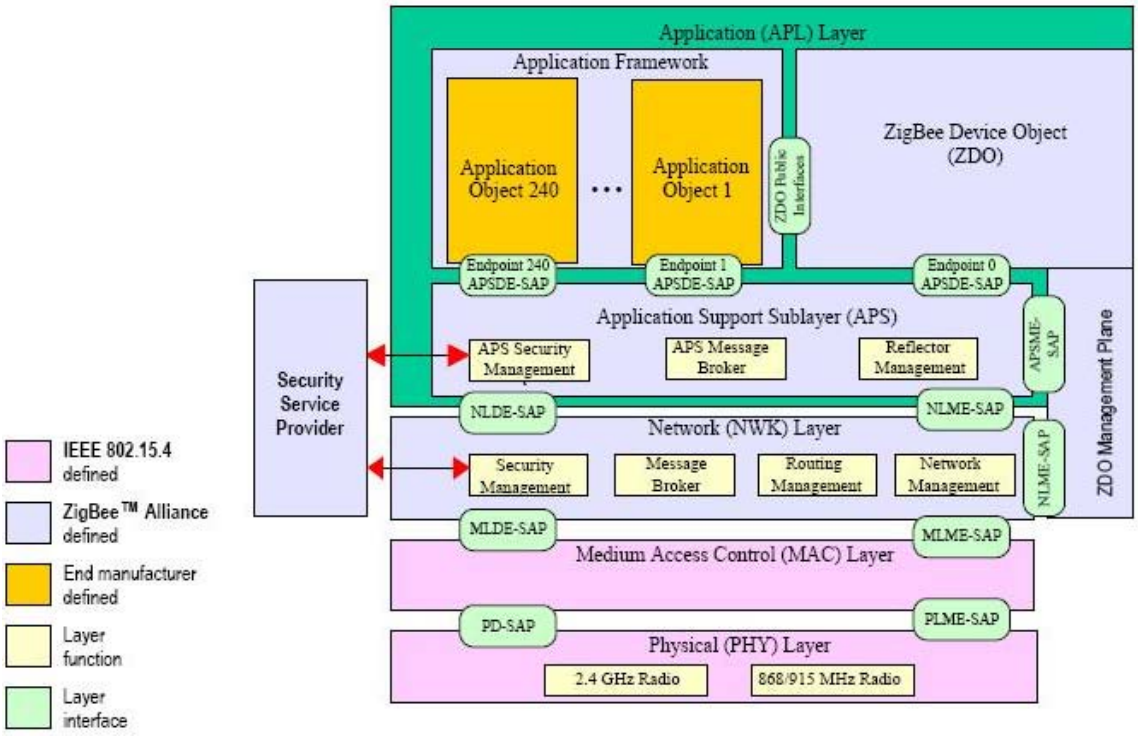

Figure 2.2: Zigbee layers architecture and protocols stack

2.2.2.1 Application (APL) Layer

As shown on the figure 2.2, the ZigBee application layer is the highest layer defined by the specification and consists of the APS sub-layer, the ZDO (containing the ZDO management plane), and the manufacturer-defined application objects.

The Zigbee application framework provides the communication between zigbee devices. Zigbee is using application profiles in order to satisfy specific applications for the end user. ZigBee application profiles are collections of common definitions and protocols that allow various devices to work together in a particular domain, such as home automation.

The application support sub-layer (APS) provides an interface between the network layer and application layer through a set of services (APSDE-SAP, APSME-SAP). It is responsible for maintaining tables for binding, which is the ability to match two devices together based on their services and their needs, and forwarding messages between bound devices. An example of bound devices could be the light and the switch connected with zigbee devices.

The zigbee device object (ZDO) is responsible for the defining of the role of a device within the network (coordinator, router or end device), provide services that allow the

application objects discovering devices on the network and determining which application services they provide, initiating and/or responding to binding requests and establishing a secure relationship between network devices.

2.2.2.2 Network (NWK) Layer

The network layer is responsible to ensure the correct operation of the lower layer (MAC) and to provide a suitable service interface to the application layer. The connection with the upper layer performed with two services the data service and management service that provide the necessary functionality. The network layer, as shown on the figure 2.2, handles the routing, the addressing, assign network addresses, adding and removing devices from the network, handles the security etc.

2.2.3 Zigbee Network device types

Zigbee network can identify three device types. Every Zigbee network can have a single coordinator device and at least 1 router device or end device, to form a zigbee network. Routers and coordinator devices can support up to 8 end device children each. More specifically:

- Coordinator: Is the device that starts and controls the network. The coordinator stores information about the network, handing out addresses and secure it.
- Router: Is the device that extends network area coverage, dynamically route around obstacles, and provide backup routes in case of network congestion or device failure. They can connect to the coordinator and other routers, and also support child devices.
- End device: An end device can send and receive information but it cannot perform any routing operations. End devices always need a router or the coordinator to be their parent device.

2.2.4 Network topology

The Zigbee supports three types of topologies:

- Star: the network is controlled by one single device called the ZigBee coordinator. The coordinator is responsible for the maintenance of the devices on its network. All the other devices must communicate with the coordinator directly.
- Tree: routers move data and control messages through the network using a hierarchical routing strategy.
- Mesh: the routing of messages is performed as a decentralized, cooperative process involving many peer devices routing on each others' behalf. In other words the mesh network allows the full peer-to-peer communication.

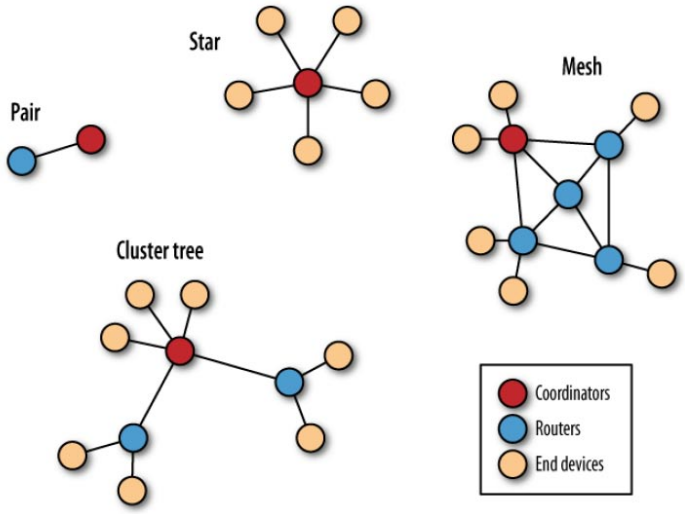

Figure 2.3: Zigbee network topologies

In addition, in tree and mesh topologies the network could be extended with the use of routers as shown on the figure 2.3.

There is also one type of network topology, the pair, which is the simplest network topology and consists of 2 nodes. One must be the coordinator and the second could be a router or end device.

2.2.5 Zigbee Addressing

The 802.15.4 protocol upon which the ZigBee protocol is built specifies two address types:

- 16-bit Network addresses which are assigned to the nodes when the nodes join a network. The network address is unique for each node to the network but not static.
- 64-bit addresses, which are unique and are identifiers for the nodes.

2.2.6 Zigbee Security

Security services provided for ZigBee include methods for key establishment, key transport, frame protection, and device management. These services form the building blocks for implementing security policies within a ZigBee device (Zigbee spec. 2007).

2.2.6.1 Security keys

Security amongst a network of ZigBee devices is based on link keys and a network key. Unicast communication between APL peer entities is secured by means of a 128-bit link key shared between two devices, while broadcast communications are secured by means of a 128-bit network key shared amongst all devices in the network (Zigbee spec. 2007).

2.2.6.2 Trust center

To function securely in a network, must have to be a device trusted by devices within a ZigBee network to distribute keys and which controls access. ZigBee introduces the concept of the Trust Center, which

- Stores the keys for the network
- Uses the security services to configure a device with its key(s)
- Uses the security services to authorize a device onto the network
- The ZigBee Coordinator is usually designated the trust center.

2.2.7 Zigbee Channels and Frequencies

The zigbee has 27 channels available that are divided on three frequency bands. One channel in the 868 MHz band for Europe countries, 10 channels for N. America, Australia and a few additional countries in the 915 MHz band, and the remaining 16 channels for global use in the 2,4GHz band.

2.3 XBee Radio

XBee module is the most popular among other zigbee modules that fully support zigbee technology, manufactured by Digi. It provides a reliable and secure communication and

has a number of features that makes it easy for beginners. There are two basic types of XBee radio physical hardware; XBee series 1 and XBee series 2. Their differences are displayed on table 2.2.

	Series 1	Series 2
Typical (indoor/urban) range	30 meters	40 meters
Best (line of sight) range	100 meters	120 meters
Transmit/Receive current	45/50 mA	40/40 mA
Firmware (typical)	802.15.4 point-to-point	ZB ZigBee mesh
Digital input/output pins	8 (plus 1 input-only)	11
Analog input pins	7	4
Analog (PWM) output pins	2	None
Low power, low bandwidth, low cost, addressable, standardized, small, popular	Yes	Yes
Interoperable mesh routing, ad hoc network creation, self-healing networks	No	Yes
Point-to-point, star topologies	Yes	Yes
Mesh, cluster tree topologies	No	Yes
Single firmware for all modes	Yes	No
Requires coordinator node	No	Yes
Point-to-point configuration	Simple	More involved
Standards-based networking	Yes	Yes
Standards-based applications	No	Yes
Underlying chipset	Freescale	Ember
Firmware available	802.15.4 (IEEE standard), DigiMesh (proprietary)	ZB (ZigBee 2007), ZNet 2.5 (obsolete)
Up-to-date and actively supported	Yes	Yes

Table 2.2: Differences between XBee series 1 and series 2

Both series 1 and series 2 are available on two transmission powers, the regular version that is simply called XBee and the PRO called XBee-PRO. The latter has more power and is more expensive.

There is also one classification of XBees according to their antenna. The available types of antennas are:

- Whip or wire antenna. It's simple and offers omnidirectional radiation, meaning the maximum transmission distance is pretty much the same in all directions when its wire is straight and perpendicular to the module (Faludi, 2011).
- Chip antenna, is a chip antenna on the XBee board. It is smaller and sturdier, than others but the signal is attenuated in many directions.

- PCB antenna. It is printed directly on the circuit board of the XBee. It is composed of a series of conducting traces laid out in a fractal pattern. The PCB antenna offers many of the same advantages (and disadvantages) as the chip antenna with a much lower cost to manufacturer (Faludi, 2011).

- U.FL connector, it is used in the case where the XBee board is enclosed on a box. In this case the signal is not attenuated by the enclosure and so the antenna must with some way placed out of the box. The U.FL connector is small, somewhat fragile, and almost always used with a short connecting cable that carries the signal from a remotely mounted antenna (Faludi, 2011).

- RPSMA connector is just a different type of socket from the U.FL connector. It's bigger and bulkier, but you can use it with an external antenna, mounted directly to the XBee without a connecting cable. For most introductory projects, you're still best off with the simple wire antenna that is smaller, cheaper, and usually just as good.

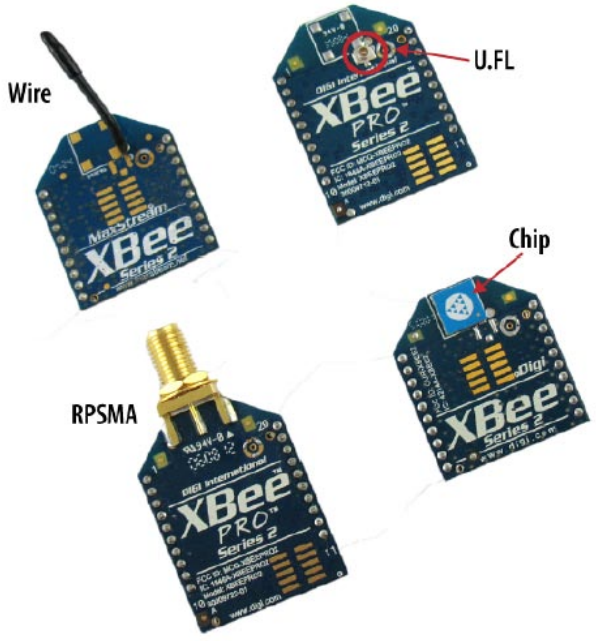

Figure 2.4: XBee antenna types

The model that is selected on this thesis is the XB24-Z7WIT-004 which is an XBee series 2 radio with wire antenna and it is selected as it is the most proposed XBee module and cheaper than the others.

Among its positives are that it allows a very reliable and simple communication with low-cost and minimal power between microcontrollers, computers, arduino boards and anything has a serial port. The specifications of XBee series 2 are:

- Indoor/Urban: up to 40m
- Outdoor line-of-sight: 120m
- Transmit power: 2mW
- RF Data rate: 250.000bps
- Supply Voltage: 2.8 – 3.4 V
- Operating Current (Transmit): 40mA (@ 3.3 V)
- Operating Current (Receive): 40mA (@ 3.3 V)
- Supported topologies: Point-to-point, Point-to-multipoint, Peer-to-peer & Mesh
- Operating Frequency Band: ISM 2.4 GHz
- Addressing Options: PAN ID and Addresses, Cluster IDs and Endpoints (optional)
- AT or API command set
- Built-in antenna
- 128-bit encryption
- 8 digital IO pins
- 6 10-bit ADC input pins

2.3.1 X-CTU Configuration & Test Utility Software

For the configuration of the XBees there are several terminal programs depending from the operating system that the user works to. As for example there is the Coolterm for Windows and Macintosh operating systems, HyperTerminal for Windows XP and older Windows versions etc. The X-CTU program is the official configuration program for XBee radios provided by Digi International. Firstly it was available only for Widows but with the latest version Digi creates a next generation configuration platform for XBees which is also compatible for Mac OS and it is free. The X-CTU will be used for the configuration of the XBee radios.

With X-CTU it is easy to update the firmware and define the parameters of XBees. The first step before the configuration of the XBee is the connection with the computer. This could be achieved with the use of an XBee USB explorer (figure 2.5).

Figure 2.5: XBee explore

The explorer is an adapter that uses standard USB A to mini-B cable to connect with the computer. After the update of the firmware with the latest version, starts the configuration of the XBee.

2.4 Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It is intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments. Arduino designed to sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. It has many applications such as musical applications, to robotics, to botanology etc.

Arduino has many positives against other platforms (Banzi M. 2008), because of these features:

- It is a multiplatform environment; it can run on Windows, Macintosh and Linux.
- It is based on the Processing programming IDE, an easy-to-use development environment used by artists and designers.
- You program it via a USB cable, not a serial port. This feature is useful, because many modern computers don’t have serial ports.
- It is open source hardware and software -if you wish, you can download the circuit diagram, buy all the components, and make your own, without paying anything to the makers of Arduino.
- The hardware is cheap. The USB board costs about €20 (currently, about US$35) and replacing a burnt-out chip on the board is easy and costs no more than €5 or US$4. So you can afford to make mistakes.

- There is an active community of users, so there are plenty of people who can help you.
- The Arduino Project was developed in an educational environment and is therefore great for newcomers to get things working quickly.

The Arduino language is implemented in C/C++ and therefore has some differences from the Processing language, which is based on Java.

2.4.1 Arduino Boards

There are many types of arduino boards depending from the use. For example LilyPad Arduino is designed for stitching into clothing and other fabric/flexible applications and on the other side Arduino Mega is a larger and more powerful Arduino board. The following two types of arduino boards, Arduino Yún and Arduino UNO were selected to the proposed prototype. Yún selected because of its Wi-Fi enabled, it has a Standard-A type USB connector to which can be connected with USB devices and it has a micro-SD card plug, for additional storage. The Arduino Uno and Leonardo have pretty much the same characteristics. Finally it is chosen the Arduino Uno because it is basic on its series, it is more tested and has no bug issues.

2.4.1.1 Arduino Yún

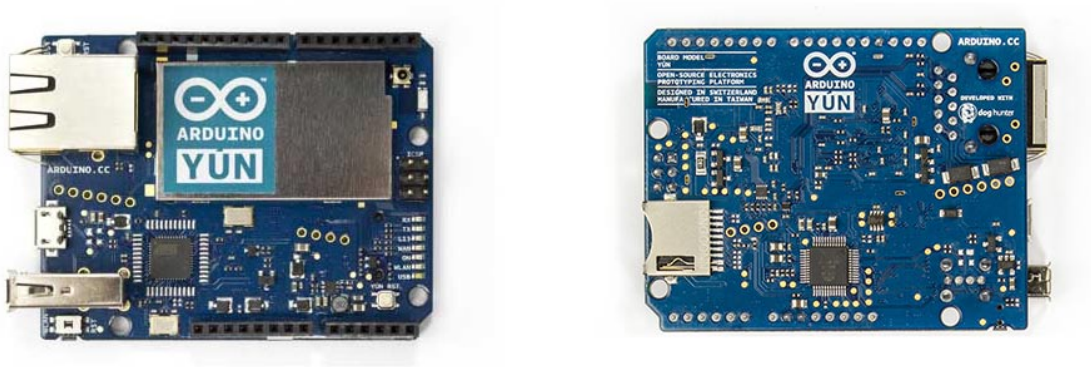

Figure 2.6: Arduino Yún front and rear

The Arduino Yún is a microcontroller board based on the ATmega32u4 which provides high performance and low power, and the Atheros AR9331. The Atheros processor

supports a Linux distribution based on OpenWrt named OpenWrt-Yún. As it is shown on figure 2.6 the board has:

- built-in Ethernet support
- built-in WiFi support. The Yún has the ability to act as an Access Point, but it can also connect to an existing network.
- a USB-A port
- micro-SD card slot, for external memory.
- 20 digital input/output pins, 7 of which can be used as PWM outputs and 12 as analog inputs
- 16 MHz crystal oscillator
- micro USB connection
- ICSP header
- 3 reset buttons with most useful the WLAN RST button that has double feature. Firstly restores the Wifi to the factory configuration by assign it the default IP address that is 192.168.240.1. In this condition it could be connected with the computer to its WiFi network that appear with the SSID name "Arduino Yún-XXXXXXXXXXXX", where the twelve 'X' are the MAC address of the Yún. When the connection established then by typing on the browser the 192.168.240.1 or "http://arduino.local" address can access to the configuration panel of Yún. The second function of the WLAN RST button is the restore the Linux image to the default factory image. Note that by restoring the factory, all the files saved and software installed on the on-board flash memory connected to the AR9331 will be lost.

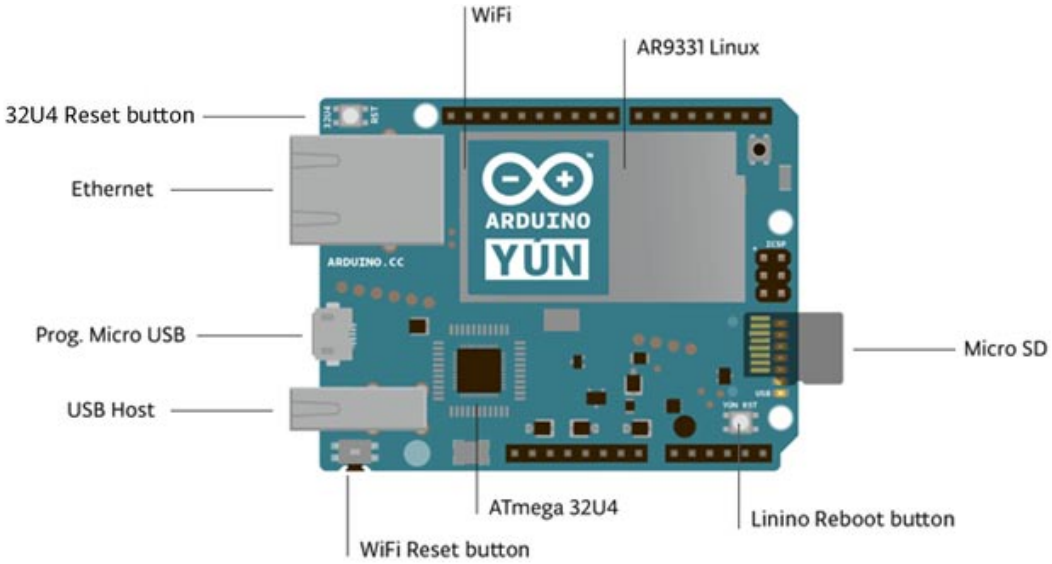

Figure 2.7: Arduino Yún

Arduino Yún is the first member of the series of wifi products combining Arduino with Linux. It distinguishes itself from other Arduino boards in that it can communicate with the Linux distribution onboard, offering a powerful networked computer with the ease of Arduino.

As already said, the Arduino Yún has two processors on board. One is an ATmega32U4 and the other is an Atheros 9331 which running Linux and the OpenWrt wireless stack. The last one has onboard Ethernet and WiFi interfaces. It is possible to call programs in Linux part and through the Arduino to connect with various internet services. That is the job of Bridge library which handles the communication between the Linux processor and the Arduino on the Yún.

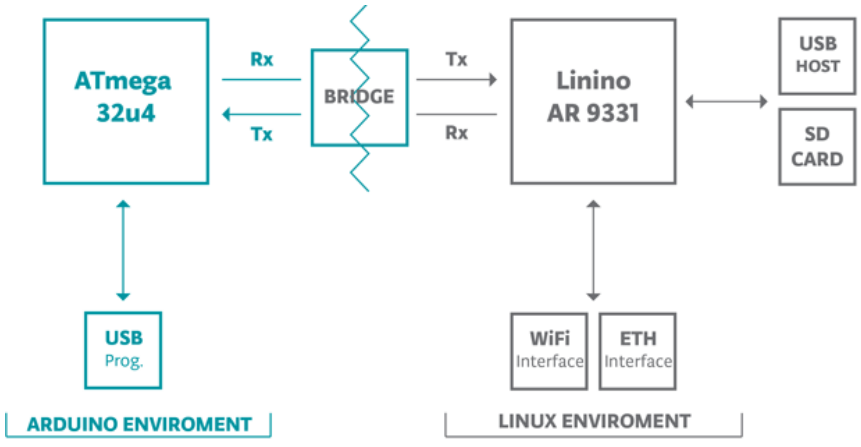

Figure 2.8: Bridge simplifies the communication of ATmega32u and AR9331.

Its power supply is recommended to be 5Volts DC. The Yún has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega32U4 provides a dedicated UART TTL (5V) serial communication. The 32U4 also allows for serial communication over USB and appears as a virtual com port to software on the computer. The chip also acts as a full speed USB 2.0 device, using standard USB COM drivers. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

A SoftwareSerial library allows for serial communication on any of the Yún's digital pins except for pins 0 RX and 1 TX. The ATmega32U4 on the Arduino Yún comes preburned with a bootloader that allows the uploading of new code to it without the use of an external hardware programmer.

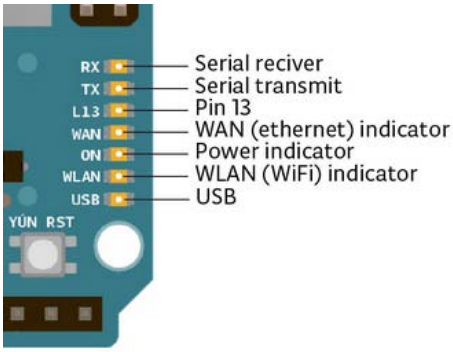

Figure 2.9: Arduino Yún's several status LEDs

There are also status LEDs on it, indicating power, WLAN connection, WAN connection and USB.

As already mentioned, arduino Yún runs a distribution of Linux called OpenWrt-Yún that allows the configuration not only from command line but also from a web page. OpenWrt-Yún uses REST for clients and servers. It is a software architecture that exposes various parts of the Arduino hardware through URLs.

By default, the REST API access is password protected but it is possible to be changed and services are accessed without a password.

The there are three REST end points recognized by the Yún:

- /arduino

- /data
- /mailbox

The "/arduino" directory doesn't have anything pre-configured. Anything added to the URL after the end point is passed from the web server to the sketch on the 32U4.

"/data" is used to access to the internal key/value storage. The available calls are:

- /put/KEY/VALUE: stores a value inside the storage
- /get/KEY: obtains the value of the requested key in JSON
- /get: obtains the entire storage list in JSON.
- /delete: deletes the internal storage.

"mailbox" is used to add messages into a queue called Mailbox. The message queue is stored inside the Linux system and can be read by the Arduino microcontroller. The REST call is "/mailbox/message".

2.4.1.2 Arduino UNO Rev3

The Arduino Uno is a microcontroller board based on the ATmega328 which is a high performance and low power AVR 8-Bit microcontroller. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button.

It is similar to the Duemilanove, but has a different USB-to-serial chip the ATMega8U2, and newly designed labeling to make inputs and outputs easier to identify.

Arduino UNO contains everything needed to support the microcontroller. It is simple to use it; simply by connecting it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

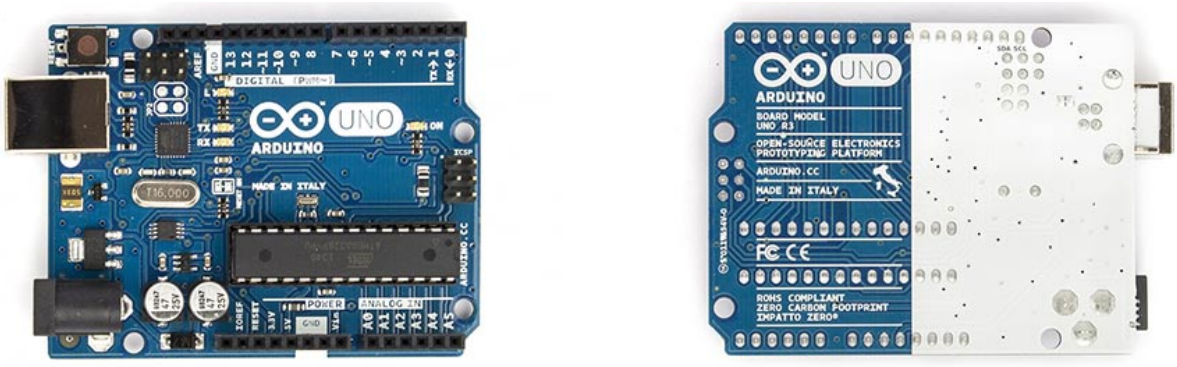

Figure 2.10: Arduino R3 front and back

It can operate with external supply of 6 to 20 V but the recommended range of power supply is among 7 to 12 V. If supplied with less than 7V then the 5V pin may supply less than 5V and the UNO may be unstable. In the other case if the UNO supplied with more than 12 volt, the voltage regulator may overheat and damage the board.

On this project it was used an Arduino UNO R3, that is the revision 3 of UNO, which has some new features such as stronger RESET circuit and the replacement of 8U2 with the Atmega 16U2.

As already said the UNO has 14 digital pins that can be used as an input or output. Additionally there are some pins that have special functions:

- Serial: 0 (RX) and 1 (TX) pin are used to receive (RX) and transmit (TX) TTL serial data.
- Pins 2 and 3 are external interrupts. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication
- AREF. Reference voltage for the analog inputs.
- Reset. Bring this line LOW to reset the microcontroller.

The arduino UNO supports communication with computers, other arduino and other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). The ATmega16U2 microcontroller channels the serial communication over USB and appears as a virtual com post to software on the computer. When the communication established then the RX and TX led on the board start flash every time the data is being transmitted via the USB-to-serial chip and USB connection to the computer.

The programming of UNO could be accomplished via the software provided by Arduino. The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer.

2.4.2 Arduino Software

Arduino provides software to the users to write code and upload it to the arduino platform. There are available versions for Windows, Mac and Linux and is free. The code written using arduino is called sketch which is written in a simple arduino language. Then it is translated into C language and is passed to the avr-gcc compiler to make it understood by the microcontroller. The arduino IDE is based on Java and includes:

- an editor to write the sketches
- a compiler
- a serial monitor which allows simple textual data to be sent to and from the Arduino board.
- libraries
- available examples

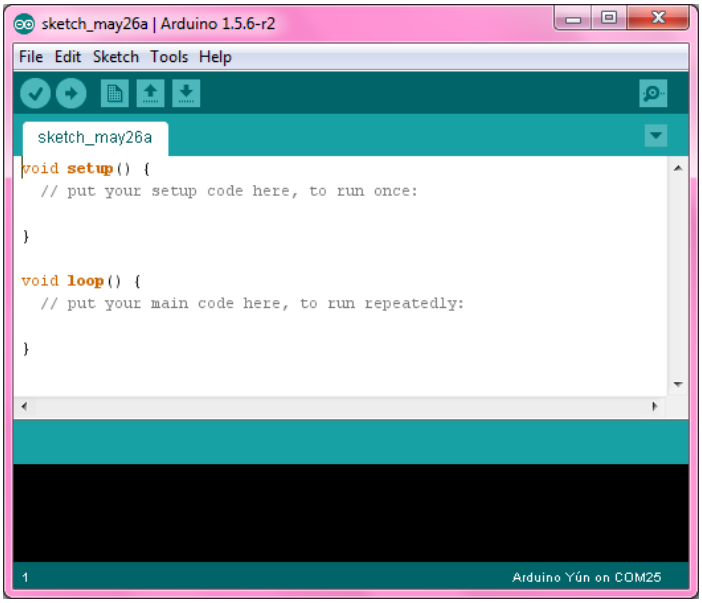

Figure 2.11: Arduino IDE

2.4 Conclusion

For the implementation of the proposed system it will take into consideration everything mentioned on this chapter as a background. On the following chapter it will be analyzed the prototype that was created to and all the above issues will be mentioned in more details.

Chapter 3

The Proposed System

3.1 The prototype

A prototype was constructed as it is displayed on figure 3.1. It could be divided in two main parts the Home side which is the wireless sensor network and the Server side which is responsible for maintaining the received data, control and monitoring. The home system is a circuit of sensors, XBees and Arduino boards. Arduino and XBee is a very popular combination for electronics projects. In more detail the home network consists of two sub-systems the transmitter and the receiver.

The transmitter sub-system consists of sensors that collect the data and send them to the arduino UNO. The UNO processing the incoming data and with serial communication forwards the data to the XBee router. The XBee router, with zigbee wireless technology sends the data to the receiver.

On the receiver's side, the data collected by the second XBee which is the coordinator and through serial communication forwards them to the second arduino which is a Yún. Finally the Yún has built-in wifi and forwards the data to the server thought the internet connection. Additionally on the receiver there is a camera connected on a USB port of arduino for the surveillance of the house an also a led replacing a real lamp of the room which could be opened or closed or switched on/off for the monitoring of the house.

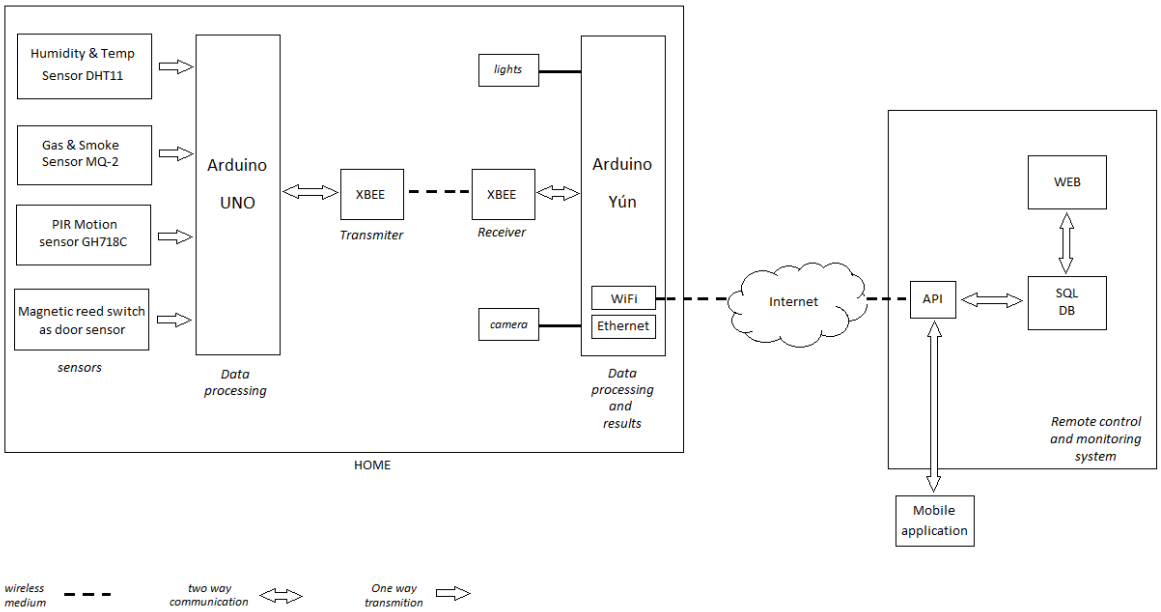

Figure 3.1: The prototype architecture

On the server side, the heart of the system is the API which is responsible for user authentication, display the sensors data, send notifications if an alert occurs, make a check if the communication with sensor network exists or not.

All the prototype components will be mentioned in the sections that follow, beginning with the sensors.

3.1.1 The sensors

3.1.1.1 Temperature and humidity sensor

For the measurement of temperature and humidity a DHT11 sensor was selected. The DHT11 is good for 20-80% humidity readings with 5% accuracy and for 0-50°C temperature readings with ±2°C accuracy. Its power supply is 3 to 5V. Among its features are the low, relative humidity and temperature measurement, the excellent quality, fast responsiveness and precise calibration.

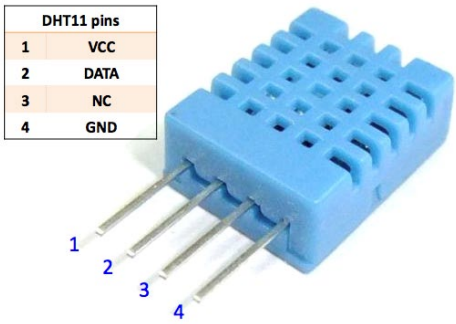

Figure 3.2: Temperature and humidity sensor DHT11.

In DHT11, a complete data transmition is 40bit and a communication process is about 4ms. The sensor sends higher data first. The data format is the 8bit humidity integer data + 8bit the Humidity decimal data +8 bit temperature integer data + 8bit fractional temperature data +8 bit parity bit (check sum).

The code that used to get the correct temperature and humidity measurements is the following:

```
byte dht11_dat[5];
  byte dht11_in;
  byte i;
  // start condition
  // 1. pull-down i/o pin from 18ms
  PORTC &= ~_BV(DHT11_PIN);
  delay(18);
  PORTC |= _BV(DHT11_PIN);
  delayMicroseconds(40);

  DDRC &= ~_BV(DHT11_PIN);
  delayMicroseconds(40);

  dht11_in = PINC & _BV(DHT11_PIN);

  if (dht11_in) {
    Serial.println("dht11 start condition 1 not met");
    return;
```

```
  }
  delayMicroseconds(80);


  dht11_in = PINC & _BV(DHT11_PIN);


  if (!dht11_in) {
    Serial.println("dht11 start condition 2 not met");
    return;
  }
  delayMicroseconds(80);
  // now ready for data reception
  for (i = 0; i < 5; i++)
    dht11_dat[i] = read_dht11_dat();
  DDRC |= _BV(DHT11_PIN);
  PORTC |= _BV(DHT11_PIN);
  byte dht11_check_sum = dht11_dat[0] + dht11_dat[1] + dht11_dat[2] +
dht11_dat[3];
  // check check_sum
  if (dht11_dat[4] != dht11_check_sum)
  {
    Serial.println("DHT11 checksum error");
  }
```

3.1.1.2 Gas and Smoke sensor

An MQ-2 Semiconductor Sensor for combustible gas was selected. This gas and smoke sensor detects the concentrations of combustible gas in the air and outputs its reading as an analog voltage. It has high sensitity to LPG, propane and hydrogen, also could be used to methane and other combustible steam, it is with low cost and suitable for different application.

Figure 3.3: MQ-2 semiconductor gas sensor.

The sensor can measure concentrations of flammable gas of 300 to 10,000 ppm. The sensor can operate at temperatures from -20 to 50°C and consumes less than 150 mA at 5 V. It was selected among other gas sensors because of its long life, low cost and sensitivity in flammable gases that may be produced on a house.

It was also used a breakout board to be connected to the breadboard and that because of the position of its pins which are not fit to these on the breadboard. The code to calculate the value for the smoke is the following:

```
float vol;
  int smokevalue = analogRead(A1);  //Analog 1 pin
  vol = (float)smokevalue / 1024 * 5.0;  //like in the datasheet
```

3.1.1.3 Door sensor

As a door sensor it was used a magnetic reed switch. Its function is to trigger an alert. It consists of two parts. The parts are close, making electrical contact. The switch is actuated (door opened) by pulled away the magnet. When bringing a magnet near the switch, the reed switch will go back to its original position.

Figure 3.4: Magnetic Reed Switch

This is the same type of switch used in many security systems to detect a window or door being open. The Reed Switch comes with a magnet. To take the value of magnet which will be high or low simply read the value from a specified digital pin:

```
magnetvalue = digitalRead(magnetpin);
```

3.1.1.4 Motion

To detect entry on the house, it was used a PIR motion sensor GH-718. It is a highly integrated module with maximum sentry distance 7m and sentry angle 110 degrees.

Figure 3.5: Tiny PIR motion Sensor Module GH-718

When the sensor detects motion then the value is high and triggers the alert. Its advantages are its simplicity and it is easy to adopt any circuit system. It was also tested a PIR Motion Sensor SE-10 but without the expected results. With the same way as door sensor to get the motion state simply read the value from the specified digital pin which called pirpin:

```
pirvalue = digitalRead(pirpin);
```

3.1.1.5 Camera

In order to capture images of an intruder, a USB 2.0 web camera from Sonix Technology Co. Ltd. was selected. The particular web camera has build-in some led lights to pure light if and when it is necessary. The important thing to notice is that this camera provides UVC support so it is easy to control it and capture images using unix shell prompt.

Figure 3.6: USB 2.0 Web Camera

The Linux Environment on the Arduino microcontroller has installed fswebcam which is program that captures images given a specific shell command. The command to capture an image, is given by the Arduino Environment, when it is necessary, using the Bridge library which is available with Arduino Yún.

3.1.2 XBee radios

There are 2 XBee radios on the project. The router XBee is connected with an XBee shield (figure 3.7) on the UNO that collects the sensors data and the coordinator XBee receive with zigbee communication the data and send them to the server through the Yún.

Figure 3.7: the XBee shield, to connect the XBee with UNO

The most important that has to be defined is the Baud rate (pulses per second). The XBee must have the same Baud rate with the computer to be achieved a successful communication. For the same reason the XBees must have the same Baud rates to communicate with each other and also the same baud rate with the arduino boards. The default settings are:

- Baud: 9600 baud
- Data: 8bit
- Parity: None
- Stop bits: 1
- Flow control: None
- Line feed: CR+LF or Auto Line Feed

The configuration starts with the coordinator. For the connection of the XBee with the computer was used an XBee explorer. To form a network should be set the following parameters with the use of X-CTU:

- ID (PAN ID):1045
- SC (Scan Channels): FFFF, to scan all the channels
- SD (Scan Duration): 3 (default), number of channels to scan
- NJ (Node Join Time) : FF , always allows join

Device addressing settings

- SH (Serial Number High): 13A200, it is set by the Digi and it is written on the back view of the XBee.
- SL (Serial Number Low): 40ABCBF3, it is set by the Digi and it is written on the back view of the Xbee.
- DH (Destination Address High): 0, which is the default value
- DL (Destination Address): FFFF

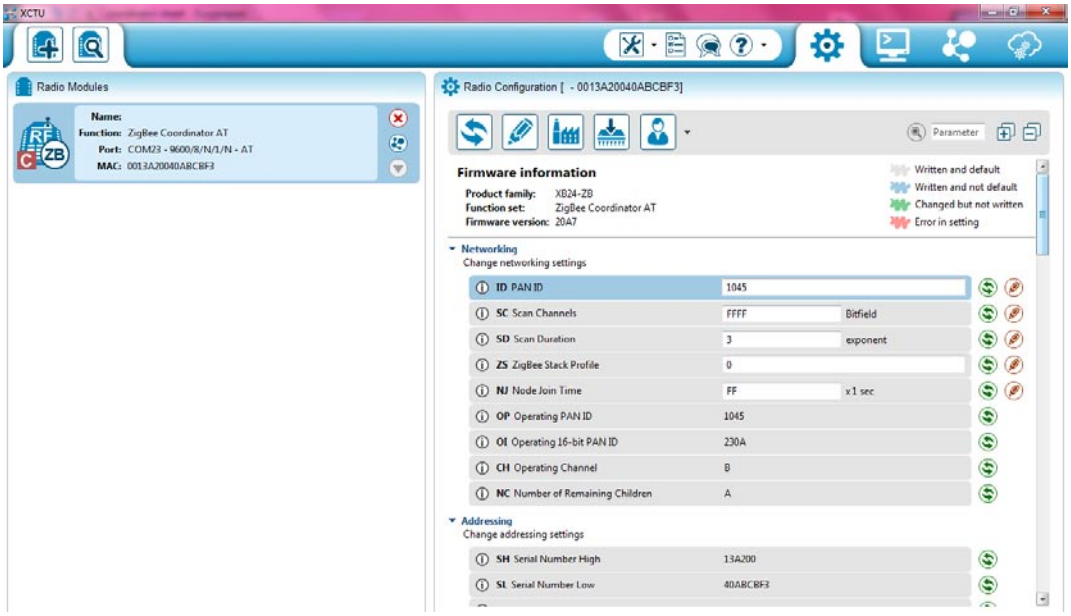

Figure 3.8: X-CTU XBee coordinator configuration

There are also optional security parameters for a secure communication. It was given a network encryption key which for security reasons does not displayed on the field as shown on the figure 3.9.

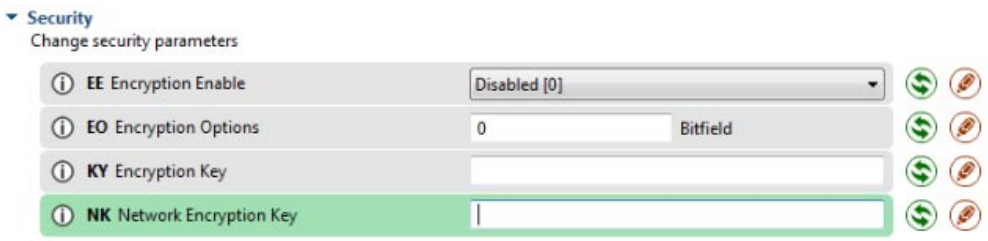

Figure 3.9: X-CTU XBee security configuration.

The router configuration settings are the same (the router has to be in the same PAN to communicate with the coordinator) except from the following value:

- DL: 40ABCA39 which is the low part of serial number of coordinator.

With the end of the configuration process the coordinator placed on the XBee shield on UNO and the router on the breadboard. Note that the XBee pins are spaced 2mm so the XBee cannot be placed directly on the breadboard. For this reason an XBee explorer regulated was used (figure 3.10).

Figure 3.10: XBee explorer regulated

3.1.3 Arduino boards

For this prototype are used two Arduino devices, an Arduino Yún and an Arduino UNO.

The first device to analyze is Arduino UNO. The SoftwareSerial library allows for serial communication on any of the UNO's digital pins except 0 and 1. When it first turns on, the device setups the serial connections with the sensors. Then, it reads the data from all the sensors and passes them through the XBee connected at the same breadboard with UNO. This is a repeating process and it stops only when something interrupts the sensors, the power supply or something unexpected.

On the other side of this system is Arduino Yún. When the Yún is turned on for the first time, it becomes an Access Point, creating a Wi-Fi network named "Arduino". In order to setup a user this device must open a browser and go to the webpanel. For the configuration must be defined a Wi-Fi network name, type and password. Then, by click to the "Configure" button, the board restarts and join the corresponding home network.

Using the Bridge library in the sketch, Yún can link the 32u4 to Linux, launching programs and scripts, passing them parameters (sensor readings for example) and reading their output, thus creating a strong integration between the creativity of the sketch and the power of Linux. The Yún supports Shell and Python scripts out-of-the-box but additional libraries have been installed in order to work properly. The installation of the additional software on OpenWrt- Yún, achieved by using the Linux package manager opkg. The additional

libraries that were used to capture images by controlling the web camera connected on Yún, upload them to the FTP server and send all sensor readings and are listed below:

- python-openssl
- kmod-video-uvc
- fswebcam

In addition to these libraries, Yún, uses two more python scripts. The first is written to upload an image to the appropriate folder for captured images on the server and the second one to upload all sensors readings.

When the Yún starts, it first setups the bridge between the Arduino Environment and the Linux Environment, a server which listens to incoming commands and all the connections with the second XBee and the led light. After that, it starts a YunClient to read the information receiving calls from the REST api with /arduino endpoint, Yún provides. This endpoint is used to allow third party applications or remote connections in general, control the led light (connected with Yún). The client detects the following calls a

- /arduino/led/on
    - this call, makes the led light turn on
- /arduino/led/off
    - this call, makes the led light turn off
- /arduino/led/switch
    - this call, makes the led light to switch state

Then, it reads all data that XBee on the same breadboard received, makes a check whether the PIR sensor detected a movement. If it did, it uses the bridge library first, to capture an image, second, to upload the image to the ftp server and third, to upload all the sensors readings to the server. Otherwise, Yún uploads the sensors readings to the server but without an image reference.

The ftp and sensor readings upload, achieved by using 3 python scripts. These scripts are located inside a memory stick in Arduino Yún and are listed below:

- uploadToFTP.py
- send_sensors_data.py
- send_lights_data.py

3.2 The web site

The web site was implemented on Microsoft Visual Studio (VS) 2010 using .NET Framework 4, jquery, CSS3, HTML5 and SQL Server. ASP.NET was selected as the language to develop the website because of the power .NET Framework provides, the ease of use to develop a website, the better interaction with SQL Server and the fact that the website and the smartphone application are on the same ecosystem.

The site uploaded with the domain http://securemyhome.gr and runs on a server with the following characteristics:

- **Hardware**

Cpu : Intel Core i7 9xx 3.2 GHZ

Ram : 8 GB

Disk Storage : 400 GB

Connection Port : 100 Mbit/s

- **Web Scripting**

ASP 7.5.7601.17514

ASP.NET 2.0.50727.4927

ASP.NET 4.0.30319.17929

FastCGI Support 7.5.7601.17514

Python 2.6.5.12

Perl v5.10.1 built for MSWin32-x86-multi-thread

PHP 5.4.23

PHP 5.2.17

PHP 5.3.28

SSI 7.5.7601.17514

- **Operating system**

Windows Server 2008 R2 Standard SP1 (64 bit)

- **Web Server :** Microsoft IIS 7.5

3.2.1 Bootrstrap

The state of the art in web sites is the responsive sites which mean that sites can respond according to some criteria such as the screen resolution and the orientation. The site was created to be responsive. There are several CSS grid frameworks available that vary on complexity, maturity and compatibility. Some of the most known are Skeleton, Foundation and Bootstrap. The Bootstrap Twitter framework was selected because it is a complete, sleek, intuitive, and powerful front-end framework for faster and easier web development. It is compatible with the latest versions of all major browsers. Also it is open source and available on Github.

Bootstrap has a few easy ways to quickly get started. NuGet which is a package manager Microsoft development platform including .NET, was used for the Twitter Bootstrap installation (figure 3.11).

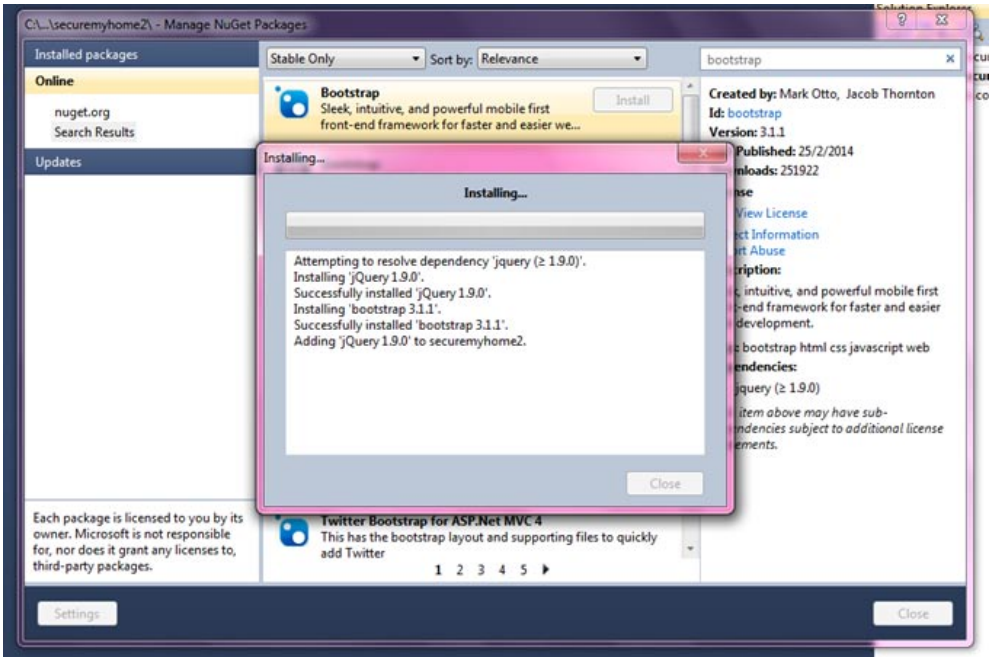

Figure 3.11: Twitter Bootstrap installation via NuGet

Bootstrap makes use of certain HTML elements and CSS properties that require the use of the HTML5 doctype. Bootstrap is mobile first. It includes a responsive, mobile first fluid grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes predefined classes for easy layout options, as well as powerful mixins for generating more semantic layouts.

The bootstrap grid is based on the following rules:

- There is a `.container` class which must include the rows. The container could be on fixed-width or fluid.
- The rows used to create horizontal groups of columns.
- The content should be placed within columns and only columns may be the immediate children of rows.
- Predefined grid classes like `.row` and `.col-md-6` are available for quickly making grid layouts.
- There are used media queries in less files of Bootstrap to create the key points in the grid system. On table 3.1 is shown the breakpoint sizes and how the grid responds on different screen widths.

	Extra small devices Phones (<768px)	**Small devices** Tablets (≥768px)	**Medium devices** Desktops (≥992px)	**Large devices** Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	`.col-xs-`	`.col-sm-`	`.col-md-`	`.col-lg-`
# of columns	12			
Column width	Auto	60px	78px	95px
Gutter width	30px (15px on each side of a column)			
Nestable	Yes			
Offsets	Yes			
Column ordering	Yes			

Table 3.1: Grid's behavior on different screen widths.

By taking the advantage of bootstrap's already existing classes, the web site on figure 3.12 was designed. On figures 3.13 and 3.14 is displayed the site on different widths. As the figures shown when the site displayed on screens less than 768px the top menu collapsed and replaced by a button. The code is the following:

```
<button type="button" class="navbar-toggle" data-toggle="collapse" data-
target=".navbar-collapse">
       <span class="sr-only">Toggle navigation</span>
       <span class="icon-bar"></span>
       <span class="icon-bar"></span>
       <span class="icon-bar"></span>
</button>
```

At this point Javascript is required. If JavaScript is disabled and the viewport is narrow enough that the navbar collapses, it will be impossible to expand the navbar and view the content within the `.navbar-collapse`. So at the end of the master page the following Javascript code must be placed:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
    <script language="javascript"
src='<%=ResolveUrl("~/bootstrap/js/bootstrap.min.js")%>'
        type='text/javascript'>
    </script>
```

To continue with, when the user press the button a drop down menu with all the menu items will be shown. It worth's mentioned that a `template.css` style sheet was created with all the css properties defined properly in order to succeed the desired result.

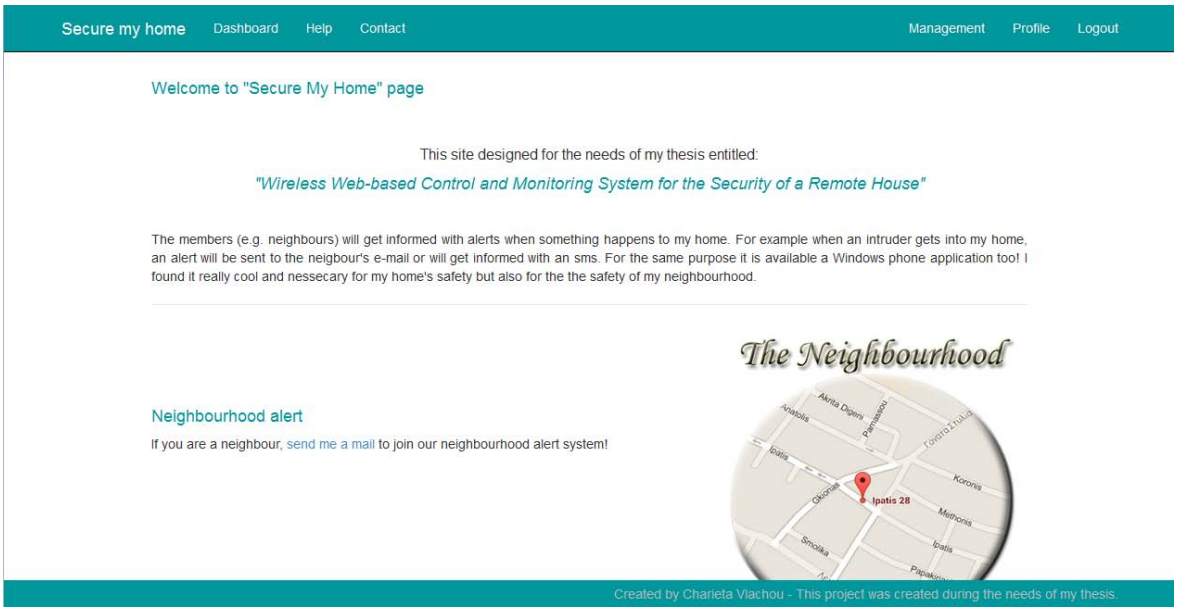

Figure 3.12: The web site default page on large devices (more than 1200px).

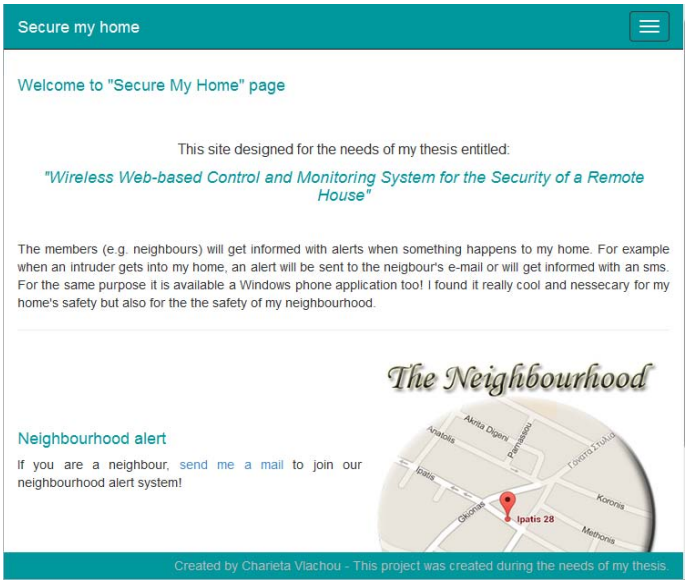

Figure 3.13: The default page on small devices (e.g.tablets)

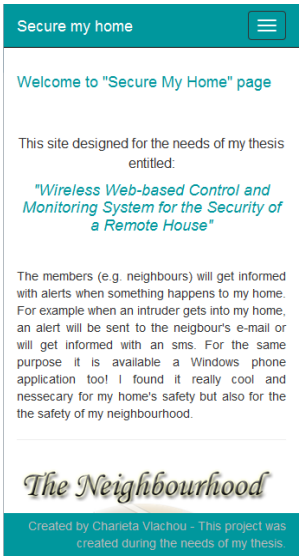

Figure 3.14: The default page on extra small devices (e.g. smartphones)

3.2.2 Web site description

The website offers the following options:

User authentication

This is a classic implementation which can be found on any website with user authentication login/logout functions and behavior minding user role (admin, neighbor etc.)

User profile

This option gives to the user the ability to edit his profile and add personal information, change the password and options about receiving notifications and how they reach him too.

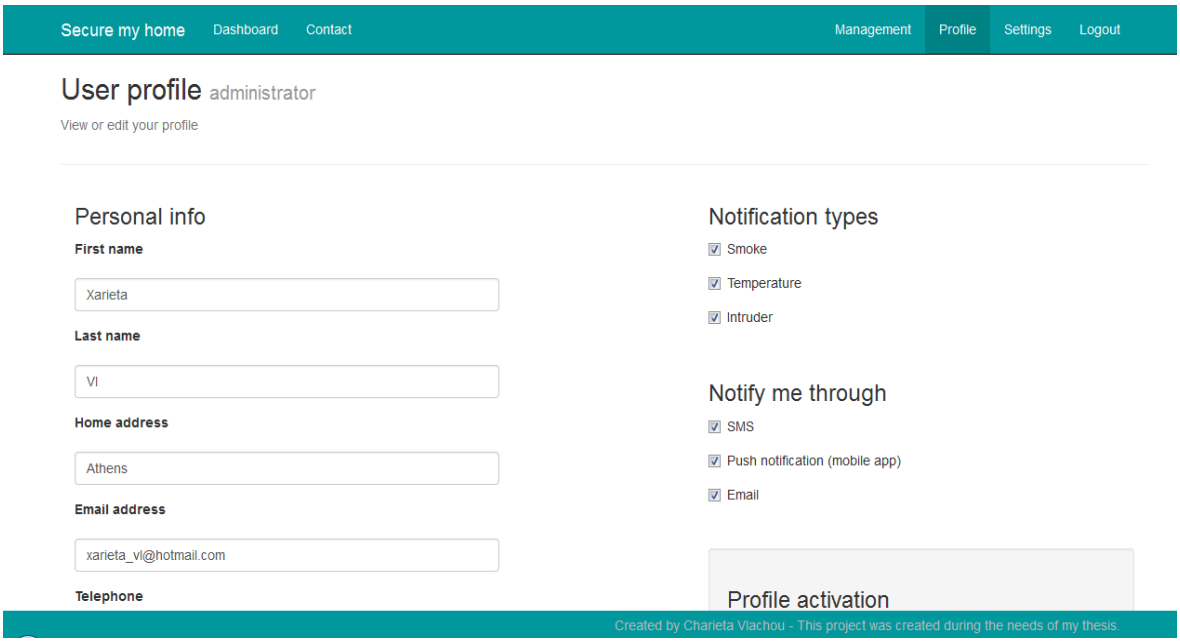

Figure 3.15 User Profile

User management

This option is available only for users with role admin and gives them the ability to edit another user's profile, change his role, delete a user, add a new one with a role of admin if the new user is a family member or a trustworthy person or add a new user with a role neighbor if the new user is a neighbor.

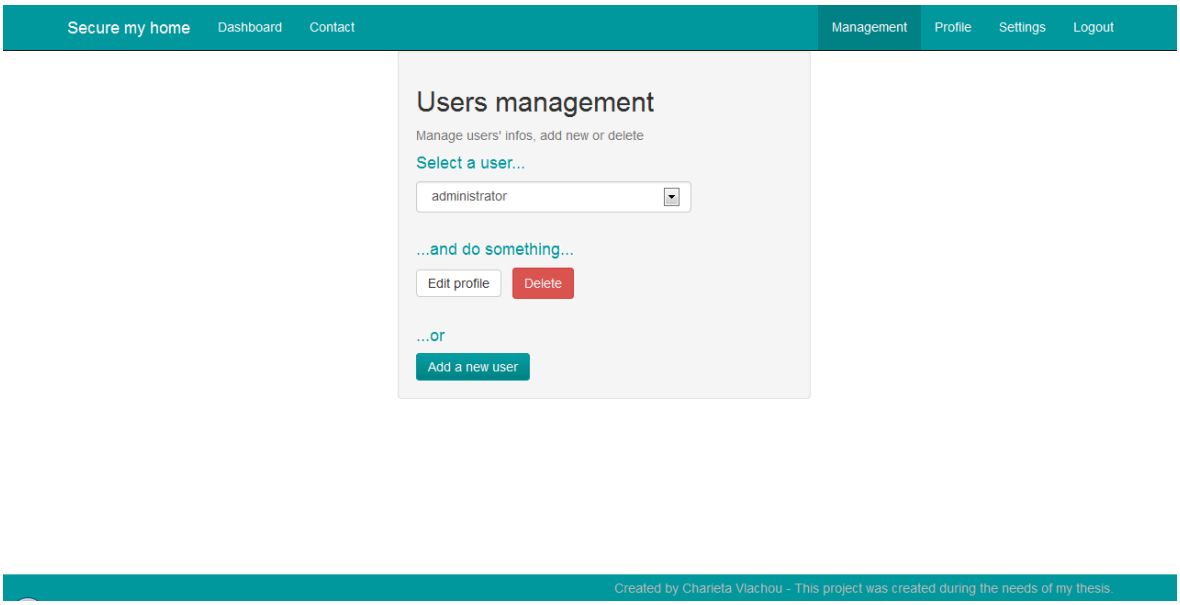

Figure 3.16 User Management

Website settings

This option is also available only for users with role admin. Any user having the role admin can activate the option to notify the authorities in case of intruder, increased temperature etc.

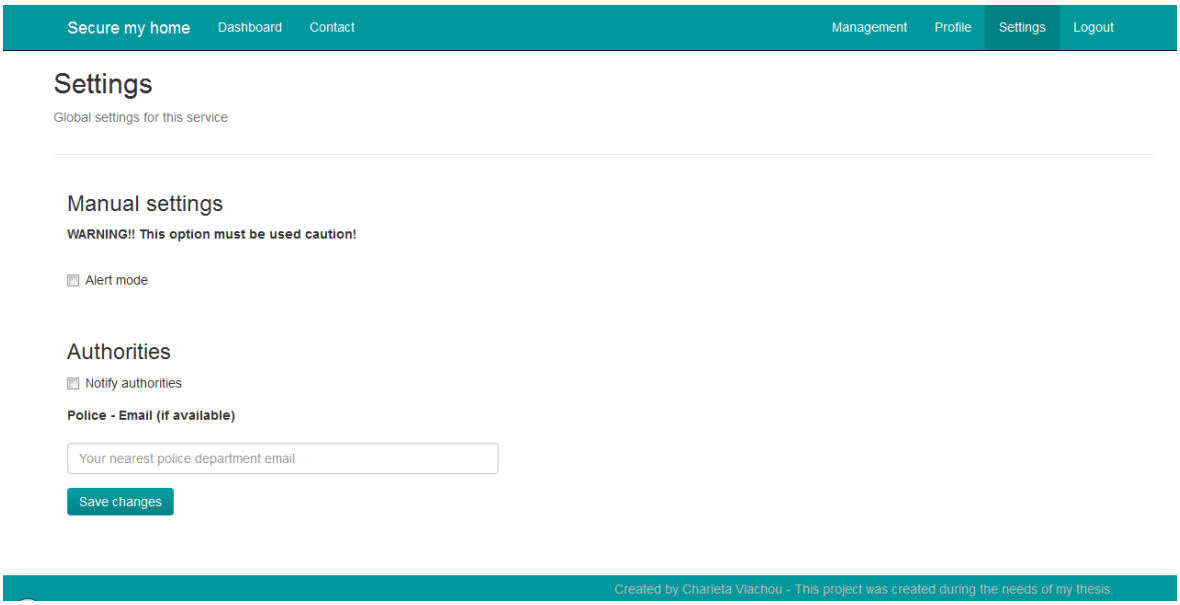

Figure 3.17: Website Settings

Dashboard

This option is also available only for users with role admin. This page shows all sensors readings with the most recent first, in a sortable table.

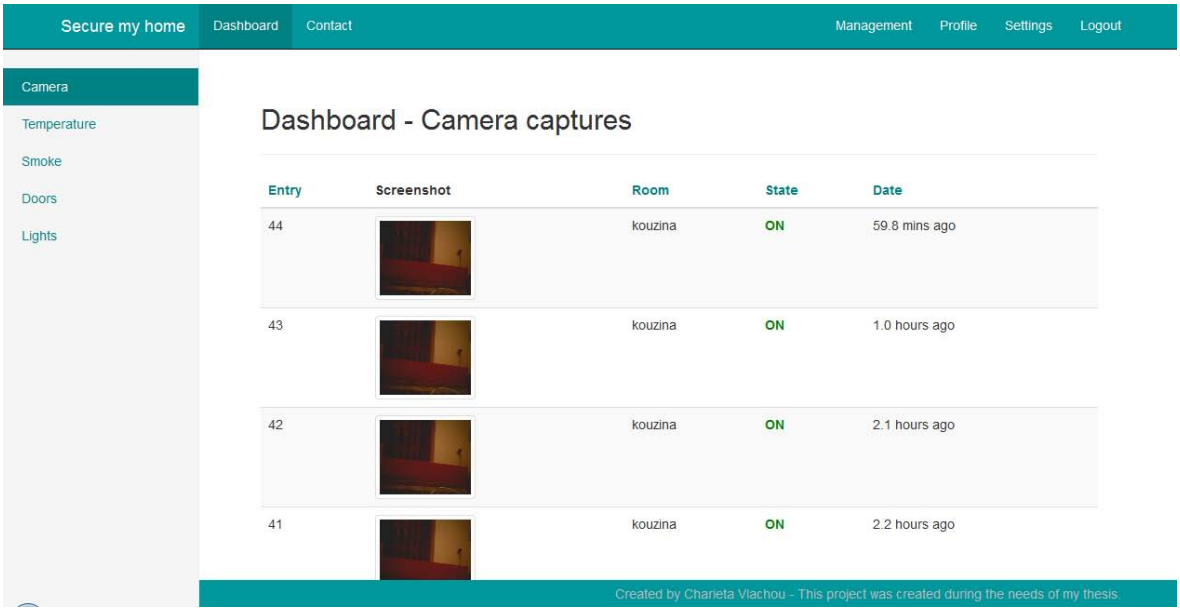

Figure 3.18: Dashboard – Camera captures

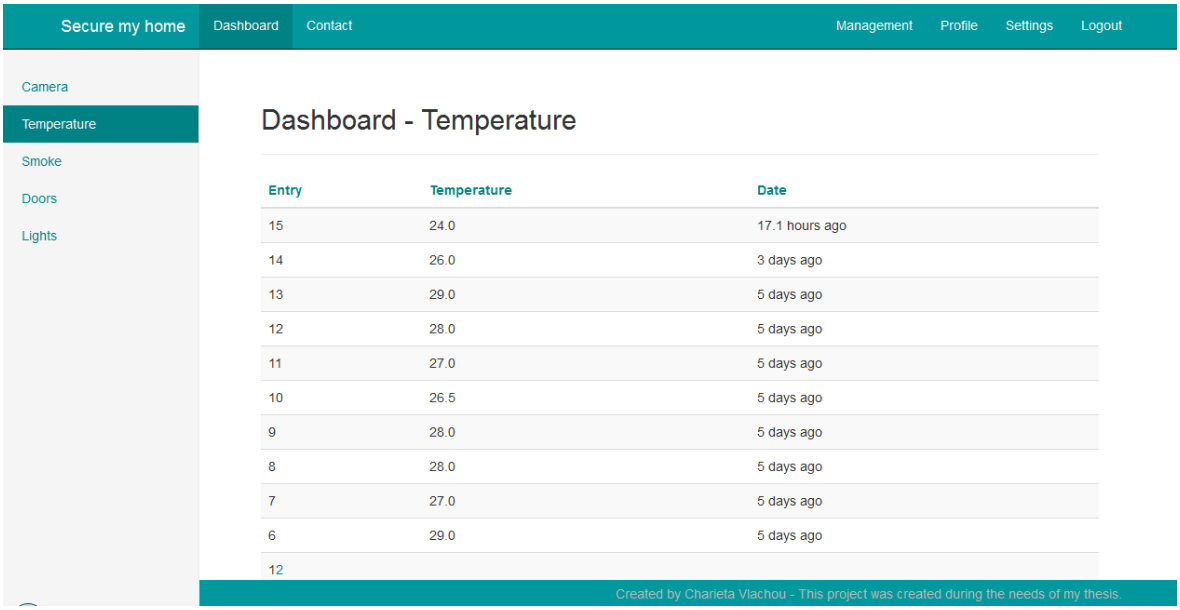

Figure 3.19: Dashboard – Temperature readings

Contact

Provides a contact form and contact information, for anyone to contact with the website administrator.

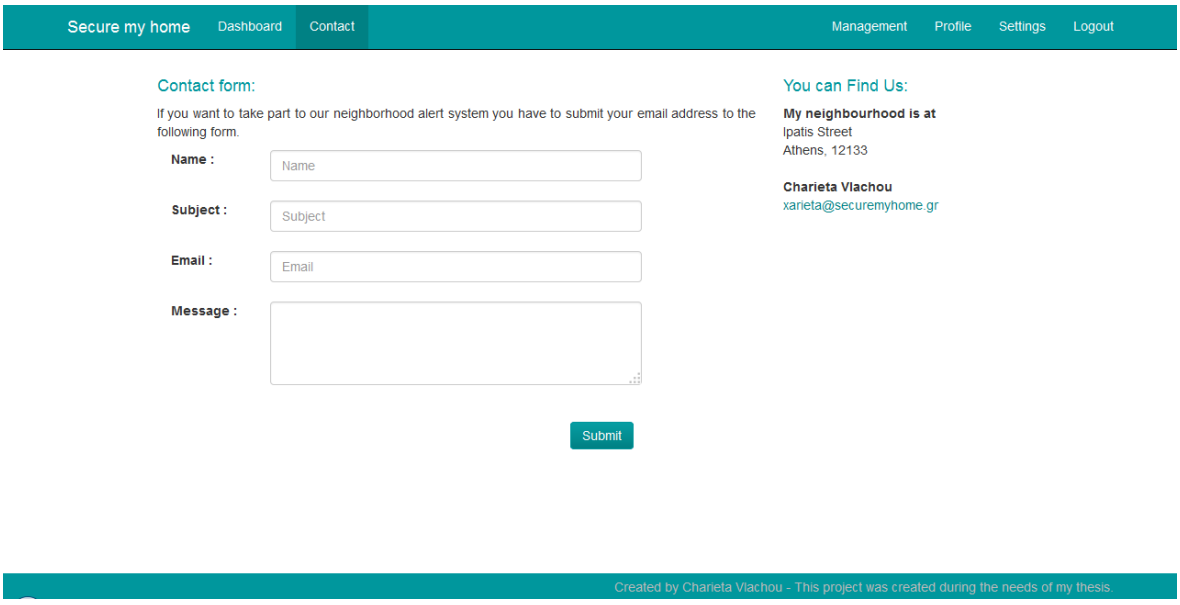

Figure 3.20: Contact form

In case of an alert, the website enters the alert mode. When enabled, a red bar is visible right below the navigation bar and users with role Admin can see a button to turn off the alert mode.

3.2.3 The Database

The database used for this prototype is the default database for using the ASP.NET Application Services, which includes profiles, roles, membership and more. Although this database is powerful enough to handle a web application's users' sessions and other use cases regarding users and their membership, it is incomplete when it comes about handle the data that sensors are receiving. For this reason, the database, is extended with tables to store values about Temperature, Smoke, Movement with the corresponding image capture by the use of a camera, Door state (opened or closed) and Lights state (on or off). It is also extended and modified to store information about each user and his preferences like his First Name, Last Name, Telephone, whether he wants to receive any notification, the service to be notified with, such as SMS or Email and general settings for the web app such as the ability to notify the authorities about any dangerous situation the system might find. Last but not least, is has the ability to store any Channel Uri in order to have the ability to send Push Notifications, to the corresponding user and the corresponding device, through Microsoft's Push Notifications Service (MPNS). These extra tables are listed below:

Temperature

Database fields:

Id: Primary key

Value: the temperature's value

Date: the date this value been logged

Smoke

Database fields:

Id: Primary key

Value: indicates whether the sensors detected smoke or not

Date: the date this value been logged

Lights

Database fields:

Id: Primary key

Room: the room that has the lights inside

TurnedOn: whether the lights turned on or off

Date: the date this value been logged

Doors

Database fields:

Id: Primary key

Room: the room that has the doors inside

TurnedOn: whether the door opened or closed

Date: the date this value been logged

Camera

Database fields:

Id: Primary key

Room: the room that has the camera inside

TurnedOn: whether the camera turned on to capture a movement

ScreenshotUrl: the filename of the image camera has captured

Date: the date this value been logged

PushNotificationsUris

Database fields:

PNotificationId: Primary key

UserId: the user's id to associate the channel uri with

ToastUri: the channel uri which used to send a push notification

DateUpdated: the date that channel uri has been updated

GlobalSettings

Database fields:

GSettingsId: Primary key

PoliceEmail: an email to communicate with the local police department

NotifyAuthorities: indicates whether the authorities must be notified for any dangerous situation or not

isAlertModeEnabled: indicates whether something happened and alert mode is on

lastValuesUpdate: this value is the latest date values have been logged

lastRequestReceived: the date that last request for data logged received

lastNotificationSend: the last time that notifications were sent to the users

UserSettings

Database fields:

UserId: Primary key

FirstName: user's first name

LastName: user's last name

Email: user's email

Address: user's home address

Telephone: user's telephone (in international format)

AllowToReceiveNotifications: indicates whether a user is allowed to receive any kind of notification

rntSmoke: option to receive notifications about smoke events

rntTemperature: the date that last request for data logged received

rntIntruder: the last time that notifications were sent to the users

rnttSMS: option to receive notifications through SMS

rnttPushNotification: option to receive notifications through Push on the WP device

rnttEmail: option to receive notifications through Email

rnttPhone: option to receive notifications by a phone call

DateIn: the date each entry first created

The full diagram of the database:

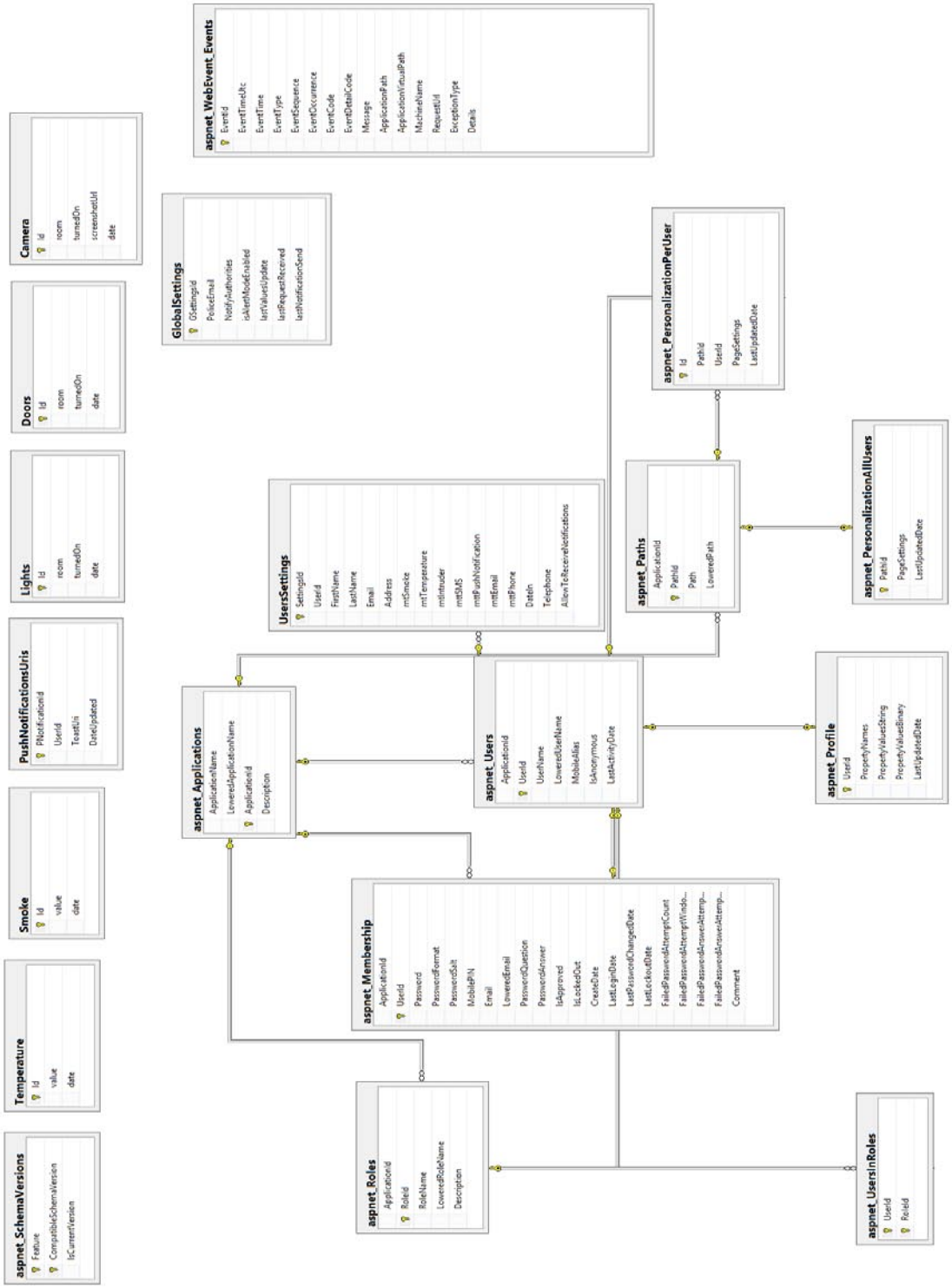

Figure 3.21: Database Diagram

3.3 Application Programming Interface (API)

The existence of an API was a necessity for the server side of this project in order to properly work and communicate with sensors and the Windows Phone application. In more details, the API consists of the following pages:

UserAuth.aspx

Description: This page is responsible to authenticate a user by passing his/hers credentials. If the user is authenticated, returns a response containing the user's id.

Parameters:

username: the user's username

password: the user's password

feed (optional): if json, the response will be in json format else, in xml

Response:

status: "ok" or "error"

info: If ok, it contains user's id. Else, an error message

roles: If ok, it contains user's roles (Admin, Neighbor etc)

GetProfile.aspx

Description: Used to return authenticated user's profile and settings information.

Parameters:

user_id: user's id

feed (optional): if json, the response will be in json format else, in xml

Response:

status: "ok" or "error"

info: If ok, it contains user's id. Else, an error message

settings_id: Settings' id

user_id: User's id

first_name: User's first name

last_name: User's last name

email: User's email

address: User's home address

telephone: User's telephone

allow_ro_receive_notifications: Whether the user is allowed to receive notifications or not

receive_notifications_type_smoke: true or false type

receive_notifications_type_temperature: true or false type

receive_notifications_type_intruder: true or false type

receive_notifications_type_sms: true or false type

receive_notifications_type_push: true or false type

receive_notifications_type_email: true or false type

receive_notifications_type_phone: true or false type

UpdateProfile.aspx

Description: Used to update authenticated user's profile.

Parameters:

username: user's username

user_id: user's id

feed (optional): if json, the response will be in json format else, in xml

first_name: User's first name

last_name: User's last name

email: User's email

address: User's home address

telephone: User's telephone

rntSmoke: 1 or 0 to receive notifications for Smoke or not

rntTemperature: 1 or 0 to receive notifications for Temperature or not

rntIntruder: 1 or 0 to receive notifications for Intruder or not

rnttSMS: 1 or 0 to receive notifications through SMS or not

rnttPush: 1 or 0 to receive notifications through Push Notifications or not

rnttEmail: 1 or 0 to receive notifications through Email or not

rnttPhone: 1 or 0 to receive notifications through Phone or not

Response:

status: “ok” or “error”

info: If ok, it contains user’s id. Else, an error message

DisplayData.aspx

Description: Used to display data that sensors logged (e.g. temperature, whether smoke is detected or not) to authenticated users with ‘admin’ role.

Parameters:

table: the type of data to return (Camera, Doors, Lights, Smoke, Temperature)

user_id: user’s id

feed (optional): if json, the response will be in json format else, in xml

Response:

status: “ok” or “error”

info: If ok, it contains user’s id. Else, an error message

items: if ok, it contains all the items sensor’s logged in order to display them

AddData.aspx

Description: Used to log sensors’ data. Room, temperature, movement etc.

Parameters:

table: the type of data to log (Camera, Doors, Lights, Smoke, Temperature)

user_id: user’s id

feed (optional): if json, the response will be in json format else, in xml

value (if available): the corresponding sensor’s value

room (if available): the corresponding sensor’s room name

turned_on (for true or false values)(optional): if 1, “turned_on” value is True. Else, is False

screenshot (only for camera): the name, camera used to upload the screenshot to the server

data: a string containing all data captured by the sensors

complete_data: if 1, the page will look for data parameter

Response:

status: “ok” or “error”

info: If ok, it contains user’s id. Else, an error message

data_info: information that refers to the data insertion

RegisterPush.aspx

Description: This page is responsible to store a Windows Phone device’s channel uri to receive push notifications and associate it with a user.

Parameters:

channel_uri: device’s channel uri for receiving push notifications

user_id: user’s id

feed (optional): if json, the response will be in json format else, in xml

Response:

status: “ok” or “error”

message: If ok, it contains user’s id. Else, an error message

CheckForConnectivityIssues.aspx

Description: The purpose of this page is to provide to an external service (e.g. windows service or desktop application) the ability to trigger a check if the “sensor’s side” works and there are no connectivity issues, possible hardware damage or something unexpected. If needed, all administrators are notified.

Response:

“ok” if the call was successful.

3.4 The Windows Phone Application

The Windows phone application was deployed on Visual Studio 2010. The Windows Phone Platform selected to be Windows Phone OS 7.1 and latest versions. It was created in addition to the deployed web site.

With the Windows Phone application the user could have the following options:

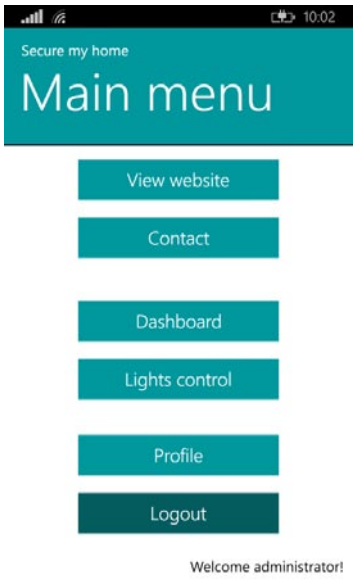

Figure 3.22: WP application main menu

3.4.1 Login page

This page contains a login form in order to login to the web application. After the user fills the form to log in, a request will be made to the web application's api. If there will be no network errors and a response with a successful message for the validation of user's credentials, every piece of information about the user, which comes with the api response, is stored locally on the windows phone IsolatedStorageSettings so they can be accessible any time from the various use cases that the application might need these information.

A user model is presented as SHUser class and it is responsible to login, logout, synchronize user profile information with the web application on the server. Each time the application needs information for the current user (if anyone is logged in) in order to use the api and retrieve data to display in the dashboard page, the user roles or anything else, the model looks into the IsolatedStorageSettings makes any necessary validation and returns the appropriate data.

3.4.2 Profile

Shows the user's profile / settings information and notices if the user has the ability to receive notifications. In the case the user notifications were disabled by the administrator, a message in red box will be shown as it displayed on the next figure.

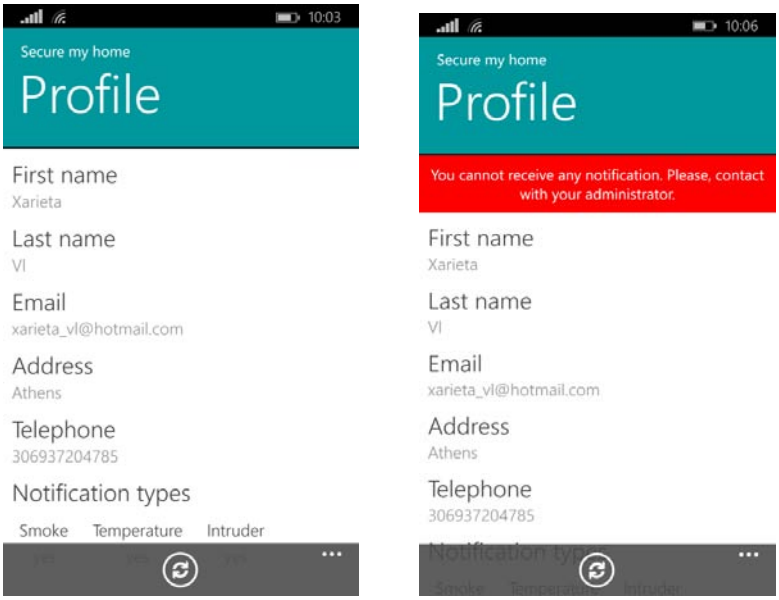

Figure 3.23: The user profile and the message when the user cannot receive notifications.

3.4.3 Dashboard

This page is the equivalent dashboard page web application shows. The user can see the activity of corresponding sensors. Dashboard is accessible only by users with role "Admin".

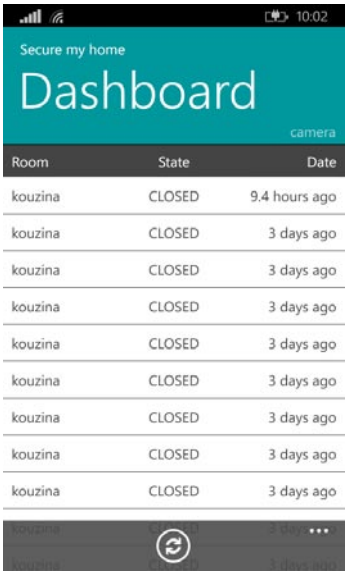

Figure 3.24: The dashboard

3.4.4 Lights control

Lights control option, gives the user the ability to control the lights and switch them on/off on demand. For simplicity, the need of existence a "Server side" infrastructure inside the house in order to communicate properly using the Server-Client model and the purposes of

this prototype, the windows phone device must be connected on the same network as the arduino which controls the lights. This prototype uses a simple led light but any light can be applied without the need to update any software. This option is accessible only by users with role “Admin”.

Figure 3.25: The lights control screen.

3.4.5 View website

This option opens a Web Browser and navigating to the website. Its purpose is to give to the user the ability to access any feature my added to the web application and needs more time for the windows phone app to be implemented.

3.4.6 Contact

This option opens the native mail app in order to give it to the user to communicate through email, with the administrator.

3.4.7 Push notifications

The Microsoft Push notification Service in Windows Phone, as Microsoft notes, is an asynchronous, best-effort service that offers third-party developers a channel to send data to a Windows Phone app from a cloud service in a power-efficient manner.

3.4.7.1 How it works

The following diagram shows how a push notification is sent.

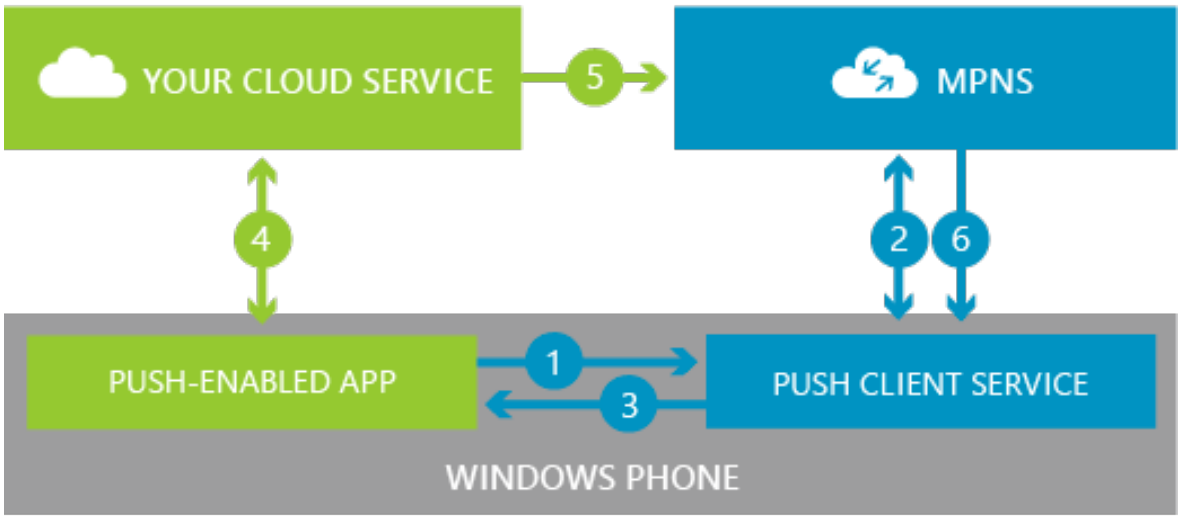

Figure 3.26: How the push notification works

- The app requests a push notification URI from the Push client service.
- The Push client service negotiates with the Microsoft Push Notification Service (MPNS), and MPNS returns a notification URI to the Push client service.
- The Push client service returns the notification URI to the app.
- The app can then send the notification URI to the cloud service.
- When the cloud service has info to send to the app, it uses the notification URI to send a push notification to MPNS.
- MPNS routes the push notification to the app.

Depending on the format of the push notification and the payload attached to it, the info is delivered as raw data to the app, the app's Tile is visually updated, or a toast notification is displayed. MPNS returns a response code to the cloud service after a push notification is sent indicating that the notification has been received and will be delivered to the device at the next possible opportunity. Although MPNS doesn’t provide an end-to-end confirmation that the push notification was delivered from the cloud service to the phone, it is possible for MPNS to return a response or error code to the cloud service which indicates that the notification will not be delivered to the device.

3.4.8 Libraries needed

Additional libraries needed in order to use the API and read responses in JSON format. All libraries are listed below:

- Newtonsoft.Json.WindowsPhone.dll

These libraries are added to the already excepted references of the project.

3.5 Desktop application

In order to catch the exception of an arduino failure, the api, provides the CheckForConnectivityIssues.aspx page that makes all the necessary checks to figure that out. An executable program running on the server's OS calls this part of the API frequently. In case of an exception or a successful call, a corresponding message is printed in a textbox.

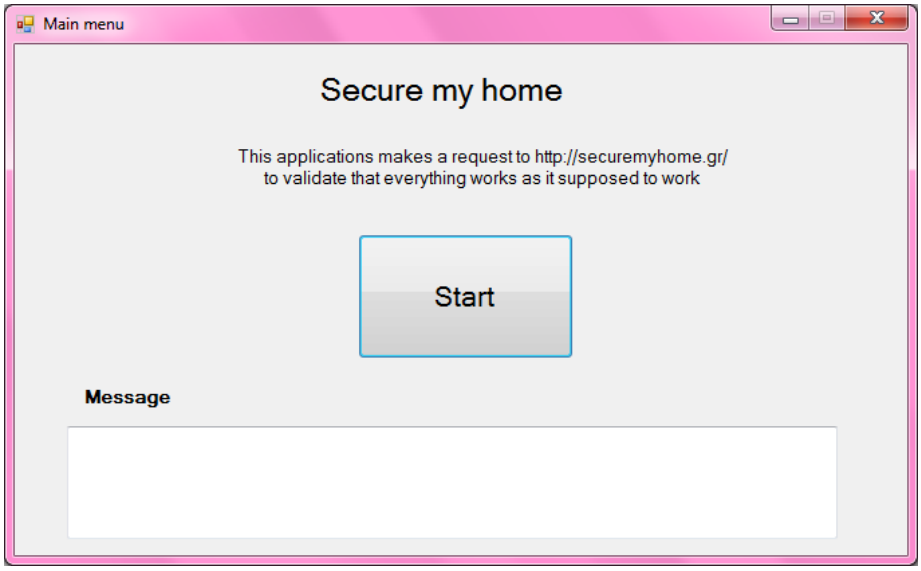

Figure 3.27: Desktop Application main interface

Chapter 5

Conclusions

The findings of the thesis are summarized in this chapter. They consist of evaluation, conclusions and observations, followed by suggestions for future research.

4.1 Aims of thesis

The aims of this thesis were:

1. The designing and implementation of a remote house security and alarming system using Zigbee wireless sensors during the absence of the owner.
2. The control and monitoring of the system via
    a. a user friendly web site,
    b. a Windows Phone application.
3. The web site were designed to be responsive to facilitate users to access the services if the usage of the application is not possible and that because the windows phone application that is implemented does not run to android smart phones or other operating systems.
4. The system targets to inform the user for the condition of the remote house. There was created a system of sensors that send on a predefined time data to the server. For the same reason there were implemented alerts that will be activated in any alteration of the system.
5. When an alert occurs then an e-mail was sent from the server, a push notification and a SMS (depending on the options of the user) and informs the owner for the situation.

6. The system can support detection of smoke, temperature variations, humidity, motion detection, any intrusion on the house from i.e. windows, doors etc.
7. A wireless camera was used for the surveillance of the house. In the case of intrusion detection, the camera will be programmed to capture images of the intruder and send them to the owner, the neighbors (neighborhood alert) and the related authority.

4.2 Evaluation

4.2.1 Literature Review

The literature in the area of wireless sensor networks was studied and particularly on zigbee technology. The research focused on XBees communication and furthers more the communication with arduino boards that were used. It has also to be studied some topics of electronics for the correct connection of the different modules and wires on the circuit.

The main source of information comes from the official web pages of manufacturers such as zigbee Allianz Digi, Arduino and Sparkfun that provide hardware specifications, datasheets, tutorials, a number of presentations about its devices, forums and communities.

On the other side, for information about the implementation of the web site and the windows phone app the main source of information will be the MSDN which includes how-to and reference documentations, sample codes, technical articles etc. As secondary sources are the w3scools for the designing of the web site and bootstrap official page to make it responsive.

Additionally, searching on the internet will help to find many other papers from several universities, conferences and journals.

4.2.2 The Proposed System

A prototype was constructed for the need of the thesis. It could be divided in two main parts the Home side which is the wireless sensor network and the Server side which is responsible for maintaining the received data, control and monitoring.

A number of scenarios were utilized to check the responsiveness of the alerts. On these included:

- The creation of smoke to trigger the alert of smoke
- The increasing of the temperature. A lighter was placed near the DHT11 sensor to increase its temperature. It was not possible to reach the sufficiently high temperatures in the case of a fire so for this reason the temperature tested in smaller temperatures.
- The magnetic reed switch also tested and it works.
- The PIR sensor module GH-718 works when it detects a motion to its sentry angle.

The correct transmition of the values between home (arduino UNO and XBee router) and receiver (arduino Yún and XBee router) checked with the use of arduino terminals, where displayed the transmitted information.

In order to catch the exception of an arduino failure, the api, provides the CheckForConnectivityIssues.aspx page that makes all the necessary checks to figure that out. This part of the api is called by an executable program running on the server's desktop.

4.3 Recommendations for Future Research

The proposed system is designed in such a way to easily extended with more functionalities. An interesting field of research could be the extension of the already system to support a fire extinguishing system. In case of an intruder the house could lock him in order to give more time to the authorities to reach to the house. Considering that operating systems are starting to be able to work on cars (e.g. iOS in the Car), another case of research could be the ability to automatically set the house, as destination in a car's gps system in case of emergency.

4.4 Conclusions

The aim of this project was the designing and implementation of a remote house security and alarming system using ZigBee wireless sensors during the absence of the owner. The prototype responds to all the proposed functionalities and even more. It has been implemented the logic to receive push notifications and already supports multi rooms (for example it is “on” the light in the room kitchen).

The cost of equipment (arduino boards, XBees, breadboards, XBee explorers, wires, led, resistors etc.) remained at reasonable levels compared with the relatively high prices of the available security systems.

Bibliography

‘ZigBee® Wireless Networking Overview’, Texas Instruments (2013)

Banzi, M. (2008), ‘Getting started with Arduino’, O’ Reilly

XBee OEM RF Modules, ‘Product Manual v1.x.1x - ZigBee Protocol’, ZigBee, Digi International (2007)

ZigBee Specification, ZigBee Alliance (2005)

XBee®/XBee-PRO® ZB RF Modules, Digi International (2014)

‘X-CTU Configuration & Test Utility Software’, Digi International (2008)

Faludi, R. (2011), ‘Building Wireless Sensor Networks’, O’ Reilly

Spaanjaars, I. (2013), ‘Beginning ASP.NET 4.5 in C# and VB’, John Wiley & Sons, Inc.

D-Robotics (2010), ‘DHT11 Humidity & Temperature Sensor’

‘DHT11 Product Manual’, www.aosong.com

Hwang, S., Yu, D. , (2012), ‘Remote Monitoring and Controlling System Based on ZigBee Networks’, International Journal of Software Engineering and Its Applications Vol. 6, No. 3, July, 2012.

Sharma, U. , S.R.N. Reddy (2012) ‘Design of Home/Office Automation using Wireless Sensor Network’, International Journal of Computer Applications, Vol43– No.22, April 2012

Zhang, Y., Zhao, G., Zhang, Y., (2009), ‘A Smart Home Security System Based on 3G’, Computer Science-Technology and Applications, 2009. IFCSTA '09.

Husina, M.H., Tening, E.B., Sabri, M.F.M., (2012), ‘Design & Implementation of Home Security System using Verilog-HDL’, Elsevier

Chun-Liang Hsu, Sheng-Yuan Yang, Wei-Bin Wu.(2009), ‘Constructing intelligent home-security system design with combining phone-net and Bluetooth mechanism’, International Conference on Machine Learning and Cybernetics 2009.

Jawarkar, N. P., Ahmed, V., Ladhake, S. A, Thakare, R. D , (2008), 'Micro-controller based Remote Monitoring using Mobile through Spoken Commands', Journal of Networks, Vol. 3, No. 2, February 2008

Aggarwal, A. , Joshi, R.C. , (2012), 'WSN and GSM based Remote Home Security System', International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012)

Kijudomsina, P. , Pavaganunb, C., Kaviyac, S., Nachareed, W. and Yupapine, P. P. , (2011) 'Home Intelligent Based on Privacy Camera and Home Alarming Systems', Published by Elsevier.

Baronti, P., Pillai, P., Chook, V., Chessa, S., Gotta, A., Hu1, Y. F., 'Wireless Sensor Networks: a Survey on the State of the Art and the 802.15.4 and ZigBee Standards'

Hogrefe, D., (2011), 'Wireless Ab Hoc & Sensor Networks'

Buratti, C., Conti, A., Dardari, D. Verdone, R., (2009), 'An Overview on Wireless Sensor Networks Technology and Evolution'

Sohrabi, K., Gao, J., Ailawadhi, V. Pottie, G., (2000), 'Protocols for self-organization of a wireless sensor network.' IEEE Personal Commun.

Web sites

http://getbootstrap.com

http://grobotronics.com

https://www.sparkfun.com/

http://arduino.cc/

http://forum.arduino.cc

http://www.digi.com/

http://www.zigbee.org/

http://playground.arduino.cc/Main/MQGasSensors

https://github.com/amicojeko/YouCantTouchThis

http://msdn.microsoft.com/en-us/library/windowsphone/develop/ff402558(v=vs.105).aspx

http://json.codeplex.com/