**M S Ramaiah Institute of Technology**

(An Autonomous Institute, Affiliated to VTU)

MSR nagar, MSRIT post, Bangalore-54

A Dissertation Report on

**Smart Fire Fighting**

Submitted by

**SPARSH GUPTA 1MS12CS110**

**VIKAS KUMAR 1MS12CS130**

**TUSHAR 1MS12CS122**

**SANDEEP KUMAR 1MS12CS097**

*in partial fulfillment for the award of the degree of*

# *Bachelor of Engineering in Computer Science & Engineering*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**M.S.RAMAIAH INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

www.msrit.edu

# Abstract

Fire losses throughout the world remain too high and fire fighting too hazardous. The fire service and other emergency first responders are currently benefiting from enhanced-existing and newly-developed electronic technologies. Fire fighters are now operating in an ever increasing sensor rich environment that is creating vast amounts of potentially useful data. The "smart" fire fighter of tomorrow is envisioned as being able to fully exploit select data to perform work tasks in a highly effective and efficient manner.

Available data, the comprehensive ability to analyse and process this data, and an increasingly sensor rich environment are all opening new possibilities for the fire service to address unwanted fires. This involves all manner of their job performance duties, and includes during pre-fire, trans-fire (i.e. during the event) and post-fire stages. This project is focused on developing the research roadmap to clarify the research needed to most effectively use the immense quantity of available data, the computational power to compute and communicate that data, the knowledge base and algorithms to most effectively process the data, convert it into significant knowledge/beneficial decision tools, and effectively communicate the information to those who need it --- on the fire ground and elsewhere.

This project senses the temperature and smoke in a room and sends the data to the cloud. In case of fire an emergency message is sent to the cloud server which can be analyzed by the nearest fire rescue team and the proper action will be taken.

The use of technology, referred here as **Smart Fire Fighting**, includes all areas of fire protection engineering and phases of fire service emergency response: pre-incident, during-incident, and post-incident. Smart Fire Fighting will transform traditional fire prevention and protection strategies and fire-fighting practices by ensuring the flow of critical information where and when it is needed. This flow will be achieved by increasing the power of information through enhanced data gathering, processing, and targeted communications.

#### Contents

***Declaration i***

***Acknowledgements ii***

***Abstract iii***

***List of Figures***

***List of Tables***

1 **INTRODUCTION**

* 1. General Introduction
  2. Statement of the Problem
  3. Objectives of the project
  4. Project deliverables
  5. Current Scope
  6. Future Scope

1. **PROJECT ORGANIZATION**
   1. Software Process Models
   2. Roles and Responsibilities
2. **LITERATURE SURVEY**

3.1 Introduction

3.2 Main Body

3.3 Conclusion of Survey

1. **SOFTWARE REQUIREMENT SPECIFICATIONS**

4.1 Product Overview

4.2 External Interface Requirements

4.2.1 User Interfaces

4.2.2 Hardware Interfaces

4.2.3 Software Interfaces

4.2.4 Communication Interfaces

4.3 Functional Requirements

4.3.1 Functional Requirement 1.1

:

4.3.n Functional Requirement 1.n

4.4 Software System Attributes

4.4.1 Reliability

4.4.2 Availability

4.4.3 Security

4.4.4 Portability

4.4.5 Maintainability

4.4.6 Performance

4.5 Performance Requirements

4.6 Database Requirement

4.7 Design Constraints

4.8 Other Requirements

1. **DESIGN**
   1. Introduction
   2. Architecture Design
   3. Graphical User Interface
   4. Class Diagram and Classes (represent Inheritance, Aggregation and Association)
   5. Sequence Diagram
   6. Data flow diagram
   7. Metric calculation
2. **IMPLEMENTATION**
   1. Tools Introduction
   2. Technology Introduction
   3. Overall view of the project in terms of implementation
   4. Explanation of Algorithm and how it is been implemented
   5. Information about the implementation of Modules
3. **TESTING**

**7.1** Results and Snapshots

**7.2** **SECURITY PROTOCOL FOR IOT**

1. **CONCLUSION & SCOPE FOR FUTURE WORK**
2. **REFERENCES**

##### Declaration

This is to certify that Mr. / Ms

**SPARSH GUPTA 1MS12CS110**

**VIKAS KUMAR 1MS12CS130**

**TUSHAR 1MS12CS122**

**SANDEEP KUMAR 1MS12CS097**

of 7th semester, B.E in Computer Science and Engineering has satisfactorily completed the subject requirements of the course on **“CSPE734 : Project Based Learning: Internet Of Things”**, prescribed by the Department of Computer Science and Engineering for the B.E course of the institute during Aug-Dec 2015.

##### Acknowledgements

I would like to extend my sincere & heartfelt obligation towards all the personnel who have helped me in this project. Their active guidance, help, cooperation & encouragement have helped me throughout the project.

I would like to thank Dr K G Srinivas and Dr Anita Kannavalli for providing conscientious guidance and encouragement to accomplish this assignment.

I am extremely thankful and pay my gratitude to Ms Soumya BJ, Mr Pradeep Kumar and other faculty members for their valuable guidance and support on completion of this project.

I also acknowledge with a deep sense of reverence, my gratitude towards my parents and member of my family, who has always supported me morally as well as economically.

At last but not least gratitude goes to all of my friends who directly or indirectly helped me to complete this project report.

Thanking You

Sparsh Gupta

Vikas Kumar

Sandeep Kumar

Tushar

##### List of Figures

*Figure 1 : Internet of Things*

Figure 2 : Transformation from traditional based fire fighting to Smart Fire Fighting

*Figure 3 : Voice Based Home Automation*

*Figure 4 : Impact of IOT in daily life*

*Figure 5 : IOT controlling everything*

*Figure 6 :Architecture of Raspberry Pi 2*

Figure 7 :MQ2 Sensor Connection Diagram

Figure 8 :MQ2 Gas Sensor

*Figure 9 : Pin Connections for temperature sensor DS18B20*

*Figure 10 :Connection Module of Raspberry Pi 2*

*Figure 11 :Unity Graphical User Interface*

*Figure 12 :Smart Fire Fighting Class Diagram*

*Figure 13 :Smart Fire Fighting Sequence Diagram*

*Figure 14 :Smart Fire Fighting Activity Diagram*

*Figure 15 :PUTTY Configuration*

*Figure 16 :PUTTY Terminal*

*Figure 17 :VNC Viewer Authentication*

*Figure 18 :VNC Viewer GUI*

*Figure 19 :VNC Viewer Terminal*

*Figure 20 :Ubidots Web Page*

*Figure 21 : Placing the Smart Fire Fighting product (Sample Layout)*

*Figure 22 : Raspberry PI Module*

*Figure 23 : Raspberry PI Specifications*

*Figure 24 : DS18B20 Diagram*

*Figure 25 : MQ2 Specification*

*Figure 26 : Python Code*

*Figure 27 : Python Code(Continued)*

*Figure 28 : Python Code Sample Output*

*Figure 29 : Python Code Sample Output*

*Figure 30 : Ubidots Source Creation*

*Figure 31 : Ubidots Smoke Source*

*Figure 32 : Ubidots Smoke Source Data*

*Figure 33 : Ubidots Temperature Source*

*Figure 34 : Ubidots Temperature Source Data*

*Figure 35 : Event Generation for Smoke Readings (GMail)*

*Figure 36 : Event Generation for Temperature Readings (GMail)*

# 1. INTRODUCTION

**1.1 General Introduction**

The **Internet of Things** (**IoT**) is the network of physical objects or "things" embedded

with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data. The Internet of Things allows objects to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more direct integration between the physical world and computer-based systems, and resulting in improved efficiency, accuracy and economic benefit.

"Things," in the IoT sense, can refer to a wide variety of devices such as heart monitoring implants, electric clams in coastal waters, automobiles with built-in sensors, DNA analysis devices for environmental/food/pathogen monitoring or field operation devices that assist firefighters in search and rescue operations. These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. Current market examples include smart thermostat systems and washer/dryers that use Wi-Fi for remote monitoring.

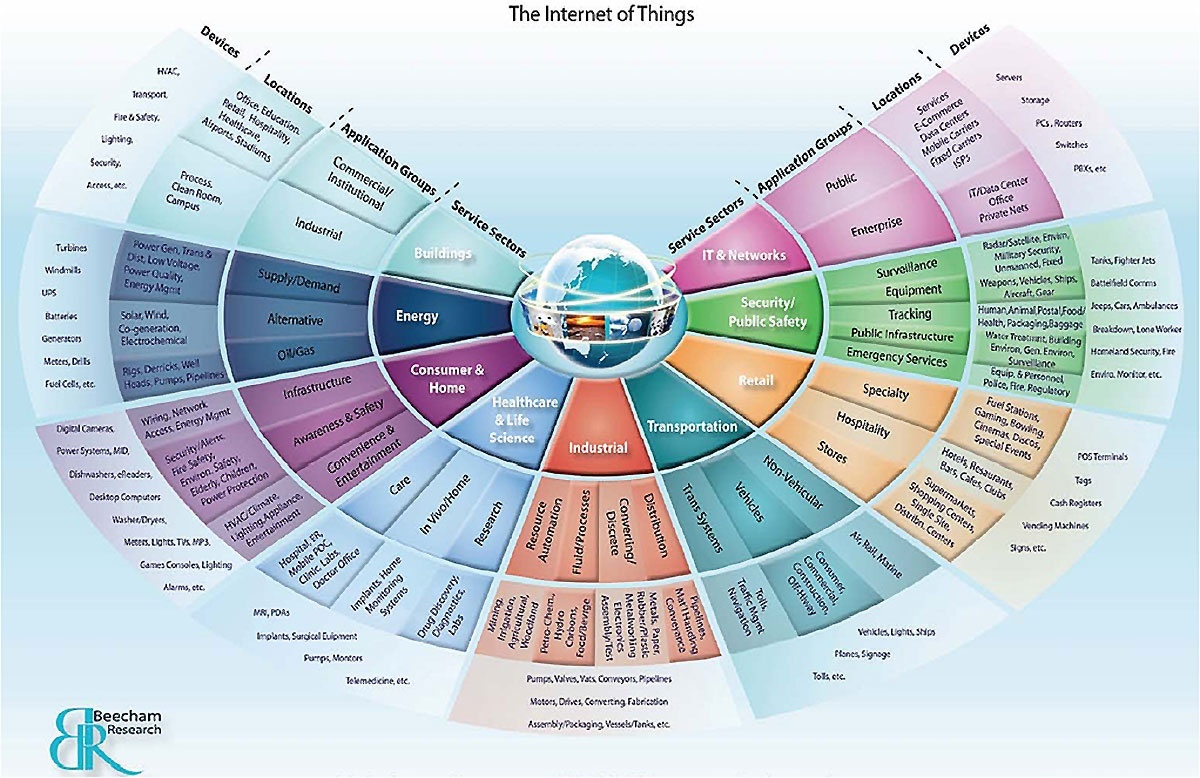


Figure 1 : Internet of Things

When it comes to putting out fire, time is often the most important factor to consider. Time, and information. The more data about the burning structure – and the people trapped in it – is available to firefighters, the more they can operate effectively, saving lives and limiting the damages to the building.

There are four things that are most important to the development of smart fire fighting technologies:

* use of sensors on the fire ground to assist in situational awareness and personnel location
* increased collection and utilization of data before the incident to aid in effective use of personnel and equipment
* enhance interoperability between data systems; and
* develop intelligent systems to assist with decision-making.

Smart Fire Fighting demonstrates how enhanced data gathering, processing and delivery could transform traditional fire protection and fire fighting practices, combining the points of strengths of "Internet of Things" with the big data analytics.

**1.2 Statement of the problem**

Fire losses throughout the world remain too high and fire fighting too hazardous. Today, data from a variety of sources are collected independently and processed separately, but evolving new technologies are enabling the use of vast amounts of information. The use of this information continues to demonstrate great promise at enhancing the effectiveness and efficiency with the duties handled by fire fighters. Importantly this equates to improved safety and health for this high risk profession, via situational and incident awareness and other factors. Knowledge is power, and harvesting the data important to fire fighters is empowering the smart fire fighter of the future.

Enhancing the fire rescue team efficiency in controlling and rescuing people in fire. This project senses the temperature and smoke in a room and sends the data to the cloud. In case of fire an emergency message is sent to the cloud which can be analyzed by the nearest fire rescue team and the proper action will be taken.

The use of technology, referred here as **Smart Fire Fighting**, includes all areas of fire protection engineering and phases of fire service emergency response: pre-incident, during-incident, and post-incident. Smart Fire Fighting will transform traditional fire prevention and protection strategies and fire-fighting practices by ensuring the flow of critical information where and when it is needed. This flow will be achieved by increasing the power of information through enhanced data gathering, processing, and targeted communications.

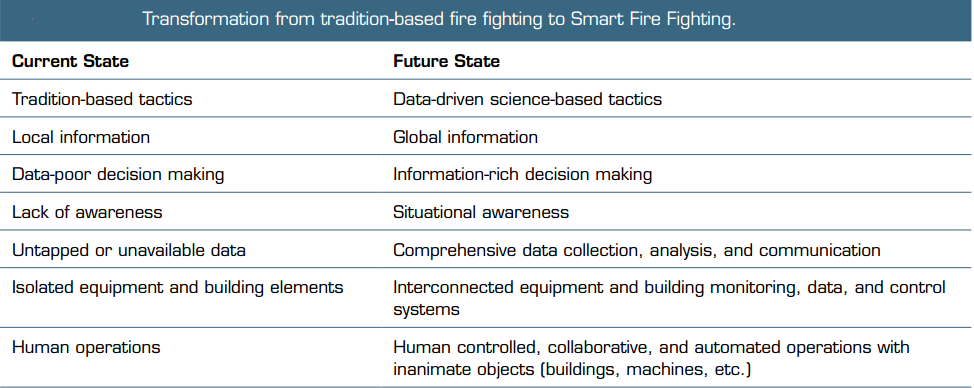


Figure 2 : Transformation from traditional based fire fighting to Smart Fire Fighting

**1.3 Objectives of the project**

The goal for Smart Fire Fighting is to remove unwanted fire and other harmful events as a limitation to life safety, technical innovation, and economic prosperity. It remotely measures the room temperature and smoke variations using **Raspberry Pi 2**.

A **DS18B20** sensor is used to measure real time temperature and **MQ2** sensor is used to measure smoke levels. Both the sensors update the readings in **Open Source Cloud** and in the **database**. The temperatures are constantly monitored and printed on the console. Also an **email** **alert** is sent to the intended recipient if the inside room temperature or smoke levels exceeds a certain predefined limit.

The vision to improve fire protection and fire fighting is undertaken with the following motivations:

* To save lives and minimize injuries to building occupants and community members due to fire.
* To improve fire fighter occupational health and safety.
* To enhance the operational efficiency of the fire service and the effectiveness of fire protection.
* To minimize property loss from fire.
* To minimize business interruption and loss of mission continuity due to fire.

**1.4 Project deliverables**

The deliverables for this project collectively review the available baseline information, identify the fundamental principles and key details involving fire/rescue tactics and strategy, provide a summary of core basics, and address and clarify related issues such as infrastructure needs, areas needing further research, and other applicable topics.

**1.5 Current Scope**

The scope of the project is to prepare a SMART system that will enhance the fire rescue team efficiency in controlling and rescuing people in fire. The system should be -

* ***S****calable* and robust and provide custom information at appropriate periods and in suitable data forms, as required by different applications and services.
* ***M****onitored and managed*easily. If software on remote sensors must be updated, the sensors need to be discoverable no matter where they are. That requires an efficient management approach.
* ***A****daptable* to the sensors changing conditions or context while being able to communicate automatically to other sensors.
* ***R****eliable.* Data uploaded wirelessly to a cloud must be dependably transmitted and reported.
* ***T****rustworthy.* A mechanism is needed to ensure data are not being manipulated while in transit and that only trusted parties can access sensitive data.

**1.6 Future Scope**

The future scope of the project will be as follows

a) Almost all things will work on your voice in future as described by this image.

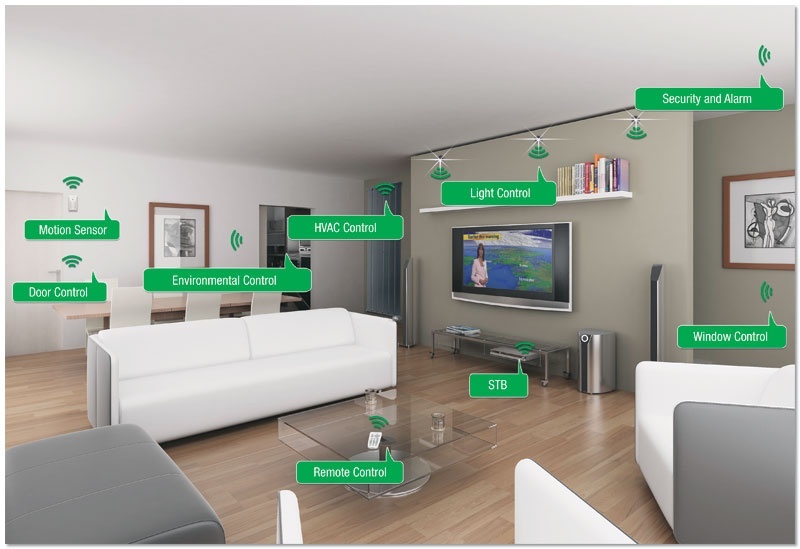


Figure 3 : Voice Based Home Automation

b) Another example of "Almost Everything is in Control with IoT".



Figure 4 : Impact of IOT in daily life

c) You will control the inside environment of your home from outside.



Figure 5 : IOT controlling everything

# 2. PROJECT ORGANIZATION

**2.1 Roles and Responsibilities**

# When the fire detection system detects the fire it should be able to alert the fire fighters as soon as possible, for which it needs to send an alert in form of data. This alert generated by the sensors can also be used to trigger an alarm or a water sprinkle system. This can be used to provide safety and also the information to the fire brigade is sent without any human delay.

# Electronic circuits can be designed for the fire based alarms and they provide very high efficiency and can be used for the security reasons. Early fire detection is best achieved by the installation and maintenance of fire detection equipment in all rooms and areas of the house or building.

# The early notification that premises are on fire is essential in the protection of life. It is accepted that the chance of a fire occurring and not being appropriately detected when a suitable fire detection system is in place is relatively small. However, the consequences of a malfunction or lack of trust in the system by the occupants of the property can be catastrophic.

# The matters which the system must address are :

# ¬ That a single system is named as the system which should undertake checks, arrange for testing, maintain a log book and report the faults to pre-assigned responsible person.

# ¬ Ensure that every occupant is aware of who the responsible person is and their contact details which must be available 24 hours a day. If the primary responsible person is not available for whatever reason, a secondary person must be appointed and their contact details provided to all tenants.

# ¬ Ensure that every occupant is aware of what the alarm sounds like and what they should do if the alarm sounds. When introducing a new tenant the alarm must be activated to ensure they are familiar with the sound of the alarm, especially how it will be heard in their accommodation.

# ¬ The system must be checked for the control and indicating equipment frequently and at least weekly to confirm there are no faults and record these checks clearly in the log book.

# ¬ The responsible person must make arrangements for testing and maintaining the equipment and retaining the consequent test certificates, which should not exceed six months, but can be less depending on your risk assessment.

# 3. LITERATURE SURVEY

**3.1 Introduction**

Due to the randomness of fire and variations in building and occupant characteristics, it is difficult to set up a general step-by-step performance-based fire safety design that could apply to all buildings. Therefore, every building should be evaluated according to its specific geometric features, its use and its occupancy. However when performing a performance-based fire safety design, there are four generic steps that should be followed.

1. Identification of performance objectives and requirements.

2. Establishment of performance criteria.

3. Quantification process.

4. Presentation of design documentation to the Authority Having Jurisdiction for approval.

**3.2 Main Body**

Fire safety in buildings is an interaction among all the components of the fire safety system which includes fire outbreak, fire growth, fire and smoke spread, the response of building elements to fire, the occupant response to fire and the fire service response to fire. In order to develop a building fire safety design, it is necessary to determine the behaviour of a fire in a fire compartment from ignition to decay. Therefore, it is essential that the designers have at their disposal the means to predict the level of life safety for any particular design. The means are, in general, in the form of fire engineering computer models that can be used to estimate the performance of building fire safety systems and evaluate the compliance with the performance criteria set initially by the design team.

Over the past few years, considerable effort has been put into the development of fire engineering computer models. Fire models can be grouped into two categories: probabilistic or stochastic fire models and deterministic fire models. Probabilistic fire models involve the evaluation of the probability of risk due to fire based on the probabilities of all parameters influencing the fire such as human behaviour, formation of openings and distribution of fuel load in the compartment of fire origin. Deterministic fire models are based on physical, chemical and thermodynamic relationships and empirical correlations used to calculate the impact of fire. Deterministic models can be very simple requiring a short computing time or highly complex requiring hours of computation. Typically, deterministic models can be classified as zone models and field models.

The overall objectives of the fire safety systems in buildings are as follows:

• To minimize risk to life and injury to people from fires.

• To minimize property loss in the building of fire origin and adjacent buildings.

• To limit the economic, operational, social and environmental impacts of fires.

**Fire Safety Objectives and Requirements -**

The following sections describe objectives and requirements, for consideration by the design team, in the following areas:

• fire outbreak and development

• spread of fire and smoke;

• means of notification and evacuation;

• fire resistance and structural stability;

• emergency response operations;

• economic and social impacts;

• environmental protection.

**3.2.1 Fire outbreak and development**

*Objective:*

* To reduce the probability of ignition of fire and limit fire growth.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that the outbreak and growth of fire is minimized.

*Performance Requirements:*

* Construction materials shall be selected to minimize the potential of fire ignition and to limit fire growth.
* Fixed appliances, using controlled combustion of fuel, shall be installed, operated and maintained so that the potential of explosion of the appliance and fire ignition or rise of temperature of combustible building elements, is minimized.

**3.2.2 Spread of fire and smoke**

*Objective:*

* To minimize the spread of fire and products of combustion from the compartment of fire origin to adjacent compartments and to other buildings.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that:
* The spread of fire and products of combustion in the building is minimized.
* The spread of fire to adjacent buildings is minimized.

*Performance Requirements:*

* Buildings may be divided into fire compartments, designed and maintained to minimize the spread of fire and products of combustion to other compartments of the building and to adjacent buildings.
* Buildings may be provided with automatic fire suppression systems in accordance with the life safety and property loss goals.
* Buildings may be provided with automatic smoke control systems in accordance with the life safety and property loss goals.
* Surface finishes on building elements shall be of materials that reduce fire spread, and minimize generation of toxic gases and heat.
* Buildings may be provided with fire fighting equipment for use by trained occupants.
* Buildings shall be provided with sufficient separation to minimize building to building fire spread.

**3.2.3 Means of notification and evacuation**

*Objective:*

* To notify building occupants of the need to take action in the event of a fire and to protect them from the effects of fire during evacuation.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that occupants shall be able to evacuate safely or remain in a safe place without being exposed to the harmful effects of fire.

*Performance Requirements:*

* Where appropriate, buildings shall be equipped with detection and occupant warning systems for notification of fire.
* Buildings shall be designed, constructed and operated with adequate, easy-to-access and easy-to-use means of egress, such as exits, doors, escape routes and safe refuge areas.

**3.2.4 Fire resistance and structural stability**

*Objective:*

* To minimize building structural failure so that building occupants and emergency responders are protected and property losses and damage to the building and to adjacent properties are minimized.

Functional Requirements:

* Building structural elements shall be designed, constructed and maintained so that the load-bearing capacity is provided in accordance with the life safety and property loss goals.

*Performance Requirements:*

* Building structural elements shall have a fire resistance rating based on the type of structural element, the fire load, the height of the building, the fire exposure, the fire suppression measures, the occupancy and occupant profile, and the emergency team response characteristics.

**3.2.5 Emergency response operations**

*Objective:*

* To support and facilitate the operations of emergency responders.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that emergency responders can locate the fire and fire fighting and rescue operations are facilitated.

*Performance Requirements:*

* Where necessary, buildings shall be provided with means to identify the fire location in the building.
* Buildings shall be provided with safe external and internal access routes to facilitate rescue and fire fighting operations.
* Buildings with internal refuge areas shall be provided with means of communication with the emergency responders.
* Where appropriate, fire fighting water supplies and equipment shall be provided.

**3.2.6 Economic and social impacts**

*Objective:*

* To minimize business loss and interruption to levels acceptable to the owner, federal government authorities and the community.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that economic and social impacts resulting from a fire are minimized.

*Performance Requirements:*

* Fire safety measures shall be implemented to limit damage to ongoing business viability, to valuable building contents, and to buildings of historic value.

**3.2.7 Environmental protection**

*Objective:*

* To minimize environmental damage from the effects of fire.

*Functional Requirements:*

* Buildings shall be designed, constructed and operated so that, in the event of a fire, the release of hazardous materials to the environment is minimized.

*Performance Requirements:*

* Buildings shall be provided with measures to minimize the release of hazardous substances to air, water, and land in the event of a fire.

**False Alarms :**

Regular false alarms can lead to occupants of properties becoming complacent and failing to respond to a sounding alarm with the appropriate urgent action. It is imperative therefore to record and take steps to ensure false alarms are kept to the very minimum possible and that your actions are communicated to all occupants in order for them to maintain confidence in the system.

A false alarm can be divided into 4 categories :

a) Unwanted alarms – in which the system has responded either as designed or as the technology may reasonably be expected to respond to any of the following : - a fire like phenomenon or environmental influence (smoke from a nearby bonfire, dust or insects, processes that produce smoke (burnt toast). - Accidental damage - Inappropriate human action (eg unplanned operation of the system).

b) resulted from a fault in the system

c) malicious false alarms in which a person operates a manual call point or causes a fire detector to initiate a fire signal whilst knowing there is no fire.

d) False alarms with good intent in which a person operates a manual call point in the belief that there is a fire.

**3.3 Conclusion of the survey**

This paper presented an update on the efforts made to move towards performance-based codes and discussed the elements of a performance-based code, as well as, the need to establish performance criteria that can be used to evaluate fire safety designs.

# 4. SOFTWARE REQUIREMENT SPECIFICATIONS

**4.1 Product Overview**

**Product Perspective**

The project is a new, self-contained product intended for use on the Raspberry pi platform. While the mobile application or web application is the main focus of the project, there is also a server-side component which will be responsible for database and synchronization services using open source Cloud services such as Ubidots.

The scope of the project encompasses both server- and client-side functionalities, so both aspects are covered in detail within this document.

**Product Features**

The following list offers a brief outline and description of the main features and functionalities of the Smart Fire Fighting system.

The features are split into two major categories:

* core features - Core features are essential to the application’s operation, whereas additional features simply add new functionalities.
* additional features - The latter features will only be implemented as time permits.

**Core features -**

1. The sensors senses the environment around it and continuously sends data to the cloud server.

2. The cloud server is integrated with the fire fighting departments so that the data can be read by authorized persons and detect whether a fire hazard is going to occur or has occurred or not.

3. Synchronizes to a computer to obtain databases previously downloaded from a web application, using a password-protected secure login.

4. Push Notifications appears if any significant event occurs. It alerts the members.

**Additional Features -**

1. GPS Tracking : Stores data and utilizes Google maps to display the location from which the data is sent using the location of the IP address of raspberry pi.

**4.2 External Interface Requirements**

**4.2.1 User Interfaces**

The software runs on Windows platform . User interface is via mouse and key board they can be used to navigate through different modules and select them.

The user interface for the system will be a web page and a cloud service on the Internet. The user interface will be limited to the types of controls that can be generated using HTML, Javascript, and Cascading Style Sheets.

User interface includes access of a web page and a notification in the case of emergencies. The concerned authority can access the data through the web interface to derive insights of the data collected and use it for fire prevention purposes.

**4.2.2 Hardware Interfaces**

**Hardware requirements -**

* The requirements mentioned below are only needed for the initial configuration and loading of OS into Raspberry Pi 2 -

15.6 inch monitor

HDMI-VGA converter

Keyboard and mouse

* Raspberry Pi 2 Model B 1GB
* DS18B20 sensors for measuring temperature
* MQ2 smoke sensor for measuring smoke levels
* Jumper Wires
* 5volt ,1ampere Power adapter
* 150mbps Mini Wireless USB adapter
* RJ45 LAN cable
* Memory card which is class 10 or higher
* Internet Connectivity

**4.2.3 Software Interfaces**

The software interface includes Raspberry Pi 2 platform.

* Operating System
  + The software is being designed to run on Windows platform.
* Web Server
  + The software is being designed to run on Internet any Server version.
* Programming Language
  + The programming is done in python language and is sent to the raspberry pi board's flash memory.
* Operating System used in Raspberry Pi -
* Raspbian is a free operating system optimized for the Raspberry Pi hardware. An operating system is the set of basic programs and utilities that make your Raspberry Pi run. However, Raspbian provides more than a pure OS: it comes with over 35,000 packages, pre-compiled software bundled in a nice format for easy installation on your Raspberry Pi.
* Putty- Telnet Software
* SD Card Formatter- To clean the card contents for loading OS
* Angry IP Scanner- To obtain the IP address of raspberry pi
* VNC Viewer- Emulation of raspberry pi OS on windows.

**4.2.4 Communication Interfaces**

* **Web Interface**
  + The application will be accessed over the Internet. All features will accessible through the web site.
* **Cloud Server**
  + The cloud server used is UBIDOTS. It is used to capture and store sensor data. The benefits of using Ubidots server are -

a)Setup your device to send data to our cloud

b) Watch your data in a clean, real-time dashboard

c) Trigger alerts when a sensor value hits a threshold

d) Extend your project through a powerful and flexible API

The data read by the sensors is pushed onto the cloud server. The values read are initially passed to an IP Address from where it is retrieved and stored in the database for further analysis.

The communication protocol used are FTP and HTTP/HTTPS.

**4.3 Functional Requirements**

The system has been divided into four modules -

* Data owner module
* Server module
* Receiver module
* Data Storage module

**4.4 Software System Attributes**

**4.4.1 Reliability**

* Data, as analyzed, must be correctly stored in the cloud server. In addition, the cloud database should use mechanisms so that partial data are not stored. Because the user of the system may be inexperienced, each feature must work correctly 99% of the time. The system is available all time and mainly the reliability of the system depends on the sensors and the board.
* The system must run stably with all features mentioned above.
* The system must be debugged completely.
* All exceptions must be handled well.

**4.4.2 Availability**

* The system must be available for use and there has to be continuous monitoring of

data. As a result the system have to be available all time.. The web site and cloud server should have 99% uptime.

**4.4.3 Security**

* The service is made available only to the fire fighting authorities. The data is made secure by making it available only to the authorized people.
* Any modification to the database and raw data on cloud server shall be synchronized and be done only by the admin.
* As the source code or the software is developed under the user machine, no security is provided to the user code. The system security itself is the security i.e. applied to the application.
* In case of forgotten passwords and unable to recover them, contact the developers.

**4.4.4 Portability**

* The system should be portable and must run on different platforms.

**4.4.5 Maintainability**

* The code and design need to be documented well. System is easily maintainable as it has fewer components and its design is not complicated. The system also provides capability to back-up the data.

**4.4.6 Performance**

Performance details are given in section 4.5

**4.5 Performance Requirements**

The performance of the system is determined by processor speed and the internet connection and hence performance varies based on these factors.

* **Modem Connection**: The users should have at least 56Kbps modem connection to the Internet should expect to be able to load any page within 10 seconds.
* **Broadband Connection**: The users of this web page who have a broadband connection should be able to load every page within 2 seconds.
* **Number of Users**: The number of supported users will be unlimited since open source cloud offers unlimited storage. This means that at any given time, the application will be able to maintain the performance levels specified in Modem and Broadband connection.

The performance is also measured in terms of the output provided by application. Requirement specification plays an important part in the analysis of the system.

The **Requirement specification** can be stated as follows -

* The system should be able to interface with the existing system.
* The system should be accurate.
* The system should be better than existing systems.

The system has very high detection rate (i.e., no less than 99%) in any circumstances. Similarly the system has very low false alarm rate (i.e., no more than 1%) in any circumstances. These requirements shall be achieved by using adaptive thresholding and a combination of several algorithms.

**4.6 Database Requirement**

* Database

The database here is stored in Ubidots cloud server.

* Operating System

The Development environment shall be Windows 7 and above.

* Web-Based

The system shall be a Web-based application.

**4.7 Design Constraints**

1. Synchronization:

* Uses USB 3.0.
* Connects only to Windows 7 and above.

2. Memory:

* Device will have 16GB class 10 SD card. Software and database cannot exceed this amount.
* Device will have a SD card slot, and the software must be able to read and write to that slot.

3. Language requirements:

* GUI is available only in English.

4. Login and password is used for the identification of users.

5. Limited to HTTP/HTTPS.

6. This system is working for single server.

**4.8 Other Requirements**

The fire fighting department officials as well as the users must agree with all the terms and conditions provided by the system administrator and local authority. They should obey all International Standards and Protocols.

# 5. DESIGN

**5.1 Introduction**

The design of the project includes architecture design, pin diagrams, connection modules, class diagram, sequence diagram etc.

**5.2 Architecture Design**

The architectural design is the design of the entire software system; it gives a high-level overview of the software system, it provides information on the decomposition of the system into modules (classes), dependencies between modules, hierarchy and partitioning of the software modules.

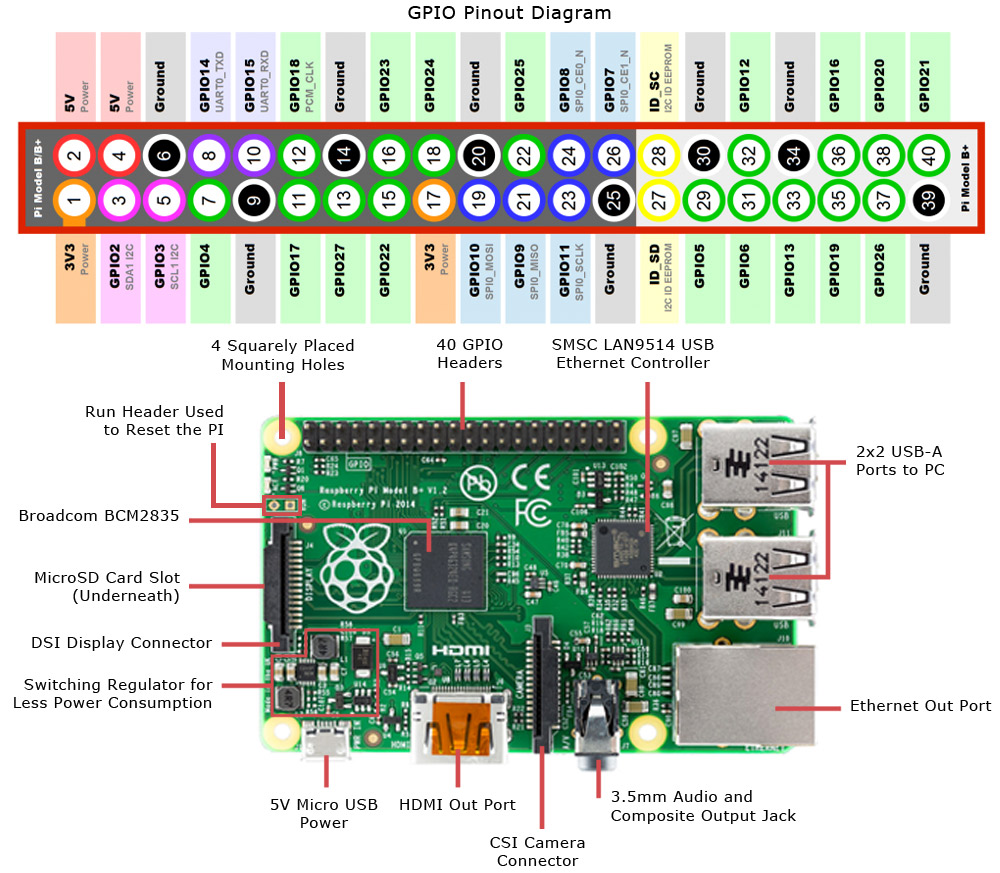


Figure 6 :Architecture of Raspberry Pi 2

**MQ2 SENSOR**

****

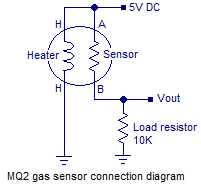
****

Figure 7 :MQ2 Sensor Connection Diagram Figure 8 :MQ2 Gas Sensor

**PIN CONNECTIONS FOR DS18B20**

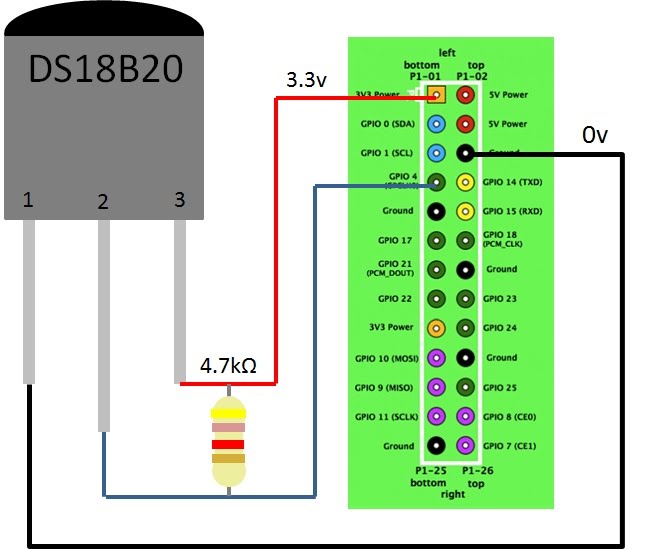
****

Figure 9 : Pin Connections for temperature sensor DS18B20

**5.3 Graphical User Interface**

A GUI uses a combination of technologies and devices to provide a platform that the user can interact with, for the tasks of gathering and producing information.

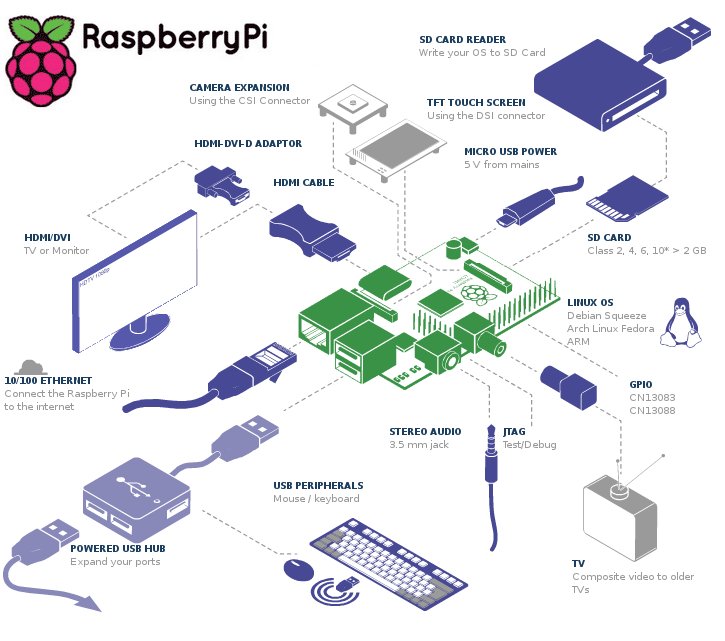


Figure 10 :Connection Module of Raspberry Pi 2

**Unity Graphical User Interface -**

Unity is a graphical shell for the GNOME desktop environment developed by Canonical Ltd. for its Ubuntu operating system. Unity debuted in the netbook edition of Ubuntu 10.10. It was initially designed to make more efficient use of space given the limited screen size ofnetbooks, including, for example, a vertical application switcher called the *launcher*, and a space-saving horizontal multipurpose *top menu bar*.The Unity user interface consists of several components:

* **Top menu bar** – a multipurpose top bar, saving space, and containing: (1) the menubar of the currently active application, (2) the capture bar of the main window of the currently active application including the maximize, minimize and exit buttons, (3) the session menuincluding the global system settings, logout, shut down and similar basic controls, and (4) the diverse global notification indicators including the time, weather, and the state of the underlying system.
* **Launcher** – a dock that also serves as a window switcher. Multiple instances of an application are grouped under the same dock icon, with a number of indicators to the side of the icon showing how many instances are open. The user has a choice whether or not to lock an application to the launcher. If it is not locked, an application may be started using the Dash or via a separately installed menu.
* **Quicklist** – the accessible menu of launcher items.
* **Dash** – an overlay that allows the user to search quickly for information both locally (installed applications, recent files, bookmarks, etc.) and remotely (Twitter, Google Docs, etc.) and displays previews of results. The Dash search feature was the subject of the privacy controversy.
* **Indicators** – a notification area (similar to an OS X *menu extra*), containing displays for the clock, network and battery status, sound volume etc.



Figure 11 :Unity Graphical User Interface

**5.4 Class Diagram and Classes**

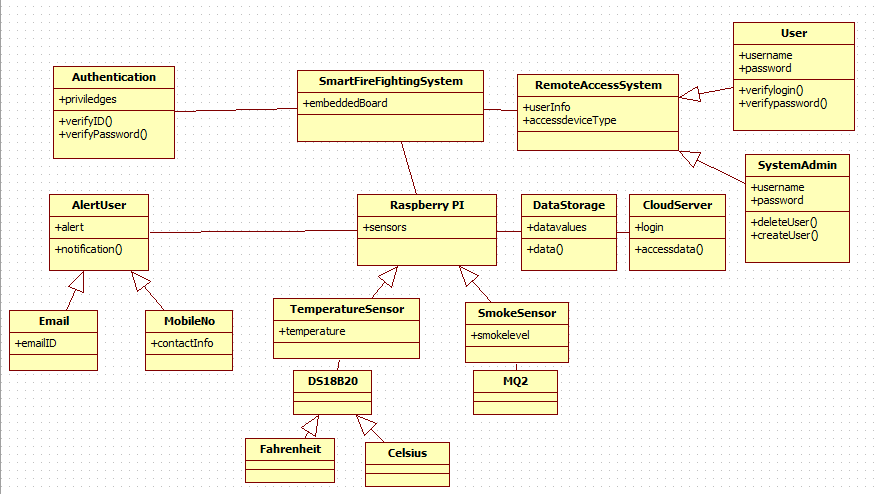


Figure 12 :Smart Fire Fighting Class Diagram

The various classes used are as follows -

* SmartFireFightingSystem -

It is the basic system used to detect and analyze data collected from sensors.

* Authentication -

Only authorized personnel can access the data.

* Remote Access System -

It can be mobile or computer.

* User -

User details are stored.

* System Admin -

System admin has administrator priviledges. He can add or delete users.

* Raspberry PI -

The **Raspberry Pi** is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.

* DataStorage -

Data analyzed from the sensors is stored.

* CloudServer -

Data is sent to Ubidots cloud server.

* TemperatureSensor -

It is used to measure temperature variations in surroundings.

* Fahrenheit -

Temperature is measured in Fahrenheit.

* Celsius -

Temperature is measured in Celsius.

* Alert User -

The user is alerted in case an emergency occurs.

* SmokeSensor -

It is used to measure smoke and carbon dioxide variations in surroundings.

* Email -

The alert notification is sent to the email id of the person.

* MobileNo -

The alert notification is sent to the email id of the person.

**5.5 Sequence Diagram**

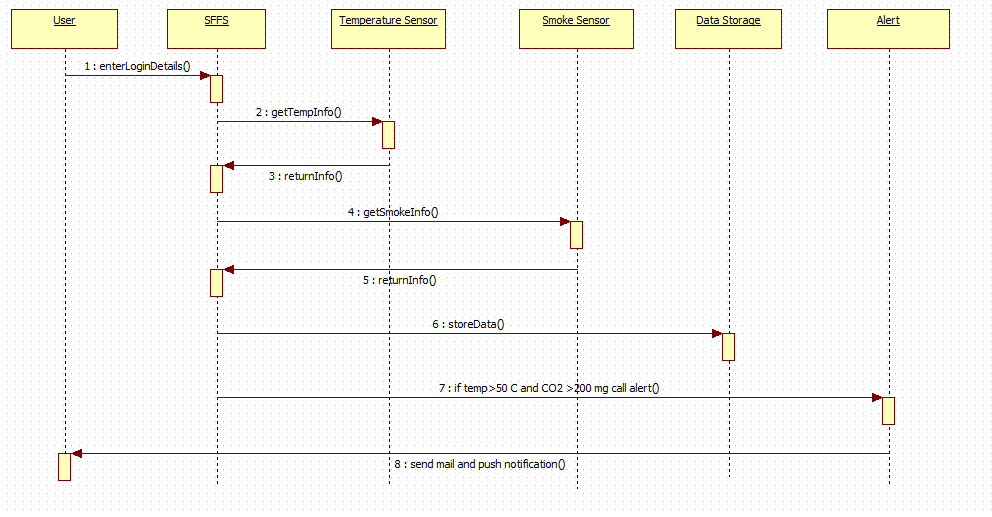


Figure 13 :Smart Fire Fighting Sequence Diagram

**5.6 Data flow diagram**

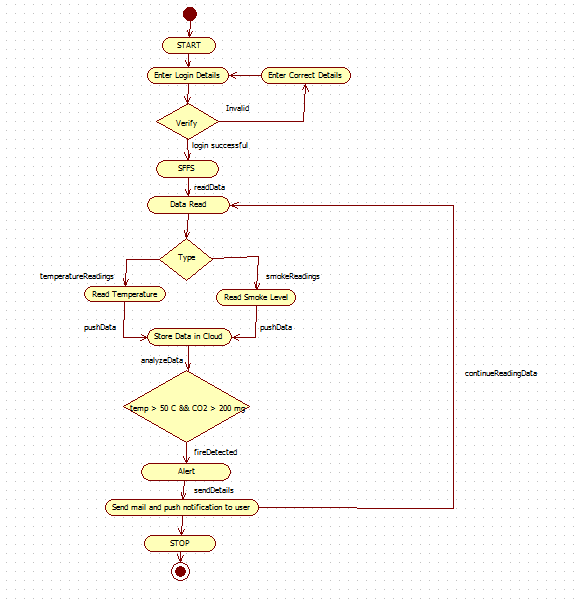


Figure 14 :Smart Fire Fighting Activity Diagram

# 6. IMPLEMENTATION

**6.1 Tools Introduction**

**6.1.1 PUTTY**

**PuTTY** is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port(since version 0.59). The name "PuTTY" has no definitive meaning.

PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and Mac OS X, and unofficial ports have been contributed to platforms such as Symbian, Windows Mobile and Windows Phone.

PuTTY was written and is maintained primarily by Simon Tatham and is currently beta software.

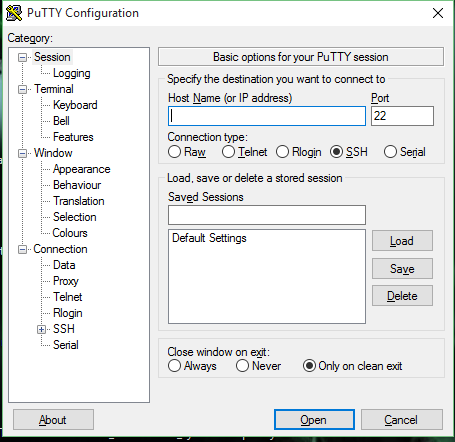


Figure 15 :PUTTY Configuration

**Features :**

PuTTY supports many variations on the secure remote terminal, and provides user control over the SSH encryption key and protocol version, alternate ciphers such as 3DES, Arcfour, Blowfish, DES, and Public-key authentication. It also can emulate control sequences from xterm, VT102 or ECMA-48 terminal emulation, and allows local, remote, or dynamic port forwardingwith SSH (including X11 forwarding). The network communication layer supports IPv6, and the SSH protocol supports the zlib@openssh.com delayed compression scheme. It can also be used with local serial port connections.

PuTTY comes bundled with command-line SCP and SFTP clients, called "pscp" and "psftp" respectively.

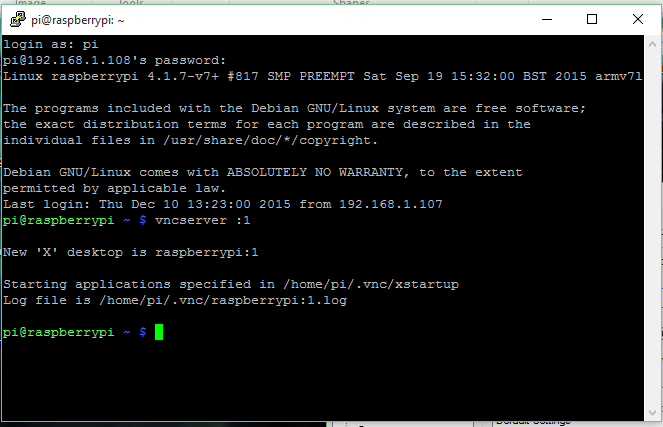


Figure 16 :PUTTY Terminal

**6.1.2 VNC Server :**

**TightVNC** is a cross-platform free and open-source remote desktop software application that uses and extends the RFB protocol ofVirtual Network Computing (VNC) to control another computer's screen remotely. It was created by Constantin Kaplinsky.

**Encodings**

TightVNC uses so-called "tight encoding" of areas, which improves performance over low bandwidth connection. It is effectively a combination of the JPEG and zlib compression mechanisms.[3][4] It is possible to watch videos and play DirectX games through TightVNC over a broadband connection, albeit at a low frame rate.[*citation needed*]

TightVNC includes many other common features of VNC derivatives, such as file transfer capability.

**Compatibility**

TightVNC is cross-compatible with other client and server implementations of VNC; however, tight encoding is not supported by most other implementations, so it is necessary to use TightVNC at both ends to gain the full advantage of its enhancements.[5]

Among notable enhancements are file transfers, support for Windows DFMirage mirror driver to detect screen updates (saves CPU time and increases the performance of TightVNC), ability to zoom the picture and automatic SSH tunneling on Unix.

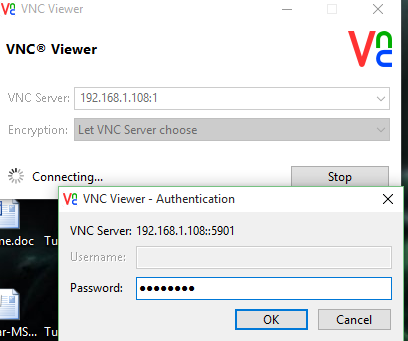


Figure 17 :VNC Viewer Authentication

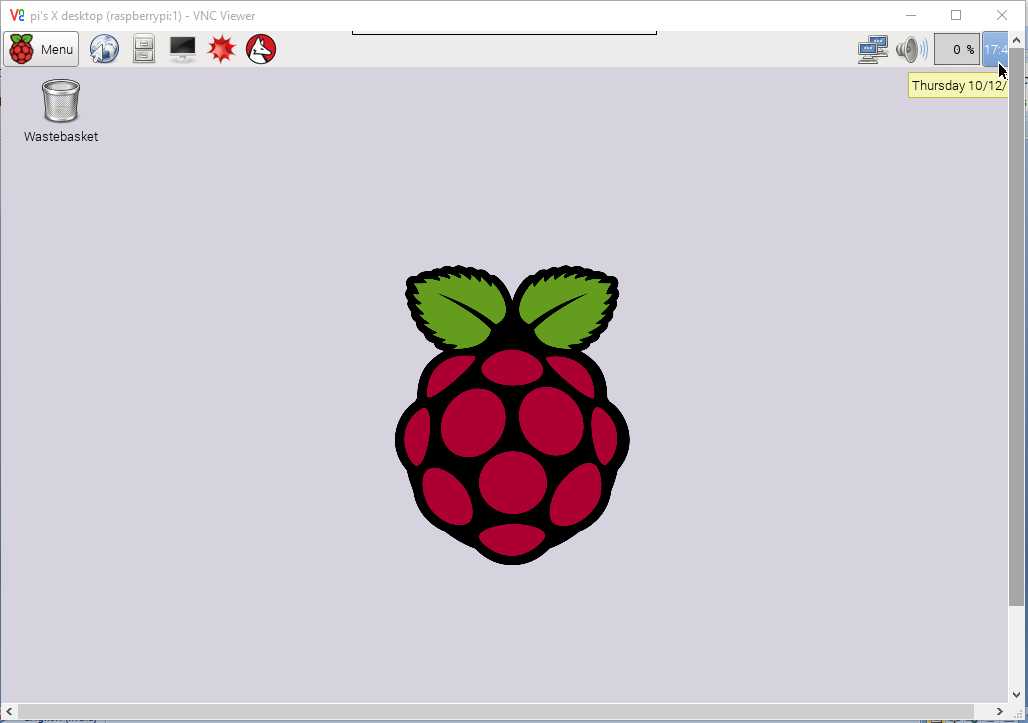


Figure 18 :VNC Viewer GUI

**Terminal of VNCSERVER**

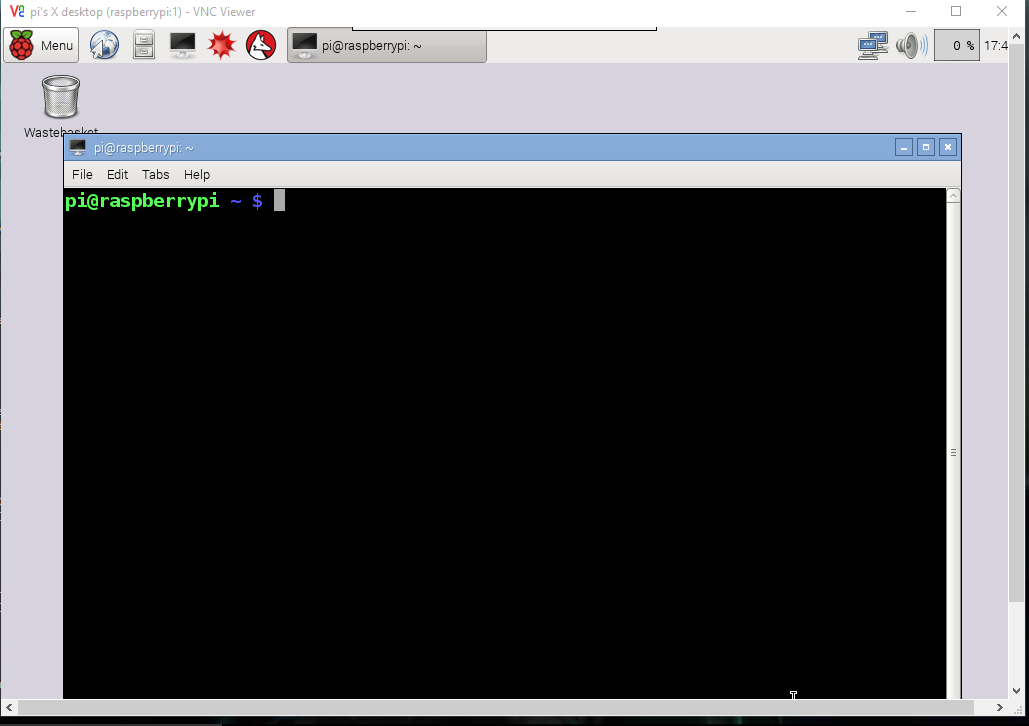


Figure 19 :VNC Viewer Terminal

**6.2 Technology Introduction**

**6.2.1 Python**

**Python** is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems. Using third-party tools, such as Py2exe or Pyinstaller,Python code can be packaged into stand-alone executable programs for some of the most popular operating systems, allowing the distribution of Python-based software for use on those environments without requiring the installation of a Python interpreter.

CPython, the reference implementation of Python, is free and open-source software and has a community-based development model, as do nearly all of its alternative implementations. CPython is managed by the non-profit Python Software Foundation.

Python has a big list of good features, few are listed below:

* IT supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* IT supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**Python code is used in this project to read the sensors data through raspberry pi 2 board.**

**Python code is also used to send data to the cloud.**

**6.2.2 UBIDOTS CLOUD SERVICE**

Cloud Service is used in this system. Ubidots is a free Internet of Things website which provides a free storage of data on cloud and allows to analyze those data. Analyzed data is stored and if some limit is crossed in data which are set by us, an event is generated which sends an e-mail to the respective person.

Ubidots helps you create applications that capture real-world data and turn it into meaningful actions and insights.

**Data Source**

A data source is your connected device. Every data source can have one or more variables, each one containing values in a time series. For example, a “Smart Fire Fighting” would be a data source, while its variables would be smoke and temperature.

**Value**

The measurement of the variable at a given point in time. For example, “the value of the room’s temperature was 70°F at 15:02:33 EDT”

**Event**

Events are “IF ... THEN ...” statements that are triggered depending of the last value of a variable. For example, you can set your sprinkler to send you an SMS if the temperature level is above a given value.

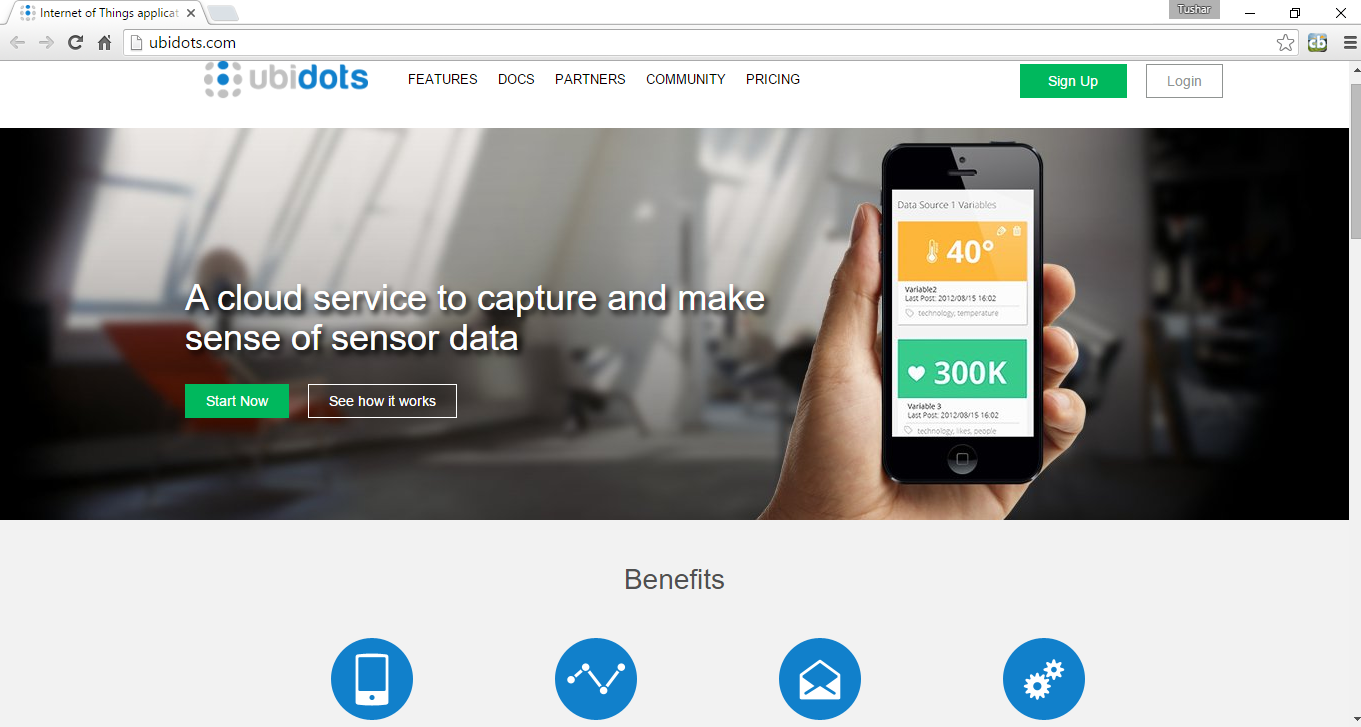


Figure 20 :Ubidots Web Page

**6.3 Overall view of the project in terms of implementation**

Automatic fire detection systems, when combined with other elements of an emergency response and evacuation plan, can significantly reduce property damage, personal injuries, and loss of life from fire in the workplace. Their main function is to quickly identify a developing fire and alert building occupants and emergency response personnel before extensive damage occurs. Automatic fire detection systems do this by using electronic sensors to detect the smoke, heat, or flames from a fire and providing an early warning.

**Installation and Restoration**

Assure that all devices and equipment constructed and installed to comply with this standard are approved for the purpose for which they are intended.

**Detector Selection:**

Fire detectors should be selected based on the burning characteristics of the materials present and the nature of location they will be used to protect.

**Smoke Detectors:**

Smoke detectors are designed to identify a fire during its smouldering or early flame stages and will meet the needs of most areas containing primarily wood, paper, fabric, and plastic materials. During combustion, these materials produce a mixture of smoke types with detectable levels of both large and small smoke particles. Smoke detectors are suitable for:

* Indoor areas with low ceilings such as offices, closets, and restrooms.
* Areas that are relatively clean with minimal amounts of dust and dirt.
* Areas that contain solid fuels like wood, paper, fabric, and plastic materials.

*NOTE*: Some locations are unsuitable for smoke detectors due to the potential for unwanted alarms -- work areas, kitchens, stairs, shafts, high air flow locations, areas that are dusty or dirty, as well as outdoor areas.

**Heat Detectors:**

Heat detectors are ideal for areas where flammable gasses and liquids are handled or any area where a fire will quickly cause a large change in the surrounding temperature. Heat detectors are also suitable for:

* Dirty, dusty or smoky environments.
* Indoor areas without winds or drafts that can prevent heat from reaching the detector.
* Manufacturing areas where large quantities of vapours, gases, or fumes may be present.
* Areas where particles of combustion are normally present, such as in kitchens, furnace rooms, utility rooms, and garages or where ovens, burners or vehicle exhaust gases are present.

**General Guidelines for Placing Fire Detectors:**

* Put at least one detector in each room, storage area, and hallway. You may need more than one detector per room for those that exceed the manufacturer's spacing requirements. For example, if your detector is rated for 30 feet, install detectors so they are evenly spaced with no more then 30 feet between detectors.
* Place the detector as close to the center of the ceiling as possible when only one detector is required in a room or space.
* Put at least one detector in each closet, elevator and dumbwaiter shaft, stairwell, and other enclosed spaces.
* Place a detector at the top of each flight of stairs.
* Place all smoke detectors at least three feet from ceiling fans.

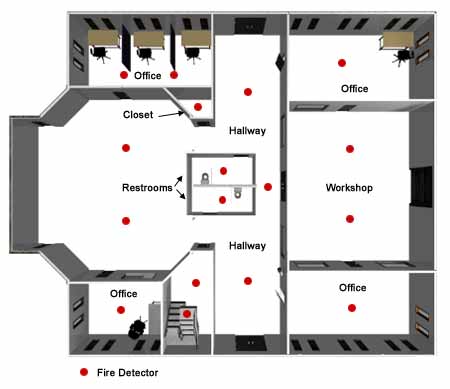


Figure 21 : Placing the Smart Fire Fighting product (Sample Layout)

**6.4 Explanation of Algorithm and how it is been implemented**

readSensorData(); // reads the data from smoke and temperature sensor

if smokeFound() or temperature > someFixedValue

onBuzzer();

triggerEvent(); // send the mail to the respective user

uploadDataOnCloud();

displaySensorData();

readSensorData():-

This function reads the raw data from the two sensors which we have used i.e smoke sensor(MQ2) and temperature Sensor(DS18B20). The smoke value is read as digital value which if is high then smoke is detected. The temperature sensor is a one wired digital temperature sensor which senses the room temperature in the range of -55C to +125C with an error of +-0.5C.

onBuzzer():-

Here if the value returned from the readSensorData() function if returned as true then the high value is passed to the vcc and the buzzer is switched on.We also have a code snippet here which tracks when the value becomes low and then it turns the buzzr Off.

triggerEvent():-

The values of the readSensorData() is used to trigger events such as sending an message, sending an email or setting the value of any variable.

updateDataOnCloud():-

The data which is being read from the sensors is uploaded to the cloud using this function for further analysis and event triggering process.

displaySensorData():-

The data which is uploaded on the cloud can be viewed using this function.

**6.5 Information about the implementation of Modules**

# 6.5.1 Number of modules

# The modules used in the project are -

# Client Module - Raspberry Pi and sensors

# Server Module - Ubidots cloud server

# 6.5.2 Module Description

# a) Client Module - Raspberry Pi

# Raspberry_Pi_B+_top.jpg

Figure 22 : Raspberry PI Module

# The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.

# r1.png

# r2.png

# r3.png

Figure 23 : Raspberry PI Specifications

# Operating System Used -

# Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware. An operating system is the set of basic programs and utilities that make your Raspberry Pi run. However, Raspbian provides more than a pure OS: it comes with over 35,000 packages, pre-compiled software bundled in a nice format for easy installation on your Raspberry Pi.

# Benefits:

# Get personal with your Pi. Welcome to the Raspberry Pi™ 2, the single board computer that thinks it’s a home PC.

# Enhanced Quad Core Processor and 1GB RAM now provides you with the opportunity to:

# Build your own workstation – create and manage your documents and spreadsheets with ease using LibreOffice™

# Faster and more enjoyable gaming – experience less lag and more seamless gaming in all your favourites. Can’t wait to demolish buildings in Minecraft™, trigger your TNT now and be amazed by the results!

# No more buffer face – boots up Raspbian in less than half the time as the Model B+

# More power for your favourite projects – create Space Progammes, Time Lapse Videos, GPS tracking, HD audio and lots more with a full ecosystem of supporting accessories.

# All your previous Raspberry Pi projects are 100% backward compatible with the new Raspberry Pi 2 Model B 1GB

# Technical Specifications:

# Broadcom BCM2836 Arm7 Quad Core Processor powered Single Board Computer running at 900MHz

# 1GB RAM

# 40pin extended GPIO

# 4 x USB 2 ports

# 4 pole Stereo output and Composite video port

# Full size HDMI

# CSI camera port for connecting the Raspberry Pi camera

# DSI display port for connecting the Raspberry Pi touch screen display

# Micro SD port for loading your operating system and storing data

# Micro USB power source

# 

# Raspberry Pi 2 Model B Features:

# Broadcom BCM2836 Arm7 Quad Core Processor powered Single Board Computer running at 900MHz

# 1GB RAM so you can now run bigger and more powerful applications

# Identical board layout and footprint as the Model B+, so all cases and 3rd party add-on boards designed for the Model B+ will be fully compatible.

# Fully HAT compatible

# 40pin extended GPIO to enhance your “real world” projects. GPIO is 100% compatible with the Model B+ and A+ boards. First 26 pins are identical to the Model A and Model B boards to provide full backward compatibility across all boards.

# Connect a Raspberry Pi camera and touch screen display (each sold separately)

# Stream and watch Hi-definition video output at 1080P

# Micro SD slot for storing information and loading your operating systems.

# Advanced power management:

# You can now provide up to 1.2 AMP to the USB port – enabling you to connect more power hungry USB devices directly to the Raspberry PI. (This feature requires a 2Amp micro USB Power Supply)

# 10/100 Ethernet Port to quickly connect the Raspberry Pi to the Internet

# Combined 4-pole jack for connecting your stereo audio out and composite video out

# b) Client Module - Temperature and Smoke Sensors

# Temperature Sensor - DS18B20

# Hardware Overview –

# The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with non volatile user programmable upper and lower trigger points.

# The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor.

# It has an operating temperature range of -55°C to +125°C and is accurate to ±0.5°C over the range of -10°C to +85°C.

# Features :

# Unique 1-Wire® Interface Requires Only One Port Pin for Communication

# Each Device has a Unique 64-Bit Serial Code Stored in an On-Board ROM

# Multi drop Capability Simplifies Distributed Temperature-Sensing Applications

# Requires No External Components

# Can Be Powered from Data Line; Power Supply Range is 3.0V to 5.5V

# Measures Temperatures from -55°C to +125°C (-67°F to +257°F)

# ±0.5°C Accuracy from -10°C to +85°C

# Thermometer Resolution is User Selectable from 9 to 12 Bits

# Converts Temperature to 12-Bit Digital Word in 750ms (Max)

# 2812.png

Figure 24 : DS18B20 Diagram

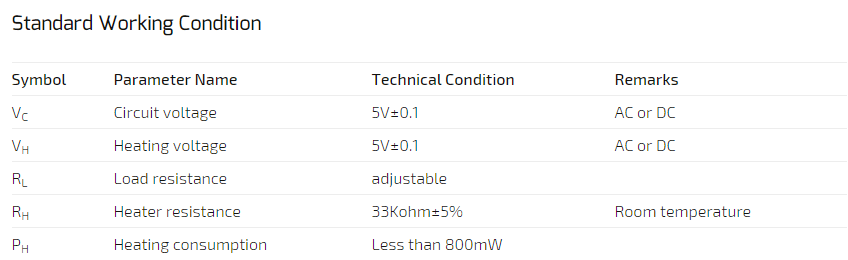
# SMOKE SENSOR - MQ2

# Hardware Overview –

# This is a Analog output sensor. This needs to be connected to any one Analog socket in Base Shield. The examples used in this tutorial makes uses of A0 analog pin. Connect this module to the A0 port of Base Shield.

# It is possible to connect the Grove module to Arduino directly by using jumper wires by using the connection as shown in the table

# The output voltage from the sensor increases when the concentration of gas increases. Sensitivity can be adjusted by varying the potentiometer.



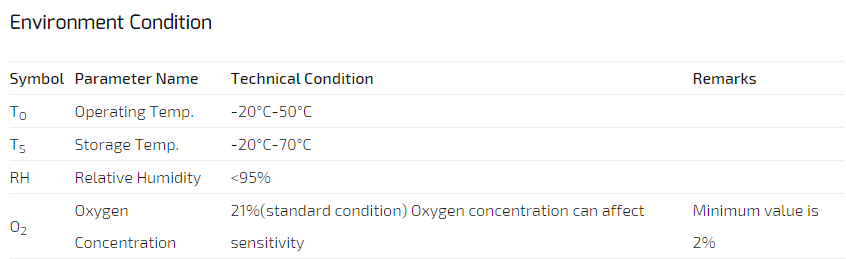


Figure 25 : MQ2 Specifications

* **Applications :** 
  + - * They are used in gas leakage detecting equipments in family and industry, are suitable for detecting of LPG, i-butane, propane, methane ,alcohol, Hydrogen, smoke.

**c) Cloud Module -**

**6.5.3 UBIDOTS Cloud Server -**

Ubidots helps you create applications that capture real-world data and turn it into meaningful actions and insights.

# 7. TESTING

**Setup Raspberry Pi to send data to Ubidots using python library -**

1. Download the Raspberry Pi operating system, we recommend Raspbian Wheezy.
2. Unzip the file and you’ll end up with a large IMG file: DON’T COPY AND PASTE THIS FILE INTO YOUR SD CARD, because it won’t work. You’ll need to write it properly to the SD card by following these steps according to your operating system:
   * MacOS: http://ivanx.com/raspberrypi/
   * Linux: http://elinux.org/RPi\_Easy\_SD\_Card\_Setup#Using\_Linux\_.28including\_on\_a\_Pi.21.29
   * Windows: http://elinux.org/RPi\_Easy\_SD\_Card\_Setup#Using\_Windows\_7\_or\_Windows\_XP
3. Plug a keyboard and a monitor through the HDMI or TV/Analog ports.

**Configure your Raspberry Pi to use our Python API Client -**

1. Let’s make sure your device is up to date so that it has the latest python tools (be aware that this will take a while):

$ sudo apt-get update

$ sudo apt-get upgrade

1. Download the pip installer and install Ubidots’ Python library

$ sudo apt-get install python-setuptools

$ sudo easy\_install pip

$ sudo pip install ubidots

**Setup a test Variable in Ubidots**

1. As a logged in user navigate to the “Data” tab.
2. Create a Data Source by clicking on the orange icon on the right. Then create a variable within that Data Source.
3. Take note of the variable’s ID to which you want to send data. For this example we’ll use a variable with the ID: “521d792df91b2816f35c8587”
4. Take note of your API key.

**Send data to Ubidots -**

Coming back to your Raspberry Pi:

1. Create a directory called “ubidots” where you can put this and future scripts:

$ mkdir ubidots

1. Create a python script using your favorite text-editor. We’ll use “nano” in this case:

$ cd ubidots

$ nano ubi-test.py

Put the following code into the created file. Please note the fields where you should put your API key and your variable ID.

from ubidots import ApiClient

import os

import glob

import time

import RPi.GPIO as GPIO

import time

api = ApiClient("181bfdacd408f5d2dcff6ebdb655b2a68387606e")

test\_variable1 = api.get\_variable("5669a22d7625422c9c7345aa")

test\_variable = api.get\_variable("56699131762542086a2f3480")

GPIO.setmode(GPIO.BOARD)

GPIO.setup(13,GPIO.OUT)

GPIO.setup(11,GPIO.IN)

os.system('moderate w1-therm')

base\_dir='/sys/bus/w1/devices/'

device\_folder=glob.glob(base\_dir + '28-0314566612ff')[0]

#g from 28\*

device\_file = device\_folder + '/w1\_slave'

def read\_temp\_raw():

        f=open(device\_file,'r')

        lines=f.readlines()

        f.close()

        return lines

def read\_temp():

        lines=read\_temp\_raw()

        while lines[0].strip()[-3:]!='YES':

                time.sleep(0.2)

                lines=read\_temp\_raw()

equals\_pos=lines[1].find('t=')

        if equals\_pos!=-1:

                temp\_string = lines[1][equals\_pos+2:]

                temp\_c=float(temp\_string)/1000.0

                temp=GPIO.input(11)

                if (temp\_c <50) and (temp==1):

                        GPIO.output(13,0)

                else :

                        GPIO.output(13,1)

                test\_variable.save\_value({'value':temp\_c})

                test\_variable1.save\_value({'value':temp})

                if GPIO.input(11):

                        print "Smoke not sensed"

                else:

                        print "Smoke sensed"

                temp\_f=temp\_c\*9.0/5.0 + 32.0

                return temp\_c,temp\_f

while True:

        print(read\_temp())

        time.sleep(1)

1. Run the code several times to send some random values to the cloud:
2. $ python ubi-test.py

The data points can now be seen in the browser.

**7.1 Results and Snapshots**

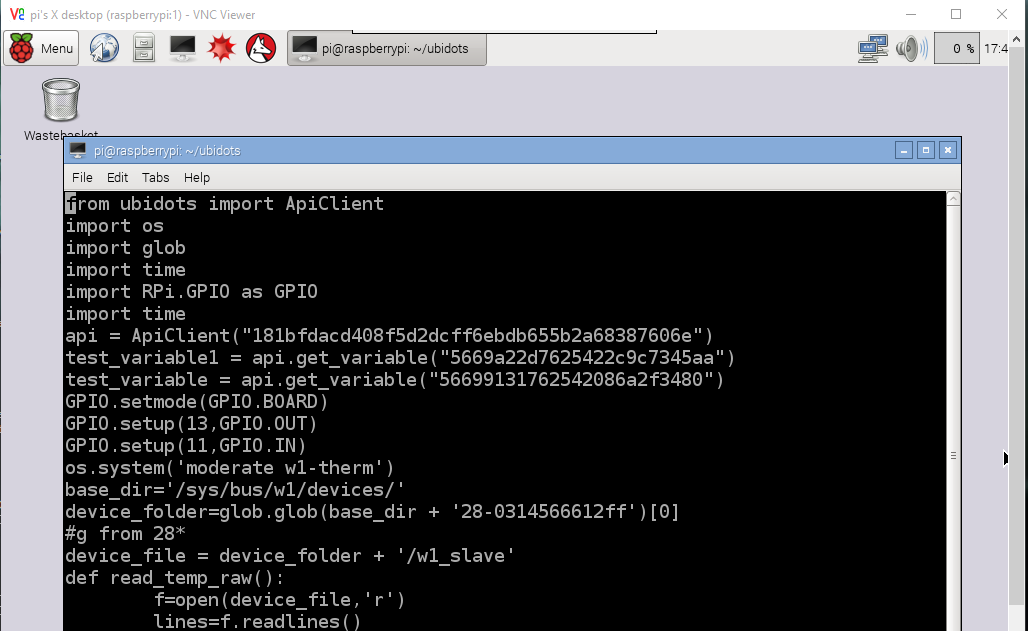


Figure 26 : Python Code

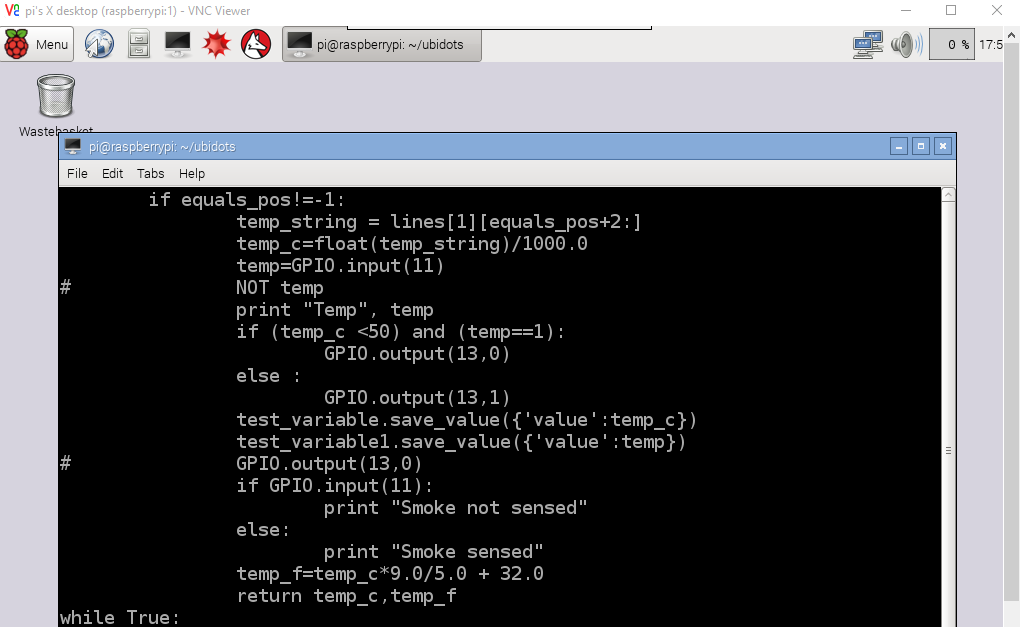


Figure 27 : Python Code(Continued)

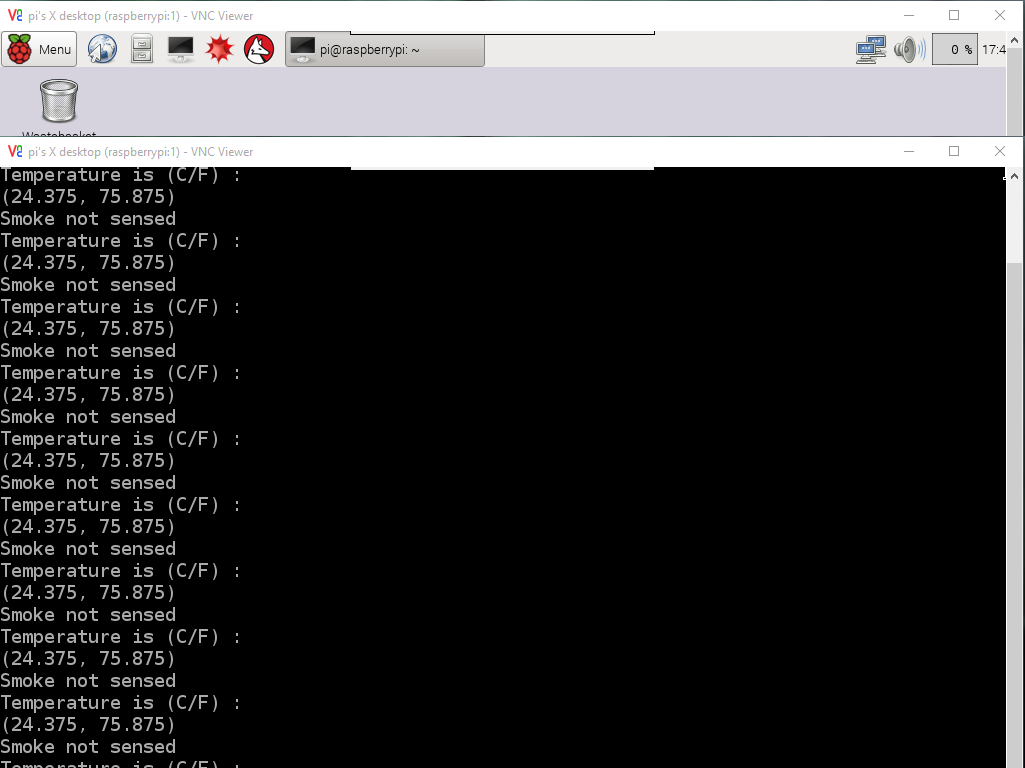


Figure 28 : Python Code Sample Output

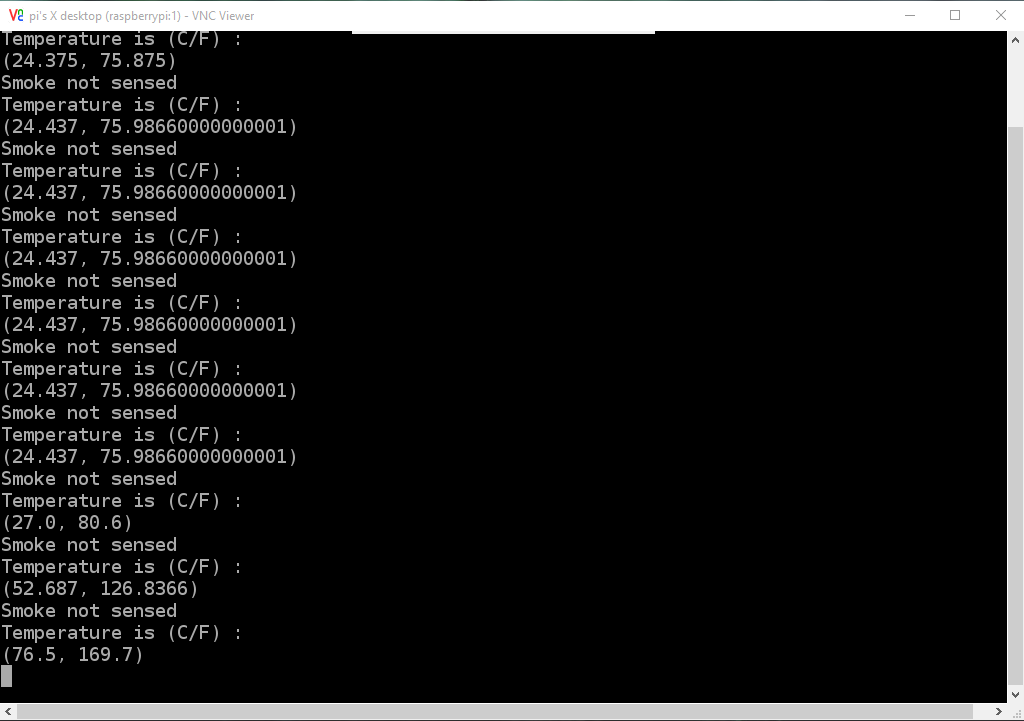


Figure 29 : Python Code Sample Output

**Sources created in the ubidots**

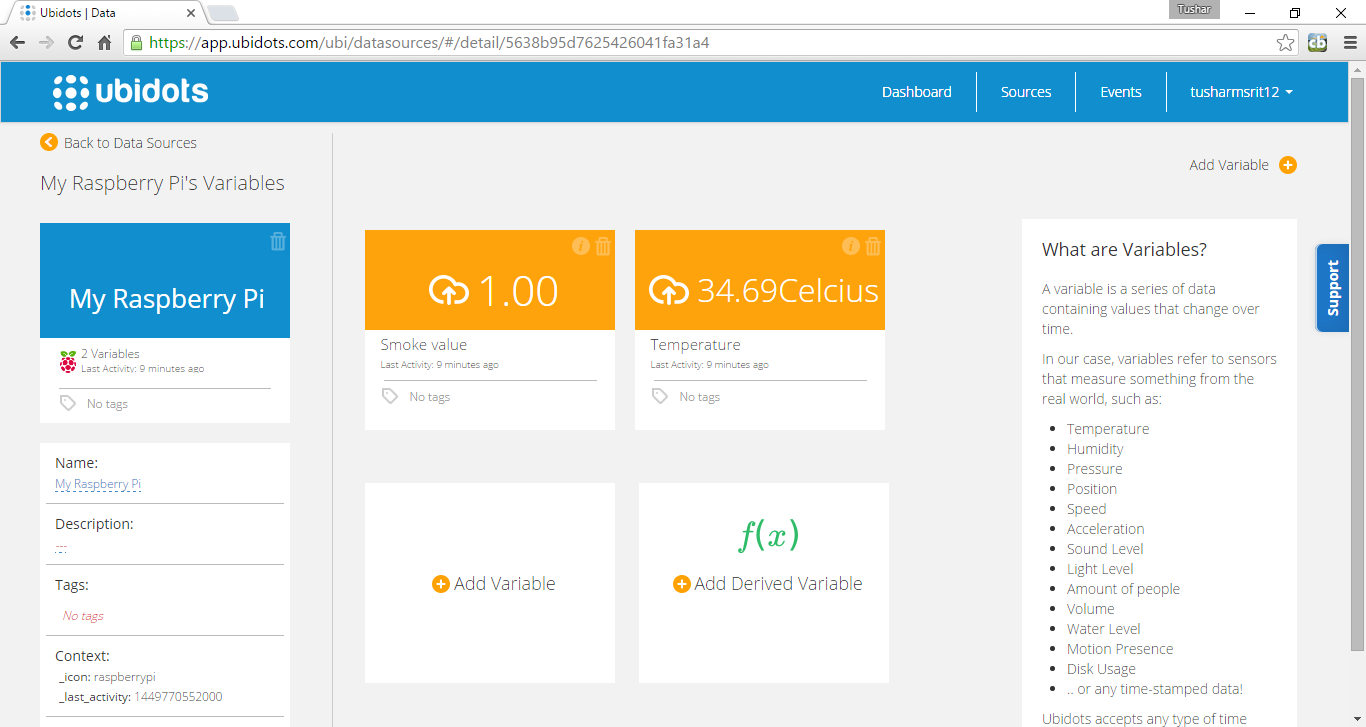


Figure 30 : Ubidots Source Creation

**Source Smoke**

This source contains the value of smoke. If the smoke value, which is a digital value, is 1 means smoke is detected else smoke is not detected.

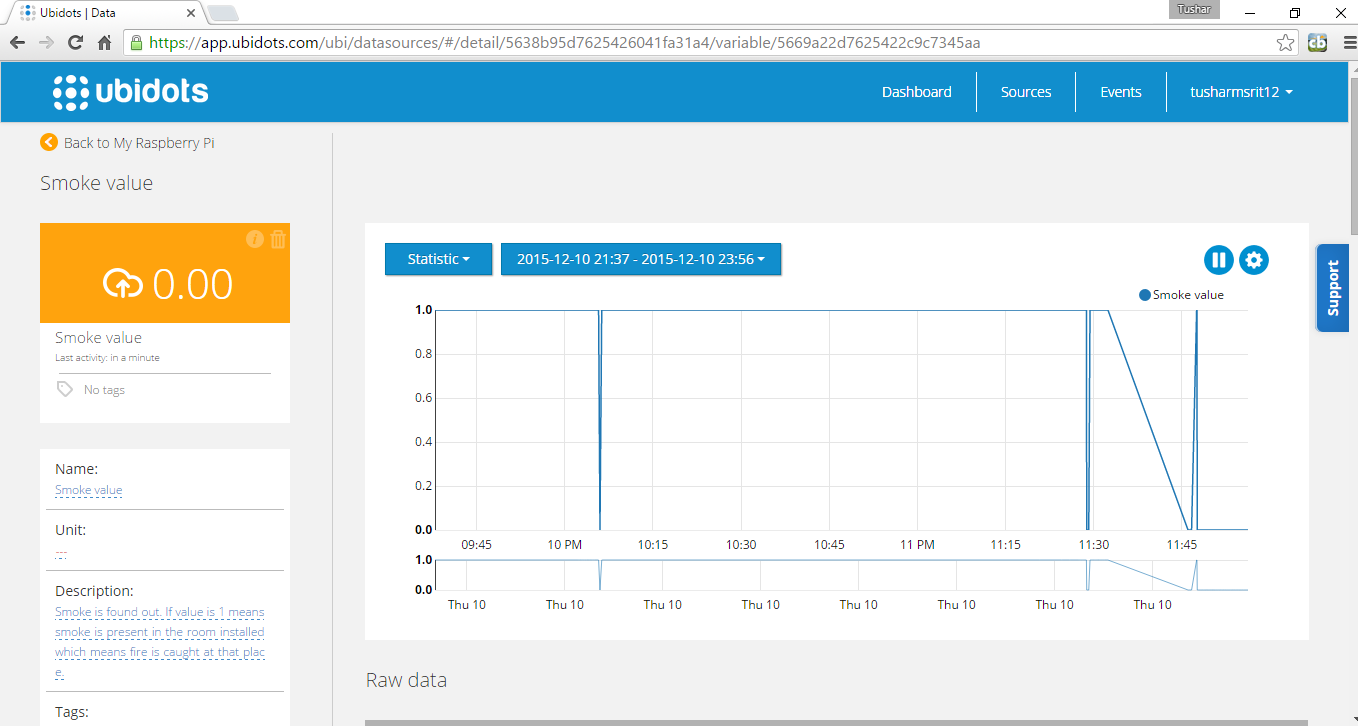
****

Figure 31 : Ubidots Smoke Source

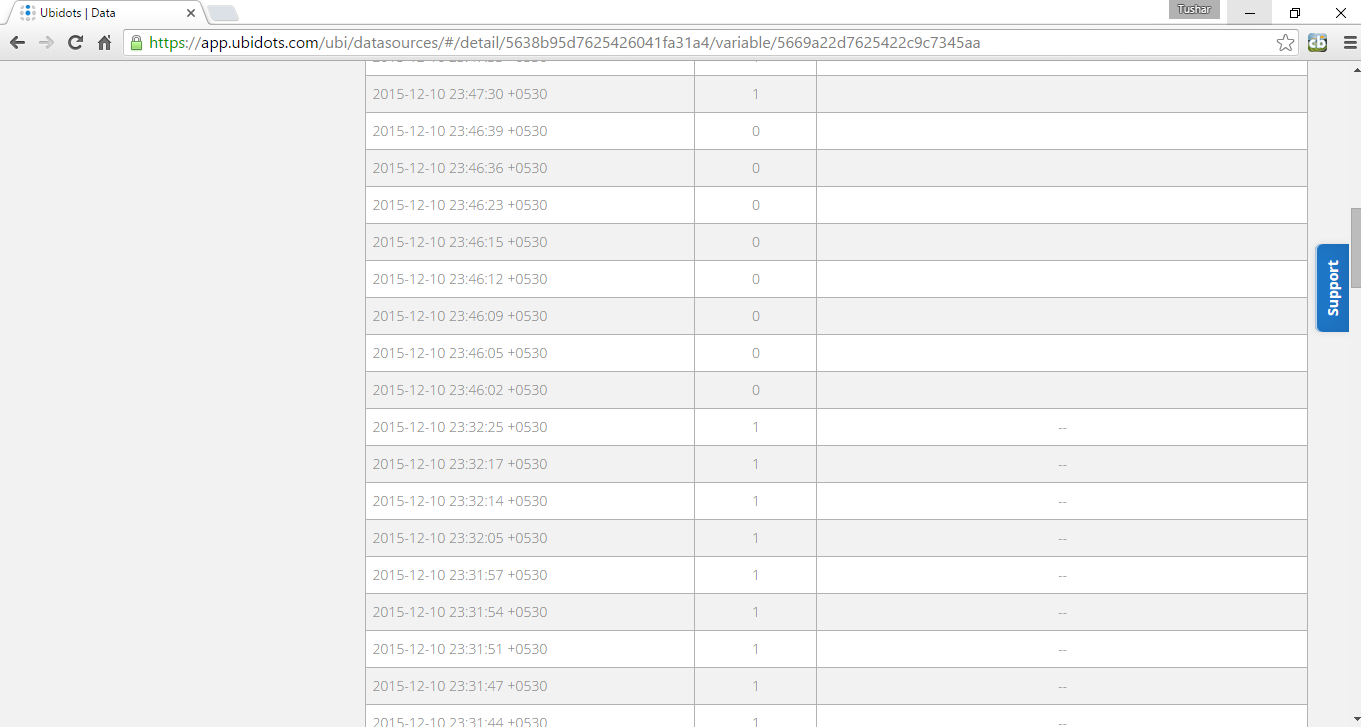
****

Figure 32 : Ubidots Smoke Source Data

**Temperature Source**

This shows the temperature source. It shows the different temperature sensed by the sensor per second. A graph is also present which shows the previous data of the temperature.

****

Figure 33 : Ubidots Temperature Source

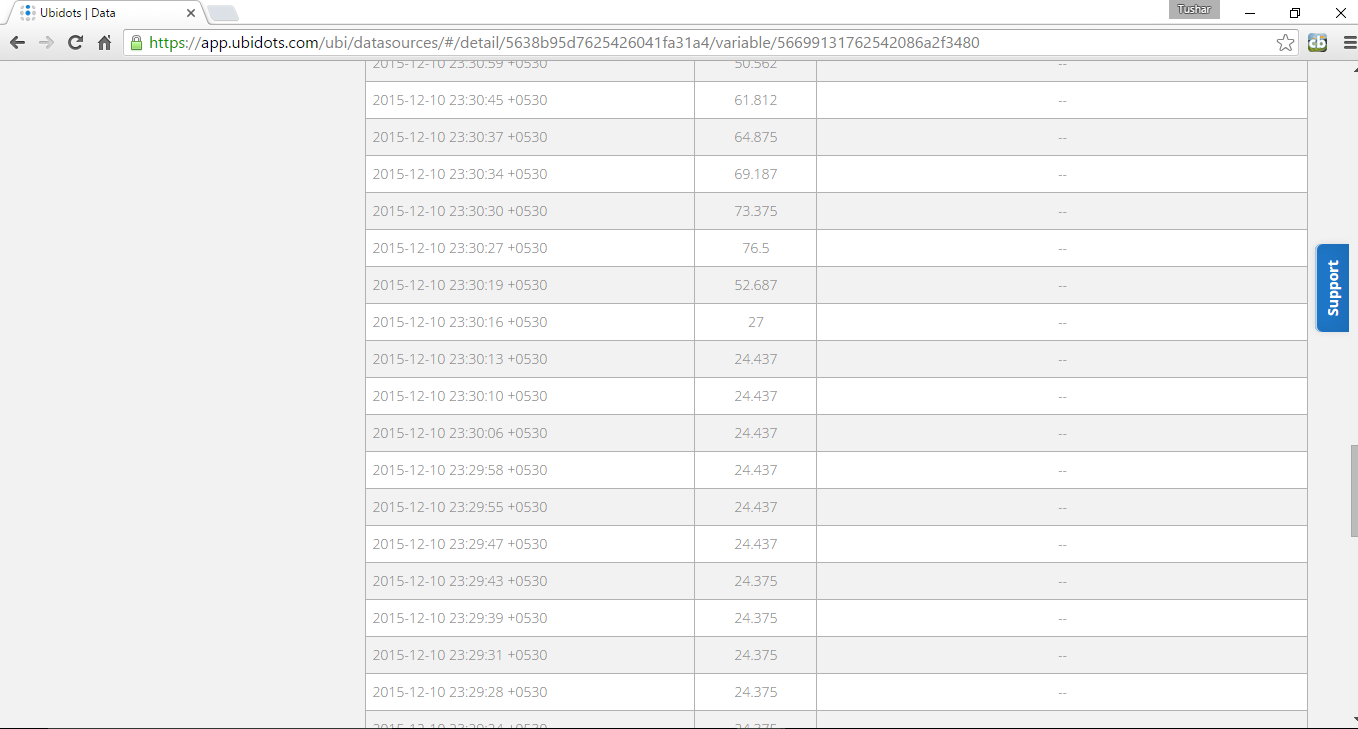
****

Figure 34 : Ubidots Temperature Source Data

**Events :**

Events are generated if smoke value is 1 or temperature is above 50. It can be changed accordingly.

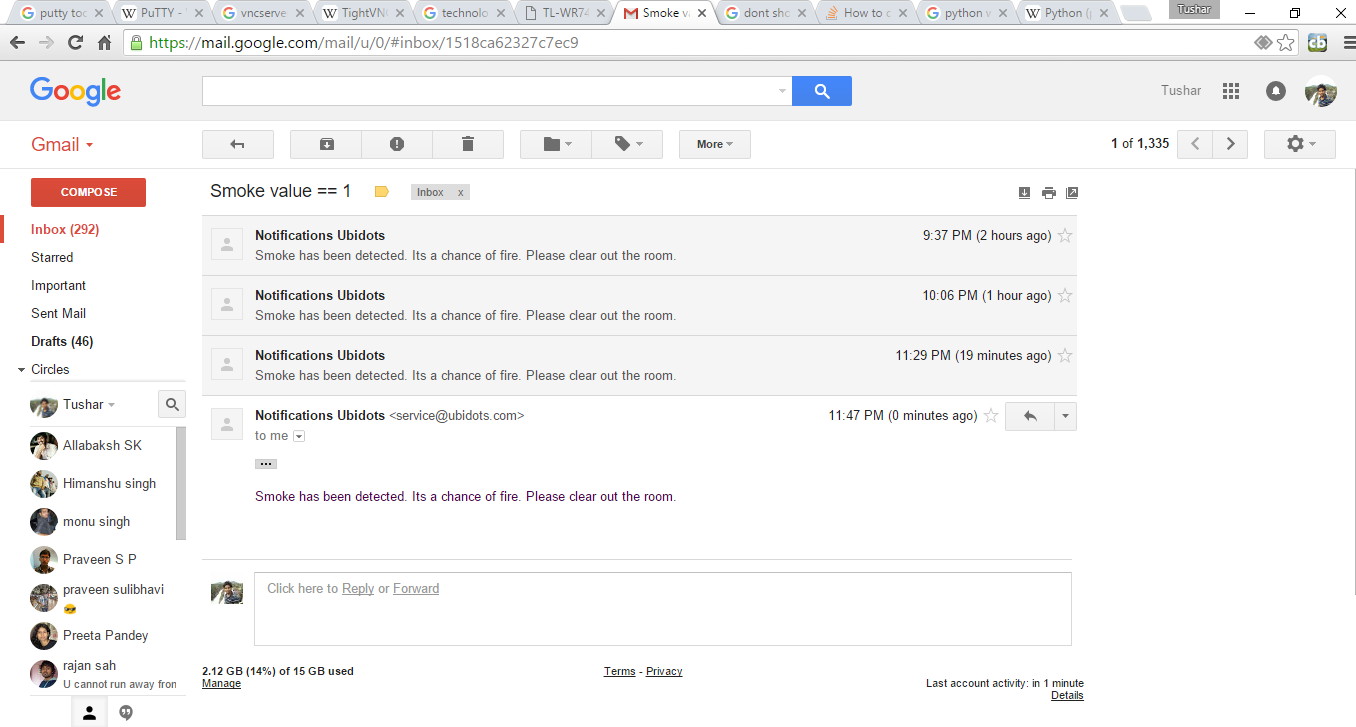
****

Figure 35 : Event Generation for Smoke Readings (GMail)

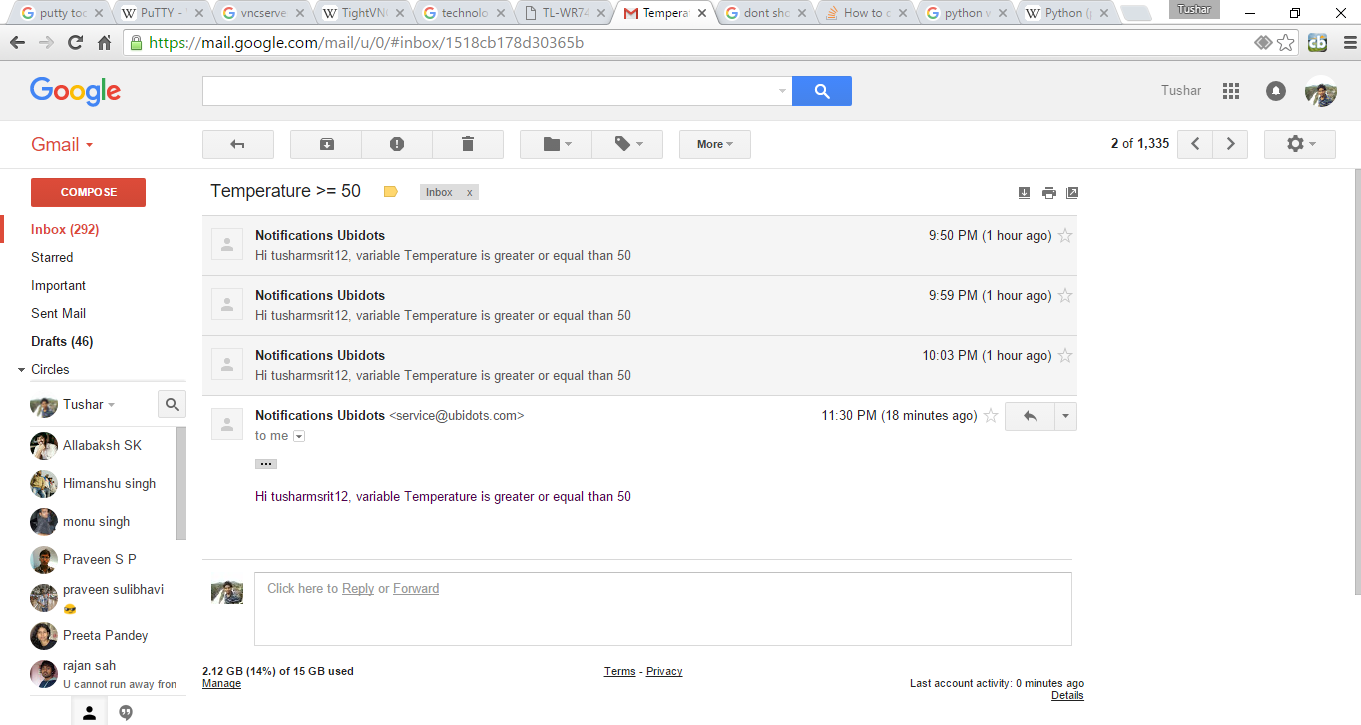
****

Figure 36 : Event Generation for Temperature Readings (GMail)

**7.2 Security Protocol for IOT**

The Internet of Things (IoT) is growing by leaps and bounds every day. But as the IoT grows, so do the security vulnerabilities of the linked objects. A security protocol to protect IoT devices will always be needed.

For example, an appliance manufacturer may want to link its air conditioning systems to smart home networks to increase sales, but it has never faced the problems of securely networking a product before. The IoT will never be secured until all its participants learn how to embrace security from the outset.

**Building In Security Measures for the IoT-**

There are some obvious things that can be done to embed security in the IoT.

* The most obvious one is securing the Web interface of the device. Simple things like making sure that default usernames and passwords are changed during initial setup helps greatly. And the changes shouldn’t allow the use of weak passwords. Perhaps measures such as an account lockout after three to five failed login attempts should be considered.
* Attention should be paid to passwords beyond just initial setup changes. Checking network traffic to ensure login codes are not being sent in cleartext is a wise move, and that goes for any password recovery schemes, as well. Additionally, two-factor authentication may be needed for sensitive areas like administrator accounts.
* Examining the Web interface for resistance to common attacks like cross-site scripting, cross-site request forgery and SQL injection should be done, too. These attacks are harmful if successful, but they’re also relatively simple in nature, which makes them preventable with the right proactive safeguards.
* One trick attackers use involves scanning for open ports with a special program and then exploiting those ports. Universal Plug and Play (UPnP) has only exacerbated this problem by standardizing network access points. Open ports can now be used to launch denial-of-service (DoS) attacks as well as buffer overflow attacks across networks and devices.

# 8. CONCLUSION & SCOPE FOR FUTURE WORK

The word Smart is appearing everywhere these days. Instances include Smart Machines, Smart Grid, Smart Manufacturing, Smart Cities, and Smart America, just to name a few. This trend is expected to increase.

Although some smart systems are already implemented, huge challenges remain in terms of engineering these products and systems to attain their full potential. A key area, ripe for development, is Smart Fire Fighting.

The purpose of this project is to provide an overview of the current state and future trends of Smart Fire Fighting. Today’s fire fighting and fire protection environment is data poor and without integrated analysis and decision making. Changing this situation will require new types of technologies.

In addition to identifying key research priorities, additional issues, including the following were addressed :

• Review of existing and available knowledge and technology

• What is needed in terms of technology or tools

• What knowledge (and/or technology) is needed

• Other pertinent information

The purpose of the project is to identify and prioritize the research and development needs for implementation of the next generation of smart systems to benefit fire protection and fire fighting. The idea of Smart Fire Fighting is based on creating, storing, exchanging, analyzing, and integrating information from a wide range of databases and sensor networks. There are many challenges that must be overcome to exploit the promise of smart technologies.

The vision of Smart Fire Fighting can be realized by harnessing the power of emerging information, communication, sensor, and simulation technologies to enable markedly better situational awareness, predictive models, and decision making.

# 9. REFERENCES

[1]https://www.nedcc.org/free-resources/preservation-leaflets/3.-emergency- management/3.2-an-introduction-to-fire-detection,-alarm,-and-automatic-fire-sprinklers

[2]. http://ubidots.com/

[3].http://www.govtech.com/fs/New-Era-in-Firefighting-Leverages-the-Internet-of-Things.html

[4].http://www.forbes.com/sites/federicoguerrini/2015/06/29/firefighters-and-the-internet-of-things/

[5].http://www.jbrehm.com/blog/2015/6/11/smart-fire-fighting-merging-iot-and-big-data-for-fire-safety

[6]. http://www.criticallink.com/2015/07/firefighting-and-the-iot/

[7]. http://www.nist.gov/el/fire\_research/201506\_smart\_fire\_roadmap.cfm

[8]. IEEE Journals and papers - http://ieeexplore.ieee.org/

[9]. http://blog.ubidots.com/topic/raspberry-pi