### 1. Introduction

The infantry soldier of tomorrow promises to be one of the most technologically advanced modern warfare has ever seen. Around the world, various research programs are currently being conducted, such as the United States Future Force Warrior (FFW) and the United Kingdom’s Future Infantry Soldier Technology (FIST), with the aim of creating fully integrated combat systems. Alongside vast improvements in protective and weaponry subsystems, another major aspect of this technology will be the ability to provide information superiority at the operational edge of military networks by equipping the dismounted soldier with advanced visual, voice, and data communications.



Fig 1.1- Picture of the module

Helmet mounted visors, capable of displaying maps and real -time video from other squad members, ranges of physiological sensors display the heartbeat, body temperature, atmosphere pressure, surrounding oxygen level etc. These devices will improve awareness for collateral military personnel as well as who will exchange information using wireless networks along with host.

The challenge was to integrate these piecemeal components into a lightweight package that could achieve the desired result without being too bulky and cumbersome or requiring too much power. Communicating with the base (control room) station become the fundamental challenges in military operations also the proper navigation between soldier’s organizations plays important role for careful planning and co-ordination.

The soldier Health and Position Tracking System allows military to track the current GPS position of soldier and also checks the health status including body temperature and heartbeats of soldier.

The System also consists extra feature with the help of that soldier can ask for help manually or send a distress signal to military if he is in need.  
The GPS modem sends the latitude and longitude position with link pattern with the help of thatmilitary can track the current position of the soldier.  
The system is very helpful for getting health status information of soldier and providing them instant help.

**2. Literature Survey**

**2.1 Health Monitoring and Tracking of Soldier Using GPS:-**

International Journal of Research in Advent Technology, Vol.2, No.4, April 2014 E-ISSN: 2321-9637 291

M.V.N.R. Pavan Kumar1 ,Ghadge Rasika Vijay2 ,Patil Vidya Adhikrao3 ,Bobade Sonali Vijaykumar4 Department of Electronics and Telecommunication Engineering 1,2,3,4, LNBCIET, Satara-415020 1,2,3,4.

In today’s world the security of the nation is depends up on the enemies’ warfare and so the safety of the soldiers is considered as vital role in it. Concerning the soldiers safety there are many instruments to view their health status as well as ammunitions on the soldiers. In soldiers security, bio-sensors systems gives different types of small physiological sensors, Biomedical sensor , transmission modules and processing capabilities, and can thus facilitate low-cost wearable unobtrusive solutions for health monitoring. GPS used to log the longitude and latitude so that direction can be known easily. These devices are being added to weapons, firearms, and militaries such as the Israeli an Army which are exploring the possibility of embedding GPS devices into soldiers vests and uniforms so that field commanders can track their soldier’s movements in real time.GSM module can be used for effective range of high-speed transmission, short-range and soldier-to-soldier wireless communications that will be required to relay information on situational awareness, tactical instructions, and covert surveillance related data during special operations reconnaissance and other missions .So by using these equipment’s we are trying to implement the basic lifeguarding system for soldier in low cost and high reliability.

**2.2 Real Time Soldier Tracking System:-**

IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834, p-ISSN: 2278-8735 PP 21-24 www.iosrjournals.org

A Conference on Wireless Communication and Android Apps "WiCAA–15" 21 | Page

K.V.N.Naik Institute of Engineering Education & Research (KVNNIEER), Nashik Pangavhane S. M.1 , Choudhary Sohanlal 2 & Pathak Bhavik 3 1,2,3(E&TC Engg. Dept.,S.I.E.R Agaskhind, SPP Univ., Pune(MS), India)

Army’s welfare is one of the most important aspects of any country. It is our duty to equip soldier with better advanced technology. This paper helps to track the soldier at any given moment using GPS. In this paper, soldier’s health parameters such as heart rate and body temperature are continuously measured and transmitted wirelessly to the control room using GSM. In case of death of the soldier, the processor detects the change in pulse rate and location of the dead soldier; trackedby the GPS module is then communicated to the military base station by the use of GSM. This information can be used to devise war strategies as to how many more soldiers (and where) should be deployed to replace the martyrs. In case of emergency, soldier can communicate directly to the base station using a panic switch.

**2.3 A Soldier Health Monitoring System for Military Applications:-**

With recent advances in technology, various wearable sensors have been developed for the monitoring of human physiological parameters. A Body Sensor Network (BSN) consisting of such physiological and biomedical sensor nodes placed on, near or within a human body can be used for real-time health monitoring. In this paper, we describe an on-going effort to develop a system consisting of interconnected BSNs for real-time health monitoring of soldiers. We discuss the background and an application scenario for this project. We describe the preliminary prototype of the system and present a blast source localization application.

Published in: 2010 International Conference on Body Sensor Networks

Date of Conference:7-9 June 2010

Page(s):246 - 249

ISSN :2376-8886

Print ISBN:978-1-4244-5817-2

INSPEC Accession Number:11483985

Conference Location :Singapore

DOI:10.1109/BSN.2010.58

Publisher:IEEE

**2.4 GPS Based Advanced Soldier Tracking With Emergency Messages & Communication System:-**

International Journal of Advance Research in Computer Science and Management Studies Research Article / Survey Paper / Case Study Available online at: www.ijarcsms.com Palve Pramod M.E (VLSI & Embedded System) G. H. Raisoni College of Engineering & Management, Chas, Ahmednagar – India

In today’s world enemy warfare is an important factor in any nation’s security. The national security mainly depends on army (ground), navy (sea), air-force (air).

The important and vital role is played by the army soldier’s. There are many concerns regarding the safety of these soldiers. As soon as any soldier enters the enemy lines it is very vital for the army base station to know the location as well as the health status of all soldiers. In our project, the soldier can ask for directions to the army base unit in case he feels that he is lost. By using the location sent by the GPS, the base station can guide soldier to safe area & GSM will help to communicate the Soldier unit with Base unit. By getting the exact location of soldiers it will help the Soldiers to discuss about their war strategies and take guidance from Base unit. The various Health Sensors such as Temperature sensor, Heart rate sensors, Humidity sensors, Gas detection sensors will help to decide the health status of that particular soldier.

3. Specifications

**3.1 SYSTEM SPECIFICATIONS:**

-Internet Connectivity: 802.11 b/g/n Wireless LAN Built In WiFi.

-Operating system: Boots from micro SD card, running a version of the Linux OS or Windows 10 IOT.

-Frequency: 1.2GHz

-Power: Micro USB socket 5V, 2.5A.

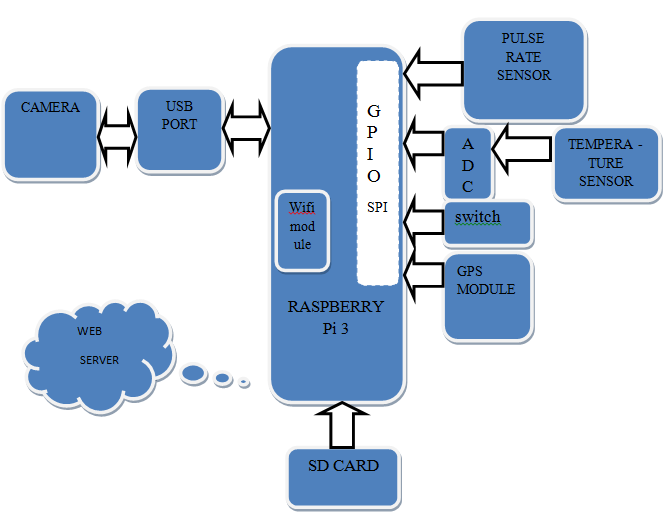
-Memory: 1GB LPDDR2

-USB: USB4 x USB 2.0 Connector

-GPS SR100: Built-in Patch antenna and MMCX RF connector for external antenna connector. Dual antenna support.

4. Methodology

**4.1 BLOCK DIAGRAM:**



*Fig 4.1.1. The block diagram*

**4.2 Working:**

The system mainly consists of RASPBERRY- PI 3 board, temperature sensor, Pulse Rate sensor, ADC, Camera, WIFI module, Emergency switch and importantly we are designing web server from which we can monitor all sensor parameters.

**Sensor application-**

If soldier gets injured and becomes unconscious by gunshot or due to any other reason, then his heart beats start increasing or decreasing gradually. In this type of situation where the information about current heart beat rate becomes the indispensable part of soldier, this project emerges out as best to acknowledge the doctors at server site with the correct and fast information. If heart beat either increases above critical level or decreases below the critical level is sensed by Pulse rate sensor. These parameters of sensor are sent to server so that we can read that parameters.

An emergency switch is provided in the system, which when pressed by the soldier an message is given to authorized person about the emergency which is done through programming.

**Camera functioning -**

The USB Camera is an important component of this project as it continuously captures images which is helpful for watching the ground situation, where the soldier is stationed.

**Tracking-**

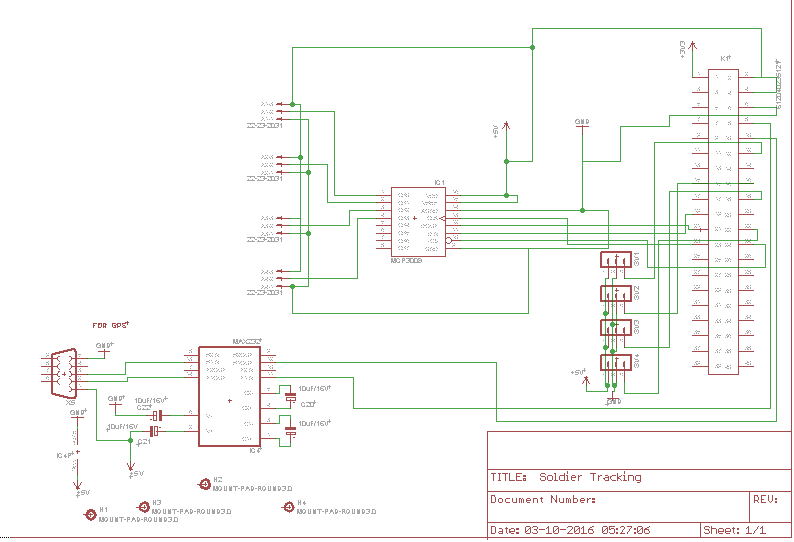
GPS module connected to the system will continuously track the location of soldier by sending the co-ordinates of location i.e. latitude and longitude.

**Emergency Key Function -**

The emergency key is provided to help the soldier communicate with the authorized persons. When the key is pressed, the emergency status on the web page changes from ‘Null’ to ‘Yes’, so that the persons in the control room understand that the soldier is in trouble. Also, the authorized persons get a text, alerting them to the position of the soldier.

### 5. Detail Design

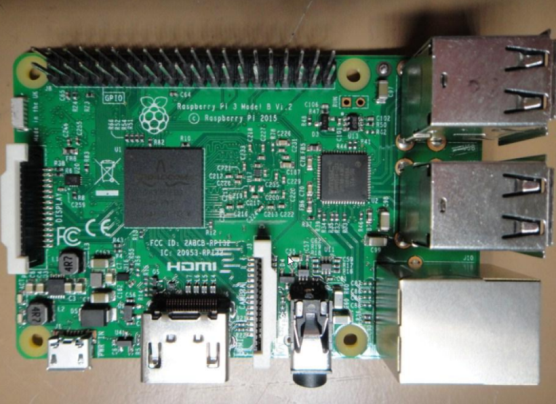
**5.1 Circuit Diagram:**



*Fig 5.1.1 - Circuit Diagram*

**5.2 Component Description:**

**5.2.1 RASPBERRY PI 3:**



*Fig 5.2.1 - Raspberry Pi module*

The stunning new Raspberry Pi 3 Model B is third generation model that maintains the same popular board format as the Raspberry Pi 2 and Raspberry Pi B+, but boasts a faster 1.2GHz 64Bit SoC, and on board WiFi and Bluetooth!

It is recommended to use a 2.5A Power Supply with the Raspberry Pi 3

**Specifications:**

Processor

• Broadcom BCM2387 chipset.

• 1.2GHz Quad-Core ARM Cortex-A53 (64Bit)

• 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)

GPU

• Dual Core Video Core IV® Multimedia Co-Processor. Provides Open GL ES 2.0, hardware-accelerated Open VG, and 1080p30 H.264 high-profile decode.

• Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure

Memory

• 1GB LPDDR2

Operating System

• Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IOT.

Dimensions

•85 x 56 x 17mm

Power

• Micro USB socket 5V1, 2.5A

Ethernet

•10/100 BaseT Ethernet socket

Video Output

• HDMI (rev 1.3 & 1.4)

• Composite RCA (PAL and NTSC)

Audio Output

• Audio Output 3.5mm jack

• HDMI

• USB 4 x USB 2.0 Connector

GPIO Connector

• 40-pin 2.54 mm (100 mil) expansion header: 2x20 strip

• Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines

Camera Connector

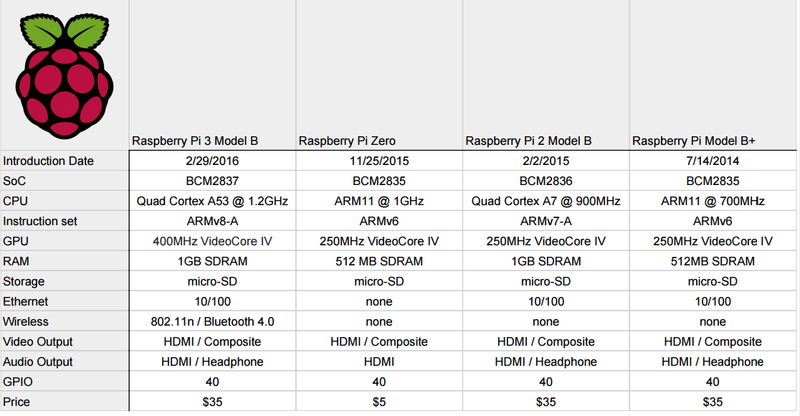
• 15-pin MIPI Camera Serial Interface (CSI-2)

Display Connector

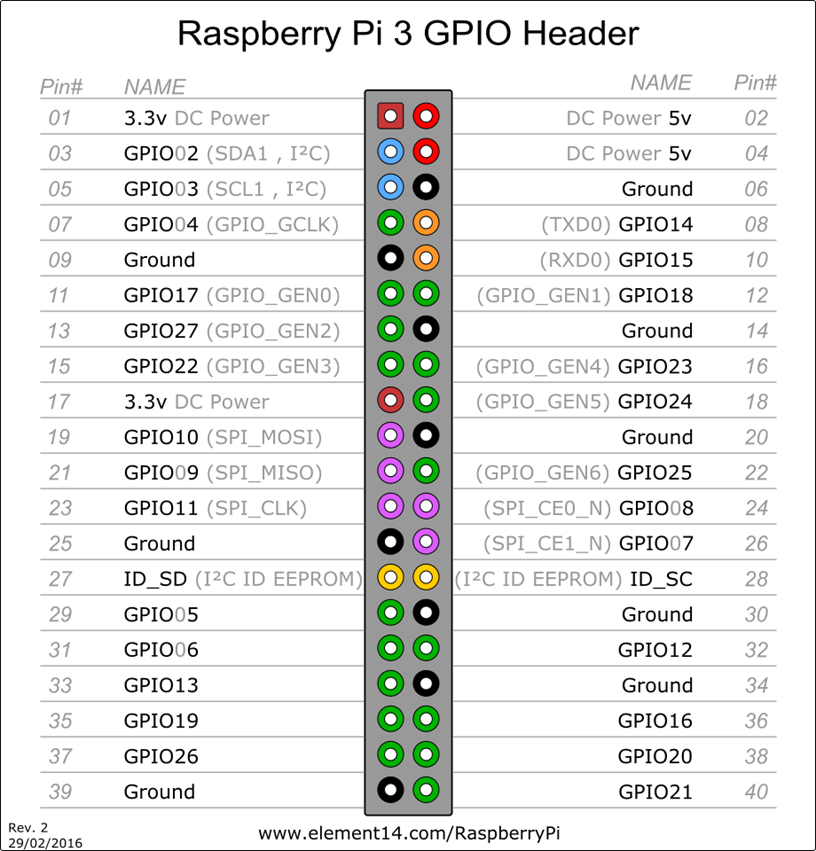
• Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane

Memory Card Slot

• Push/pull Micro SDIO

[](https://hackadaycom.files.wordpress.com/2016/02/pispecs2.png)

*5.2.1 Raspberry Pi Comparison*



*Fig 5.2.2 - Raspberry Pi pin diagram*

**5.2.2 HEART BEAT SENSOR:**

The working of Heart Beat sensor is based on the principle of

photoplethysmography (PPG) which is a non-invasive method of measuring the variation in blood volume in tissues using a light source and a detector. Since the change in blood volume is synchronous to the heartbeat, this technique can be used to calculate the heart rate.

****

*Fig****5.2.3-*** *Heartbeat sensor*

**Features:**

-Work voltage: DC 5V

-The output valid signal is high, the light goes out

-Sensitivity adjustable (fine tuning)

-Heart beat detection range, non-directional

**Applications of sensor:**

• Digital Heart Rate monitor

• Patient Monitoring System

• Bio-Feedback control of robotics and applications

**5.2.3 TEMPERATURE SENSOR (LM35):**

**LM35 Features**

• Calibrated directly in o Celsius (Centigrade)

• Linear + 10.0 mV/degrees C scale factor; that is for every +1 centigrade rise in temperature

there will be +10mV higher voltage at output pin.

• 0.5oC accuracy guarantee able (at +25oC)

• Rated for full −55o to +150oC range

• Suitable for remote applications

• Low cost due to wafer-level trimming

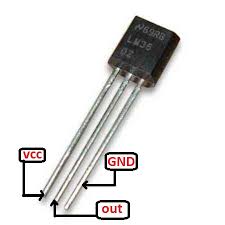
• Operates from 4 to 30 volts

• Less than 60 µA current drain

• Low self-heating, 0.08oC in still air

•Nonlinearity only ±1⁄4oC typical

• Low impedance output, 0.1 Ω for 1 mA load



*fig5.2.4- LM35 sensor*

**5.2.4. ADC (MCP3008):**

The MCP3008 has been chosen as it has 10-bit Analog-to-Digital Converter (ADC) combines high performance and low power consumption in a small package, making it ideal for embedded control applications. The MCP3008 features a successive approximation register (SAR) architecture and an industry-standard SPI serial interface.The MCP3008 features 200k samples/second, 8 input channels, low power consumption (5nA typical standby, 425µA typical active), and is available in 16-pin PDIP and SOIC packages. Applications for the MCP3008 include data acquisition, instrumentation and measurement, multi-channel data loggers, industrial PCs, motor control, robotics, industrial automation, smart sensors, portable instrumentation and home medical appliances.

**Features:**

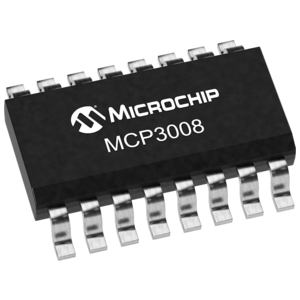
 10-bit resolution

 8 single-ended channels

 SPI interface

 200 ksps sample rate at 5V

 -40 to +85°C temperature range



*Fig 5.2.5 - ADC MP3008*

Design Considerations:

1. The ADC is interfaced to Raspberry Pi because it is required to convert the analog output of temperature sensor LM35 to digital output.

2. This ADC is 10 bit so it can report a range of numbers from 0 to 1023 (2 to the power 10).

3. A reading of 0 means the input is 0V and a reading of 1023 means the input is 3.3V. Thus 0-3.3V range would equate to temperature range of -55 to 150 degrees C.

4. The smallest analog voltage the ADC can detect is VREF/1024, which in this case is 3.3V/1024 =3.22mV and it represents digital value of 1.

5. The equation that converts analog voltage to its digital interpretation is 1024\*VIN/VREF. Where VIN is the analog input voltage and VREF is the reference voltage.

**5.2.5. GPS Module:**



*Fig 5.2.6. GPS Module*

SR-100 is a low-power, ultra-high performance, easy to use GPS smart antenna module based on SiRF’s third generation single chip. Its low power consumption, tiny size, and ultra-high performance enable the adoption of handheld applications.

The slim design allows SR-100 to be placed on top side of the housing to have best GPS signal reception. The 6-pin I/O interface is then connected to the main board with either connector or wire soldering. The integrated antenna helps reducing the RF and EMI issues to minimum. Fast adoption and high yield production becomes possible. The embedded RF connector allows receiving GPS signal using external active antenna when the built-in antenna is blocked from GPS signal.

**Main Features:-**

Not only handheld but also any other GPS applications can share the following major features of SR-100.

 Small – 13 x 16 x 9 (mm) with patch antenna of 12.8 x 12.8 x 4 (mm)

 Dual antenna support: Built-in patch antenna and MMCX RF connector for

 External antenna connection

 Easy use - connecting 4~6 digital pins & ready to use

 Backup power pin allowing hot/warm starts and better performance

 Minimum RF and EMI efforts

High tracking sensitivity of -159dBm

**5.2.6 USB Camera**:



*Fig 5.2.7. USB Camera*

Iball Robo K20 Webcam

HD: No Still Image Sensor Resolution: 20 MP Video Sensor Resolution : 2

MP Built-In Microphone: Yes Warranty: 2 years

DESCRIPTION OF IBALL ROBO K20

-Built in High Sensitive USB Microphone

-High quality 5G wide angle lens for sharp and clear picture,

-4 LEDs for night vision with Brightness controller, snapshot Button to take pictures and change special effects,

-4 x Digital zoom and auto face tracking. 20Megha Pixel still image resolution,

-2 Mega Pixel Video resolution.

**Specifications:-**

 Image Sensor: High quality 1/6 " CMOS \* Sensor

 Sensor resolution: 300k Pixels

 Video Format: 24 Bit True Color

 Lens: high quality 5G wide angle lens

 LED's: 4 LEDs for night vision, with brightness controller

 USB Interface: USB 2.0 Backward compatible with USB1.1

 Microphone: Built in high sensitive USB microphone

 Max Image Resolution: 5500x3640 pixels

 Max Video Resolution: 1600x1200pixels

 Adjustable Focus: 5 cm to Infinity

 Automatic Exposure: YES

 Automatic White Balance: YES

 Automatic Compensation: YES

 Power Supply: USB bus powered

 Operating Systems Supported: Windows XP, vista,7,8 (Mac and Linux with limited

functions)

**Software Features:**

 Camera Driver

 4x Digital Zoom Function

 Auto face tracking Function

 7 Photo frames

 9 Special effects

**5.2.7. HDMI to VGA adapter:**

Pi-View is a high quality, high performance, convenient and compact electronic device which converts the digital HDMI video signals which are outputted from Raspberry Pi, into an analogue signal required by the widely used VGA based computer displays still widely used around the world.



*Fig 5.2.8.- HDMI to VGA cable*

 Simple device design in the form of a small adaptor with integrated VGA socket output, with short cable complete with HDMI plug for connection to the HDMI socket on Raspberry Pi.

 No external power supply is required, and in operation, the device is plug and play\*

 The device handles display resolutions of up to 720p/1080i/1080p (the maximum resolution is limited by the resolution supported by the connected display device).

 Input: integrated Type-A HDMI plug,

 Output: standard female DE-15 connector.

 Material: 99% high-purity oxygen-free copper wire core, 30μ gold-plated terminals for minimal signal loss, and gold-plated HDMI plug for high abrasion resistance.

* + 1. **Power Supply Design :-**

**DC Power Input:**

The Power supply to be used has to be 12 Volts DC, 1Amp. The DC jack connectivity details are shown in the figure.



*Fig 5.2.9.- DC Power Jack*

A slide switch is provided for power ON/OFF control. The slide switch is useful only when an external DC adapter is used. When using the adapter, sliding the switch towards the arrow shown in figure will turn the board ON.

Design Steps:

To convert 230V AC and 50 Hz supply to 5V DC

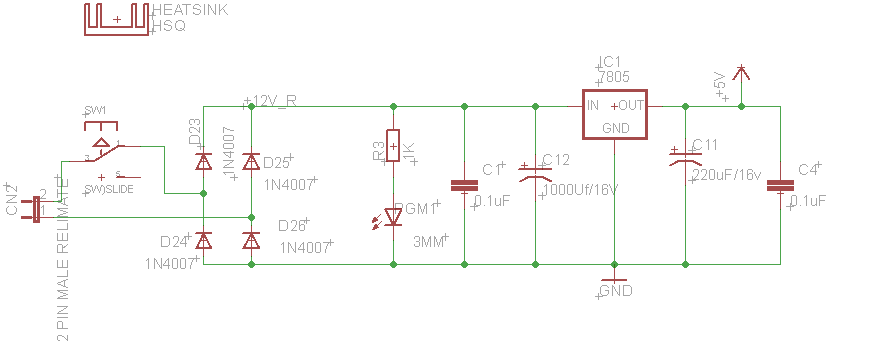
1. Step down transformer-It is used to convert 230V AC to 12V AC.

2. Bridge Rectifier- Includes 4 1N4007 diodes which are used to convert AC to DC, thus 12V AC is converted to 12V pulsating DC.

3. Filter Capacitor- A 1000uF electrolytic capacitor is required for filtering the pulsating DC signal. Thus we get pure 12V DC at the output of capacitor.

4. Voltage Regulator- IC7805 is a 3 terminal voltage regulator used to supply 5V supply voltage to the Raspberry Pi.

5. Heat Sink- It is used to protect the voltage regulator IC from excess heat.

****

*Fig 5.2.10 Power Supply Design*

**5.3 Software:**

**5.3.1 Algorithm:**

1 ) Start.

2 ) Read the data from GPS (latitude and longitude) and upload it on the web server.

3 ) Read the data from the sensors(pulse rate and temperature) and upload it on the web server.

4 ) Check for current situation by clicking on ‘Current Snapshot’ button on the web page.

5 ) Check the recorded data by clicking on ‘Soldier Table Values’ on web page.

6 ) We can add or remove records as required in the soldier table.

7 ) Check if emergency status is ‘NULL’ or ‘YES’. If ‘YES’ then check for the message regarding the position of the soldier of the soldier on the registered phone number.

8 ) Stop.

**5.3.2 JAVA:**

Over the ages people have used tools to help them accomplish tasks, but lately their tools have been getting smarter and interconnected. Microprocessors have appeared inside many commonly used items, and increasingly, they have been connected to networks. As the heart of personal computers and workstations, for example, microprocessors have been routinely connected to networks. They have also appeared inside devices with more specific functionality than the personal computer or the workstation. Televisions, VCRs, audio components, fax machines, scanners, printers, cell phones, personal digital assistants, pagers, and wrist-watches--all have been enhanced with microprocessors; most have been connected to networks. Given the increasing capabilities and decreasing costs of information processing and data networking technologies, the network is rapidly extending its reach.

The emerging infrastructure of smart devices and computers interconnected by networks represents a new environment for software--an environment that presents new challenges and offers new opportunities to software developers. Java is well suited to help software developers meet challenges and seize opportunities presented by the emerging computing environment, because Java was designed for networks. Its suitability for networked environments is inherent in its architecture, which enables secure, robust, platform- independent programs to be delivered across networks and run on a great variety of computers and devices.

The Architecture

Java's architecture arises out of four distinct but interrelated technologies:

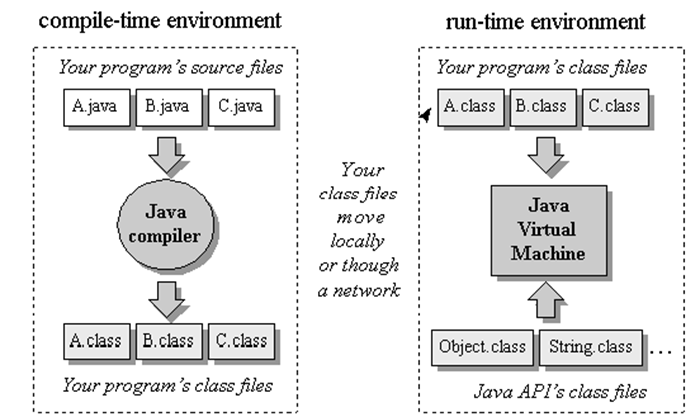
- The Java programming language

- The Java class file format

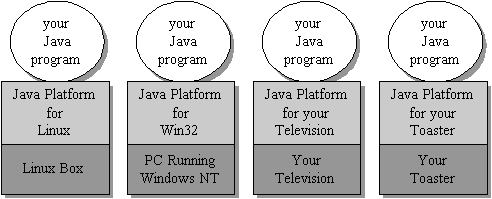
- The Java Application Programming Interface -the Java virtual machine.

When you write and run a Java program, you are tapping the power of these four technologies. You express the program in source files written in the Java programming language, compile the source to Java class files, and run the class files on a Java virtual machine. When you write your program, you access system resources (such as I/O, for example) by calling methods in the classes that implement the Java Application

Programming Interface, or Java API. As your program runs, it fulfils your program's Java API calls by invoking methods in class files that implement the Java API. You can see the relationship between these four parts in Figure 5.3.2.1

  
*Fig 5.3.2.1.- Java programming environment.*

Together, the Java virtual machine and Java API form a "platform" for which all Java programs are compiled. In addition to being called the *Java runtime system*, the combination of the Java virtual machine and Java API is called the *Java Platform* (or, starting with version 1.2, the *Java 2 Platform*). Java programs can run on many different kinds of computers because the Java Platform can itself be implemented in software. As you can see in Figure 1- 2, a Java program can run anywhere the Java Platform is present.

  
  
*Fig 5.3.2.2. Java programs run on top of the Java Platform.*

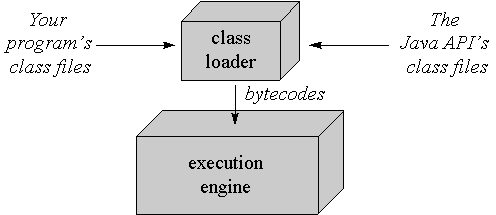
**The Java Virtual Machine**

At the heart of Java's network-orientation is the Java virtual machine, which supports all three prongs of Java's network-oriented architecture: platform independence, security, and network-mobility.

The Java virtual machine is an abstract computer. Its specification defines certain features every Java virtual machine must have, but leaves many choices to the designers of each implementation.

For example, although all Java virtual machines must be able to execute Java bytecodes, they may use any technique to execute them. Also, the specification is flexible enough to allow a Java virtual machine to be implemented either completely in software or to varying degrees in hardware. The flexible nature of the Java virtual machine's specification enables it to be implemented on a wide variety of computers and devices.

A Java virtual machine's main job is to load class files and execute the bytecodes they contain. As you can see in Figure 1-3, the Java virtual machine contains a *class loader*, which loads class files from both the program and the Java API. Only those class files from the Java API that are actually needed by a running program are loaded into the virtual machine. The bytecodes are executed in an *execution engine*.

  
  
*Fig 5.3.2.3. A basic block diagram of the Java virtual machine.*

The execution engine is one part of the virtual machine that can vary in different implementations. On a Java virtual machine implemented in software, the simplest kind of execution engine just interprets the bytecodes one at a time. Another kind of execution engine, one that is faster but requires more memory, is a *just-in-time compiler*. In this scheme, the bytecodes of a method are compiled to native machine code the first time the method is invoked.

The native machine code for the method is then cached, so it can be re-used the next time that same method is invoked. A third type of execution engine is an *adaptive optimizer*.

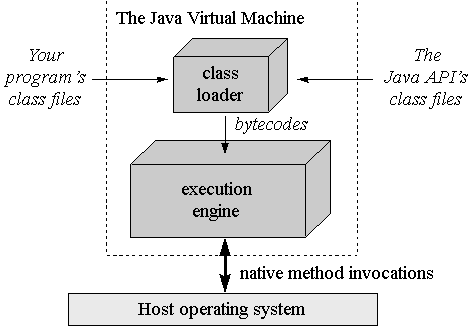
In this approach, the virtual machine starts by interpreting bytecodes, but monitors the activity of the running program and identifies the most heavily used areas of code. As the program runs, the virtual machine compiles to native and optimizes just these heavily used areas. The rest of the code, which is not heavily used, remain as bytecodes which the virtual machine continues to interpret. This adaptive optimization approach enables a Java virtual machine to spend typically 80 to 90% of its time executing highly optimized native code, while requiring it to compile and optimize only the 10 to20% of the code that really matters to performance. Lastly, on a Java virtual machine built on top of a chip that executes Java bytecodes natively, the execution engine is actually embedded in the chip.

Sometimes the Java virtual machine is called the *Java interpreter*; however, given the various ways in which bytecodes can be executed, this term can be misleading. While "Java interpreter" is a reasonable name for a Java virtual machine that interprets bytecodes, virtual machines also use other techniques (such as just-in-time compiling) to execute bytecodes. Therefore, although all Java interpreters are Java virtual machines, not all Java virtual machines are Java interpreters.

When running on a Java virtual machine that is implemented in software on top of a host operating system, a Java program interacts with the host by invoking *native methods*. In Java, there are two kinds of methods: Java and native. A Java method is written in the Java language, compiled to bytecodes, and stored in class files. A native method is written in some other language, such as C, C++, or assembly, and compiled to the native machine code of a particular processor. Native methods are stored in a dynamically linked library whose exact form is platform specific.

While Java methods are platform independent, native methods are not. When a running Java program calls a native method, the virtual machine loads the dynamic library that contains

the native method and invokes it. As you can see in Figure 1-4, native methods are the connection between a Java program and an underlying host operating system.



*Fig 5.3.2.4. A Java virtual machine implemented in software on top of a host operating system.*

You can use native methods to give your Java programs direct access to the resources of the underlying operating system. Their use, however, will render your program platform specific, because the dynamic libraries containing the native methods are platform specific.

In addition, the use of native methods may render your program specific to a particular implementation of the Java Platform. One native method interface--the *Java Native Interface*, or *JNI*--enables native methods to work with any Java Platform implementation on a particular host computer. Vendors of the Java Platform, however, are not necessarily required to support JNI. They may provide their own proprietary native method interfaces in addition to (or depending on their contract, in place of) JNI.

Java gives you a choice. If you want to access resources of a particular host that are unavailable through the Java API, you can write a platform-specific Java program that calls native methods. If you want to keep your program platform independent, however, you must access the system resources of the underlying operating system only through the Java API.

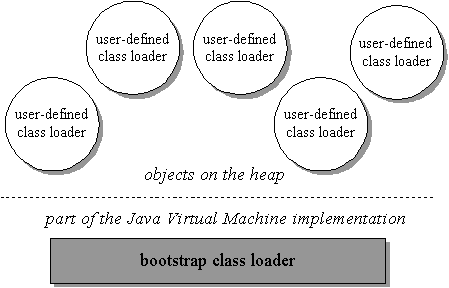
**- The Class Loader Architecture**

One aspect of the Java virtual machine that plays an important role in both security and network- mobility is the class loader architecture. In the block diagrams of Figures 1-3 and 1-4, a single mysterious cube identifies itself as "the class loader," but in reality there may be more than one class loader inside a Java virtual machine. Thus the class loader cube of the block diagram actually represents a subsystem that may involve many class loaders. The Java virtual machine has a flexible class loader architecture that allows a Java application to load classes in custom ways.

A Java application can use two types of class loaders: a "bootstrap" class loader and user-defined class loaders. The bootstrap class loader (there is only one of them) is a part of the

Java virtual machine implementation. For example, if a Java virtual machine is implemented as a C program on top of an existing operating system, then the bootstrap class loader will be part of that C program. The bootstrap class loader loads classes, including the classes of the Java API, in some default way, usually from the local disk. (The bootstrap class loader has also been called the primordial class loader, system class loader, or default class loader. In 1.2, the name "system class loader" was given a new meaning, which is described in Chapter 3.)

At run-time, a Java application can install user-defined class loaders that load classes in custom ways, such as by downloading class files across a network. While the bootstrap class loader is an intrinsic part of the virtual machine implementation, user-defined class loaders are not. Instead, user-defined class loaders are written in Java, compiled to class files, loaded into the virtual machine, and instantiated just like any other object. They are really just another part of the executable code of a running Java application. You can see a graphical depiction of this architecture in Figure 1-5.

  
  
*Fig 5.3.2.5. Java's class loader architecture.*

Because of user-defined class loaders, you don't have to know at compile-time all the classes that may ultimately take part in a running Java application. User-defined class loaders enable you to dynamically extend a Java application at run-time. As it runs, your application can determine what extra classes it needs and load them through one or more user-defined class loaders. Because you write the class loader in Java, you can load classes in any manner

expressible in Java code. You can download them across a network, get them out of some kind of database, or even calculate them on the fly.

For each class it loads, the Java virtual machine keeps track of which class loader--whether bootstrap or user-defined--loaded the class. When a loaded class first refers to another class, the virtual machine requests the referenc*ed* class from the same class loader that originally loaded the referenc*ing* class. For example, if the virtual machine loads class Volcano through a particular class loader, it will attempt to load any classes Volcano refers to through the same class loader. If Volcano refers to a class named Lava, perhaps by invoking a method in class Lava, the virtual machine will request Lava from the class loader that loaded Volcano. The Lava class returned by the class loader is dynamically linked with class Volcano.

Because the Java virtual machine takes this approach to loading classes, classes can by default only see other classes that were loaded by the same class loader. In this way, Java's architecture enables you to create multiple *name-spaces* inside a single Java application. Each class loader in your running Java program has its own name-space, which is populated by the names of all the classes it has loaded.

A Java application can instantiate multiple user-defined class loaders either from the same class or from multiple classes. It can, therefore, create as many (and as many different kinds of) user-defined class loaders as it needs. Classes loaded by different class loaders are in different name- spaces and cannot gain access to each other unless the application explicitly allows it. When you write a Java application, you can segregate classes loaded from different sources into different name-spaces. In this way, you can use Java's class loader architecture

to control any interaction between code loaded from different sources. In particular, you can prevent hostile code from gaining access to and subverting friendly code.

One example of dynamic extension is the web browser, which uses user-defined class loaders to download the class files for applets across a network. A web browser fires off a Java application that installs a user-defined class loader--usually called an *applet class loader*-- that knows how to request class files from an HTTP server. Applets are an example of dynamic extension, because the Java application doesn't know when it starts which class files the browser will ask it to download across the network. The class files to download are determined at run-time, as the browser encounters pages that contain Java applets.

The Java application started by the web browser usually creates a different user-defined class loader for each location on the network from which it retrieves class files. As a result, class files from different sources are loaded by different user-defined class loaders. This places them into different name-spaces inside the host Java application. Because the class files for applets from different sources are placed in separate name- spaces, the code of a malicious applet is restricted from interfering directly with class files downloaded from any other source.

By allowing you to instantiate user-defined class loaders that know how to download class files across a network, Java's class loader architecture supports network-mobility. It supports security by allowing you to load class files from different sources through different user-defined class loaders. This puts the class files from different sources into different name-spaces, which allows you to restrict or prevent access between codes loaded from different sources.

**The Java Class File**

The Java class file helps make Java suitable for networks mainly in the areas of platform-independence and network-mobility. Its role in platform independence is serving as a binary form for Java programs that is expected by the Java virtual machine but independent of underlying host platforms. This approach breaks with the tradition followed by languages such as C or C++. Programs written in these languages are most often compiled and linked into a single binary executable file specific to a particular hardware platform and operating system. In general, a binary executable file for one platform won't work on another. The Java class file, by contrast, is a binary file that can be run on any hardware platform and operating system that hosts the Java virtual machine.

When you compile and link a C++ program, the executable binary file you get is specific to a particular target hardware platform and operating system because it contains machine language specific to the target processor. A Java compiler, by contrast, translates the instructions of the Java source files into bytecodes, the "machine language" of the Java virtual machine.

In addition to processor-specific machine language, another platform- dependent attribute of a traditional binary executable file is the byte order of integers. In executable binary files for the Intel X86 family of processors, for example, the byte order is *little-endian*, or lower order byte first. In executable files for the PowerPC chip, however, the byte order is *big-endian*, or higher order byte first. In a Java class file, byte order is big-endian irrespective of what platform generated the file and independent of whatever platforms may eventually use it.

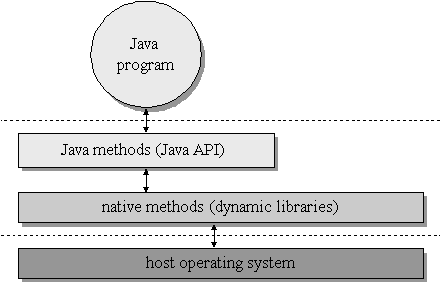
In addition to its support for platform independence, the Java class file plays a critical role in Java's architectural support for network-mobility. First, class files were designed to be compact, so they can more quickly move across a network. Also, because Java programs are dynamically linked and dynamically extensible, class files can be downloaded as needed. This feature helps a Java application manage the time it takes to download class files across a network, so the end-user's wait time can be kept to a minimum.

**The Java API**

The Java API helps make Java suitable for networks through its support for platform independence and security. The Java API is set of runtime librariesthat give you a standard way to access the system resources of a host computer. When you write a Java program, you assume the class files of the Java APIwill be available at any Java virtual machine that may ever have the privilege of running your program. This is a relatively safe assumption because the Java virtual machine and the class files for the Java API are the required components of any implementation of the Java Platform. When you run a Java program, the virtual machine loads the Java API class files that are referred to by your program's class files. The combination of all loaded class files (from your program and from the Java API) and any

loaded dynamic libraries (containing native methods) constitute the full program executed by the Java virtual machine.

The class files of the Java API are inherently specific to the host platform. The API's functionality must be implemented expressly for a particular platform before that platform can host Java programs. To access the native resources of the host, the Java API calls native methods. As you can see in Figure 1-6, the class files of the Java API invoke native methods so your Java program doesn't have to. In this manner, the Java API's class files provide a Java program with a standard, platform-independent interface to the underlying host. To the Java program, the Java API looks the same and behaves predictably no matter what platform happens to be underneath. Precisely because the Java virtual machine and Java API are implemented specifically for each particular host platform, Java programs themselves can be platform independent.



*Fig 5.3.2.6. A platform-independent Java program.*

The internal design of the Java API is also geared towards platform independence. For example, the graphical user interface libraries of the Java API, the Abstract Windows Toolkit (or AWT) and Swing, are designed to facilitate the creation of user interfaces that work on all platforms. Creating platform- independent user interfaces is inherently difficult, given that the native look and feel of user interfaces vary greatly from one platform to another. The AWT library's architecture does not coerce implementations of the Java API to give Java programs a user interface that looks exactly the same everywhere. Instead, it encourages implementations to adopt the look and feel of the underlying platform. The Swing library offers even more flexibility -- enabling the look and feel to be chosen by the programmer. Also, because the size of fonts, buttons, and other user interface components will vary from platform to platform, the AWT and Swing include *layout managers* to position the elements of a window or dialog box at run- time.

Rather than forcing you to indicate exact X and Y coordinates for the various elements that constitute, say, a dialog box, the layout manager positions them when your dialog box is displayed. With the aim of making the dialog look its best on each platform, the layout manager will very likely position the dialog box elements slightly differently on different platforms. In these ways and many others, the internal architecture of the Java API is aimed at facilitating the platform independence of the Java programs that use it.

In addition to facilitating platform independence, the Java API contributes to Java's security model. The methods of the Java API, before they perform any action that could potentially be harmful (such as writing to the local disk), check for permission. In Java releases prior to 1.2, the methods of the Java API checked permission by querying the security manager.

The security manager is a special object that defines a custom security policy for the application. A security manager could, for example, forbid access to the local disk. If the application requested a local disk write by invoking a method from the pre-1.2 Java API, that method would first check with the security manager.

Upon learning from the security manager that disk access is forbidden, the Java API would refuse to perform the write. In Java 1.2, the job of the security manager was taken over by the *access controller*, a class that performs stack inspection to determine whether the operation should be allowed. (For backwards compatibility, the security manager still exists in Java 1.2.) By enforcing the security policy established by the security manager and controller, the Java API helps to establish a safe environment in which you can run potentially unsafe code.

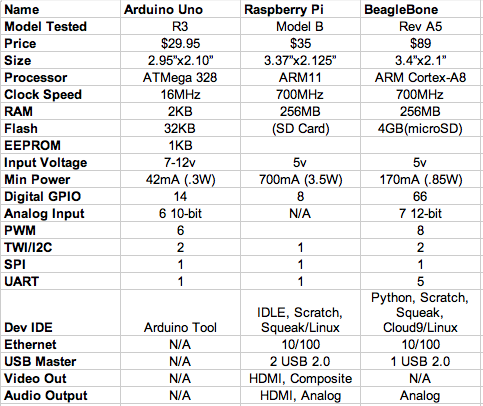
**5.3.3 Web Server:**

Web server is used to communicate with Web Browsers as its clients and the communication protocol used in this case is HTTP (Hyper Text Transfer Protocol). This is why a Web Server is also called an HTTP Server.

The client (i.e., the Web Browser) and the server (i.e., HTTP/Web Server) should be able to communicate with each other in a defined way. This pre-defined set of rules which form the basis of the communication are normally termed as a protocol

**Comparison of Raspberry with the competitors**

The chief competitors of the Raspberry Pi are the Arduino and the Beagleboard. Both are single board computers and have applications similar to the Raspberry Pi. A brief comparison of the three of them is shown below:



*5.3.1: Comparison of RPi with chief competitor*

Different wireless network standards support different maximum network speeds:

**802.11a**

802.11a is an IEEE wireless networking standard that specifies a maximum data transfer rate of 54Mbps and an operating frequency of 5GHz.

**802.11b**

802.11b is an IEEE wireless networking standard that specifies a maximum data transfer rate of 11Mbps and an operating frequency of 2.4GHz.

**802.11g**

802.11g is an IEEE wireless networking standard that specifies a maximum data transfer rate of 54Mbps and an operating frequency of 2.4GHz. 802.11g is backwards compatible with 802.11b. Some wireless products come with the extended 802.11g standard called super

G/G+ (different manufacturers may call it something else) capable of offering transfer rates of 108Mbps, 125Mbps or even higher.

**802.11n**

The next generation of high-speed wireless networking standard, the draft 802.11n standard specifies a maximum data transfer rate of up to 540Mbps. It is built on the basis of previous 802.11 standards with the addition of MIMO (multiple-input multiple-output) technology. It is required to be backward compatible and interoperable with 802.11b/g devices. The final 802.11n specification is expected to be approved in 2007 so that contemporary 802.11n-

supporting products can provide different maximum data transfer rates (e.g. 270Mbps, 300Mbps) .

The maximum wireless data transfer rate of a wireless router will only be realized with the support of the client network device (wireless network adapter).

This means that a transfer rate of 54Mbps (for example) will be realized only if both the router and the network adapter support the 802.11g standard. For more information about wireless network adapters

**Security protocols**

Security is a vital concern for wireless network users. Wireless network security protocols may be capable of safeguarding data and preventing it from being used without authorization when it is transferred across a wireless network. Here are the most popular security protocols:

**WEP**

WEP (Wired Equivalent Privacy) is a security protocol for wireless networks. It aims to protect data via encryption over radio waves using point-to-point transmission. A shared key (similar to a password) is used to allow communication between the wireless adapter and the wireless AP/router. It does not protect users on the same network from each other, however.

**WPA**

WPA (Wi-Fi Protected Access) is built on the foundation of WEP. WPA also protects wireless data transmission via a key similar to WEP, but the added strength of WPA is in its automatic encryption key changes making it much more difficult for a hacker to invade a wireless network.

**WPA2**

As the successor to WPA, WPA2 (Wi-Fi Protected Access 2) is offered to users requiring the highest level of wireless security and is capable of offering a stronger encryption mechanism over WPA via AES (Advanced Encryption Standard).

Like data transfer rates, wireless security protocols require the support of both the wireless network adapter and the wireless network AP/router, meaning that protection via the wireless security protocol will not take effect if either of the two does not support it.

**Antenna**

Like any wireless device, the wireless router requires the help of an antenna to communicate with a wireless network. There are two types of antennas applied to wirelessrouters: external and internal. A wireless router with an external antenna sends and receives wireless signals at a higher level of signal performance than a wireless router that does not. Internal antennas have broad applications in ultra-slim/portable wireless routers, some of which also provide external antenna ports that allow users to install external antennas for improved signal performance.

**5.3.4 WiFi Hotspots**

A **WiFi hotspot** is simply an area with an accessible wireless network. The term is most often used to refer to wireless networks in public areas like airports and coffee shops. Some are free and some require fees for use, but in either case they can be handy when you are on the go. You can even create your own mobile hotspot using a cell phone or an external device that can connect to a cellular network. And you can always set up a WiFi network at home.

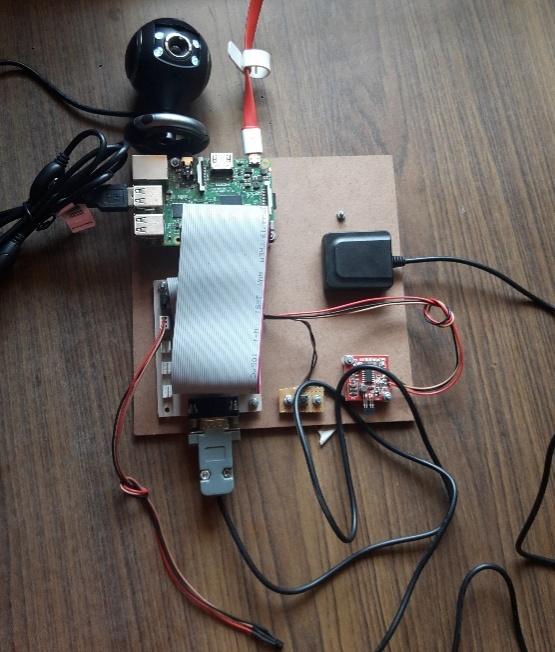
If you want to take advantage of public WiFi hotspots or your own home-based network, the first thing you'll need to do is make sure your computer has the right gear. Most new [laptops](http://computer.howstuffworks.com/laptop.htm) and many new desktop computers come with built-in wireless transmitters, and just about all mobile devices are WiFi enabled. If your computer isn't already equipped, you can buy a **wireless adapter** that plugs into the PC card slot or [USB](http://computer.howstuffworks.com/usb.htm) port. Desktop computers can use USB adapters, or you can buy an adapter that plugs into the PCI slot inside the computer's case. Many of these adapters can use more than one 802.11 standard.

Once you've installed a wireless adapter and the drivers that allow it to operate, your computer should be able to automatically discover existing networks. This means that when you turn your computer on in a WiFi hotspot, the computer will inform you that the network exists and ask whether you want to connect to it. If you have an older computer, you may need to use a software program to detect and connect to a wireless network.

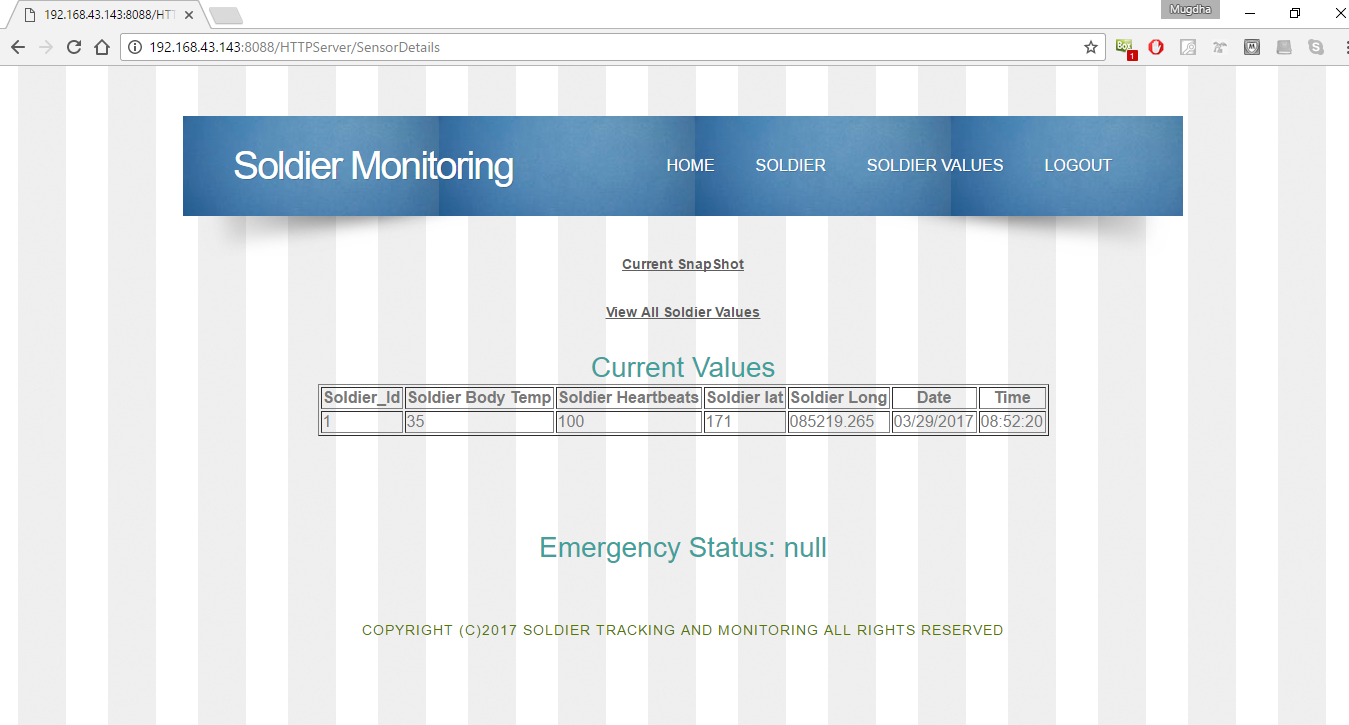
Being able to connect to the Internet in public hotspots is extremely convenient. Wireless [home networks](http://computer.howstuffworks.com/home-network.htm) are convenient as well. They allow you to easily connect multiple computers and to move them from place to place without disconnecting and reconnecting wires. In the next section, we'll look at how to create a wireless network in your home.

### Results

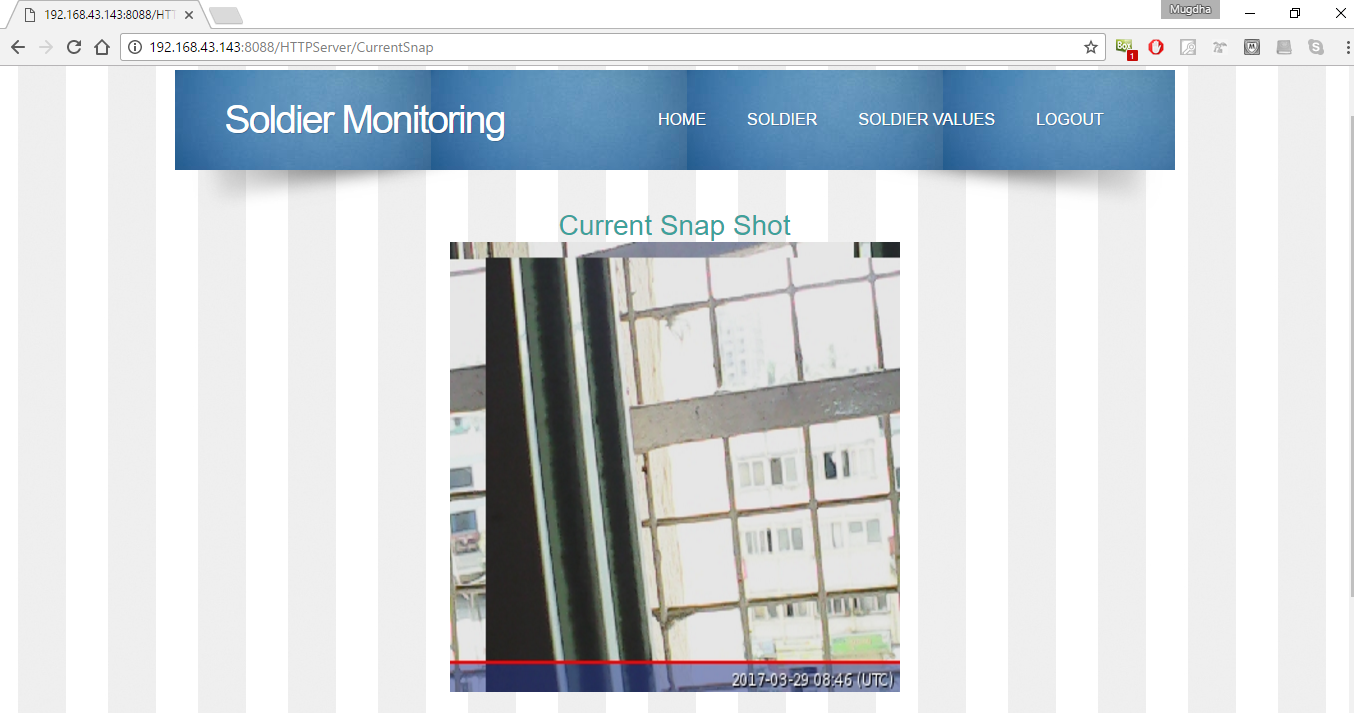
The following figures show the final results of our project.



*Fig 6.1- Project Setup*



*Fig 6.2 – Web Page*



*Fig 6.3 – Current Snapshot*

### 7. Conclusion

In conclusion, this system can be useful to the army for applications such as real time tracking, long range coverage, and low cost.

It will also provide high level of security and reliability and it gives an option to operate the entire system as a mobile entity. However, this project can only be used during the day. The camera only takes pictures in case the switch has been pressed, which might delay the time of response in sending help. Even considering the shortcomings of this project, it can prove very useful for the dismounted soldiers now, and in future with some improvements.

### 8. Future scope

The future scope of this project includes variety of applications such as:-

1. Live video feed can be provided by interfacing an USB webcam to the Raspberry Pi. This will help us to monitor the situation in the field continuously, rather than just in case of emergencies. Hence, we can send help as soon as possible.

2. As the main aim is to provide an integrated system, we can also include some audio codecs which will help the soldiers to directly communicate with his comrades. Raspberry Pi does not have sophisticated audio capabilities therefore; we need a separate hardware and software for installation of audio codecs on RPi. This can be done using I2S communication protocol designed for audio applications. Example- TLV320A1C23, WM8731 etc.

3. Thermal imaging is another application that can be included in future for this system. MLX90620 and MLX90621 are both remote temperature sensors that operate via I2C.

4. A night vision camera that plugs into the CSI connector of Raspberry Pi can be used. This will help the soldiers when it’s dark.

It features 2 high intensity infrared LED spotlights for night time recording.

9. References

1) Wararkar, P., Mahajan, S., Mahajan, A., Banerjee, A., Madankar, A., & Sontakke, A. (2013). Soldier Tracking And Health Monitoring System International Journal of Computer Science & Applications, 2(02).

2) Govindraj, A., &Banu, D. S. GPS based soldier tracking and health indication system with environmental analysis. International Journal of Enhanced Research in Science Technology & Engineering, ISSN, 2319-7463.

3) Pramod, M. P. (2014). GPS Based Advanced Soldier Tracking With Emergency Messages & Communication System. International Journal, 2(6).

4) Sawan Mahajan, Ashu Mahajan, Arjit Banerjee,anchal mandankar,ashish sontakke and Prof Pravin wararkar.-“Soldier tracking and health monitoring system”- The International Journal of computer science and application(TIJCSA)

ISSN\_2278\_1080,Vol 2 no.02 April 20135) Govindaraj A. and Dr. S. Sindhuja Banu –

“GPS based soldier tracking and health indication system with environmental analysis.” International journal of enhanced research in science technology and engineering, ISSN: 2319\_7463 vol.2 Issue 12, December\_2013 pp: (46-52) Page no(s) 46, 51

Page no(s) 82, 83.