Arab Open University – Egypt

Faculty of Computer Studies

Information Technology and Computing Department

IoT GSM smart fire control system

Esraa Fahmy Abd-Elfattah Mohamed

1851710146

TM471: Final Year Project, 2022

Supervisor: Dr. / Amani Abdo

Contents

I. Abstract: 4
II. Acknowledgements: 4
1. Introduction: 5
1.1 Background: 5
1.2 Motivation: 5
1.3 Aims: 6
1.4 Objectives: 6
1.5 Deliverables: 7
1.6 Problem Definition: 7
1.7 Scope: 8
1.8 Target Customer: 8
1.9 Suggested Solution: 8
1.10 Next Chapters Summary: 9
2. Literature Review: 10
2.1 Overview: 10
2.2 Research Methodology: 10
2.3 Related Work: 11
2.3.1 First System: GSM Smart Home Security System. 11
2.3.2 Second System: Smart IoT Home Access Security System. 12
2.3.3 Third System: IoT Wireless Smart Home Security System. 13
3. Requirements & Analysis: 14
3.1 Project Lifecycle Model: 14
3.2 Project Requirements: 17
3.2.1 Functional Requirements: 17
3.2.2 Non-Functional Requirements: 18
3.2.3 Hardware Requirements: 19
3.2.4 Software Requirements: 20
3.2.3 Hrdware Tools……..20
3.2.3 Software Tools ……21
3.2.5 Gantt Chart: 21
3.3 Use Case Scenarios: 22
3.4 Use Case Diagram: 23
3.5 Data Flow Diagram: 24
4. Design, Implementation and Testing …….25
4.1.1 System Model……..…….....26
4.1.2 Main Approach.....27
4.1.3 Alternative Model……...29
4.1.4 PCB Design Schematic……..31
4.1.5 PCB Design Layout……...32
4.2 Implementation……..33
4.2.1 PCB (Top Layer)……...33
4.2.2 PCB (Bottom Layer)……34
4.2.3 Arduino-C Ciruit 1 code……...35

4.2.4 Arduino-C Ciruit 2 code……..36
4.2.5 knowledge of the main components……37
4.2.5 IoT Blynk Application……37
4.2.6 IoT Blynk Application Configuration……...39
4.2.7 Embedded System Testing……43
5 Results and Discussion……48
5.1.1 Blynk IoT……...48
5.1.2 PCB Manufacturing Process……..51
2.2 Goals Achieved……62
2.3 Future Work……..62
2.4 Ethical Issues……63
6 Conclusion……64
7 References:……65

List of Figures

Figure 1: System Figure..... 10
Figure 2: System Figure..... 11
Figure 3: System Figure..... 12
Figure 4: System Figure..... 13
Figure 5: Iterative Model 17
Figure 6: Use Case Diagram 23
Figure 7: Data Flow Diagram 24
Figure 8: System Model……26
Figure 9: LSD Software Approach……28
Figure 10: FDD Approach……30
Figure 11: PCB Design Schematic……31
Figure 12: PCB Design Layout……..32
Figure 13: Testing Technology Domains……..47
Figure 14: Blynk Architecture……..….49
Figure 15: PCB Prepreg……56
Figure 16: PCB Drilling……..…58
Figure 17: PCB Electrical Test……..….60

I. Abstract:

Internet of Things conceptualizes the idea of remotely connecting and monitoring real world objects (things) through the Internet. When it comes to our house, this concept can be aptly incorporated to make it smarter, safer and automated. This IoT project focuses on building a smart wireless home security system which sends alerts to the owner by using Internet in case of any trespass and raises an alarm optionally. Besides, the same can also be utilized for home safety by making use of the same set of sensors. The leverage obtained by preferring this system over the similar kinds of existing systems is that the alerts and the status sent by the LTE connected microcontroller managed system can be received by the user on his phone from any distance irrespective of whether his mobile phone is connected to the internet. The microcontroller used in the current prototype is the AVR ATmega328P.

Total Words: 150

II. Acknowledgements:

I cannot express enough thanks to my project supervisor, **Dr. Amani Abdo** for his continued support and encouragement. I offer my sincere appreciation for the learning opportunities provided by my supervisor.

Chapter 1:

1. Introduction:

1.1 Background:

Smart Home Environments integrate multiple IoT devices and services that collect, process and exchange data. They provide users several possibilities to control and adapt the status of their home, either manually or automatically. For that purpose, Smart Home devices and services exchange data with internal and external actors. These interactions take place with mobile applications on an end-user's equipment (smartphone) and also with remote services in the Cloud. Due to their interconnected nature, Smart Home devices are subject to a number of security threats either from remote attackers or from inside the Home Area Network (HAN). Moreover, these threats have an impact not only on a user's data but also on his/her health and safety: this changes the accepted idea that the home is usually a safe place to live in. Smart Home Environments being an emerging domain and because the liabilities are not well defined, it becomes important for all actors to develop adapted security measures to prevent cyber threats. For that purpose, there is a need to secure Smart Home Environments and effectively reduce the threats.

1.2 Motivation:

During the past few years, Internet was known as a big mass that we can acquire data from. Embedding mobile transceivers to everyday items and gadgets enabled new forms of bi-directional communication between people with other people, and people with things. That paradigm, known as Internet of Things that was first introduced in 1998 by Kevin Ashton has received recently more attention in the academia and industry, and this would add a new dimension to the world of Information and

communication technology. While that paradigm is growing and have high positive impact on many aspects of our lives, challenging issues arise, that should be considered and addressed. The central issues are guaranteeing security and privacy of users and their data. Another issue is fully achieving smartness of interconnected devices by enabling their interaction. Exchanging data and autonomous behavior is the key to achieving the latter. IoT has different definitions from different perspectives, however, they all revolve around" things" generally, collecting, exchanging and communicating data with each other's and with people through the" internet". IoT helps in decision-making and secure almost everything around us. The smarter life IoT vision promises soon through various applications, made smart Home Security possible, starting from basically monitoring different parts of home, to actually controlling them. Integration of IoT and Home Security, made it possible to monitor and secure homes from different parts of the world.

1.3 Aims:

- Remotely secure homes from theft and breach incidents.
- Easily monitor the safety and security statues of homes in a real-time manner.
- Speed up the reaction process in case of susceptible security breaches or safety problems.
- Ensure that all safety and security precautions are strict and valid.

1.4 Objectives:

- Using IoT for achieving a successful real-time communication system.

- Using a cloud IoT-based mobile platform for easy monitoring anytime, anywhere.
- Embedding intelligent precise sensors and actuators for detecting abnormalities.
- Embedding a reliable GSM chip for ensuring continuous stability.

1.5 Deliverables:

- IoT Cloud Platform.
- Hardware System (PCB).
- IoT Cloud Server (MQTT Broker).

1.6 Problem Definition:

Many people are always on the move from place to place due to business demands. Some people can spend a couple of days away from their home leaving all their household appliances without any kind of monitoring and control. Some devices are left plugged into power sockets whereas others are supposed to be plugged into and out of power sockets at different intervals depending on the time of the day. When you look at your family, and your home, you know you want them to be safe, always out of harm's way. When you leave for work, you expect to come back to a smiling family, and to a home that is secure. But as they say, hope is not a strategy. The growing crime rates across cities reflects the bitter reality. Many people overlook, ignore, and underestimate the need of taking appropriate home security measures. Therefore, we propose to design an Internet based home security system, which will alert user about risks involved with home anywhere, anytime.

1.7 Scope:

This project supports the following:

- Detecting smoke and gas leakages.
- Detecting elevated temperature and humidity.
- Detecting abnormal movements.
- Sending warning notifications and emails.

This project does <u>NOT</u> support the following:

- GSM mobile SMS warning messages.

1.8 Target Customer:

- Homes.
- Villas.
- Small workplaces.
- Private establishments.

1.9 Suggested Solution:

The suggested solution is to implement a remotely interconnected system based on IoT GSM real-time monitoring and notification. The suggested solution comprises smart and remotely connected sub-systems in order to ensure a secure and safe state in a home. The first sub-system is complete hardware system based on a GSM smart chip attached to smart sensors and actuators for detecting security breaches or safety vulnerabilities. This hardware system attaches a >113dB siren alarm that is automatically activated in case of emergencies. The second sub-system

is an IoT cloud online real-time broker server that applies “MQTT” protocol for handling remote ubiquitous data transmission. The third sub-system is an implementation of “Blynk” IoT platform that is supported on Android and iOS devices. This platform enables home owner to monitor and get notified in case of security emergency.

1.10 Next Chapters Summary:

In fact, **Chapter 2** will discuss many technical academic-wise and research methodology procedures for accomplishing this project. An illustrative literature review that will include similar systems comparison. **Chapter 3** will include a lot of technical information and issues related to my project’s requirements analysis.

Chapter 2:

2. Literature Review:

2.1 Overview:

At first, this chapter will include important data about the academic research that I will perform to implement my system. A literature review which is a survey of scholarly sources on a similar system to with my project. It will provide an overview of current knowledge, allowing me to identify relevant theories, methods, and gaps in my existing research. In addition, I will mention the research methodology that I will take advantage of to implement this system.

2.2 Research Methodology:

In fact, the following figure shows the academic steps I will go through during project research and implementation.

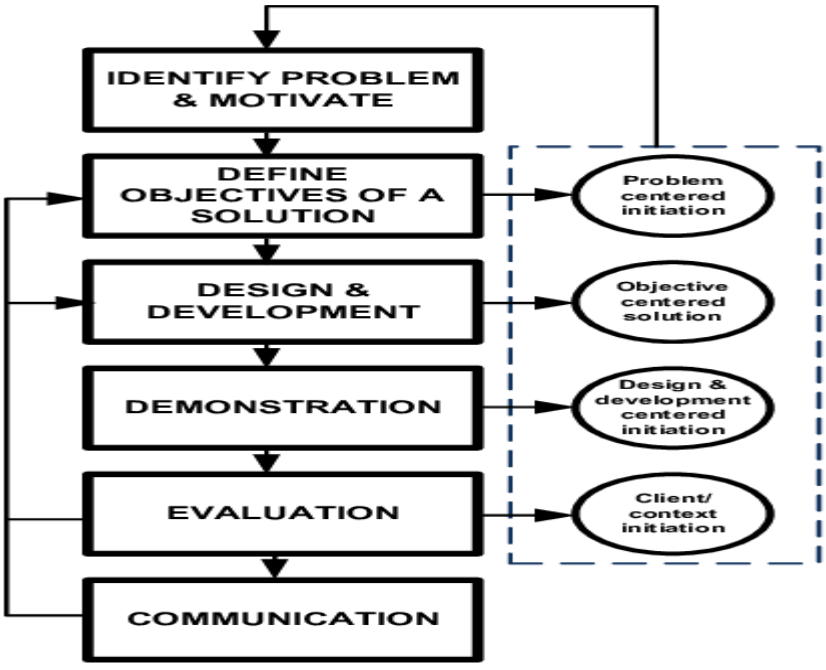


Figure 1: System Figure

2.3 Related Work:

Below, similar systems advantages and drawbacks will be mentioned.

2.3.1 First System: GSM Smart Home Security System.

Illustrative Image:

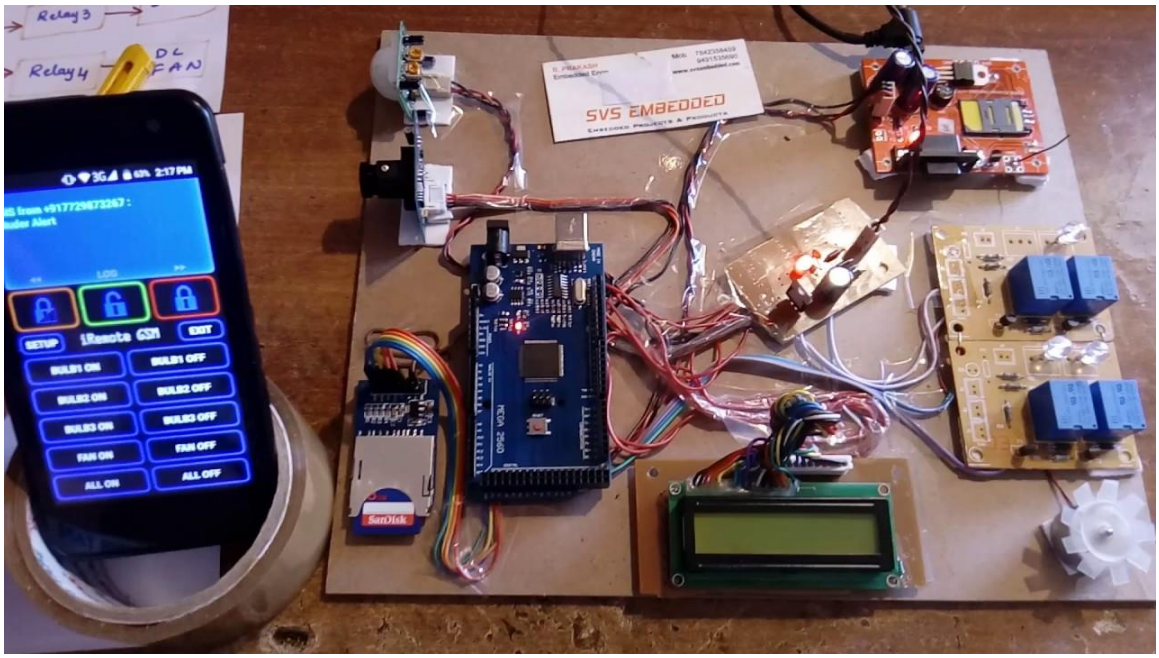


Figure 2: System Figure

Pros:

- Wireless communication through GSM network.
- SMS warning messages are supported.

Cons:

- No Printed Circuit Board is manufactured.
- Not all sensors are detectors are available.

Reference:

https://www.youtube.com/watch?v=sncULqQA5gs

2.3.2 Second System: Smart IoT Home Access Security System.

Illustrative Image:

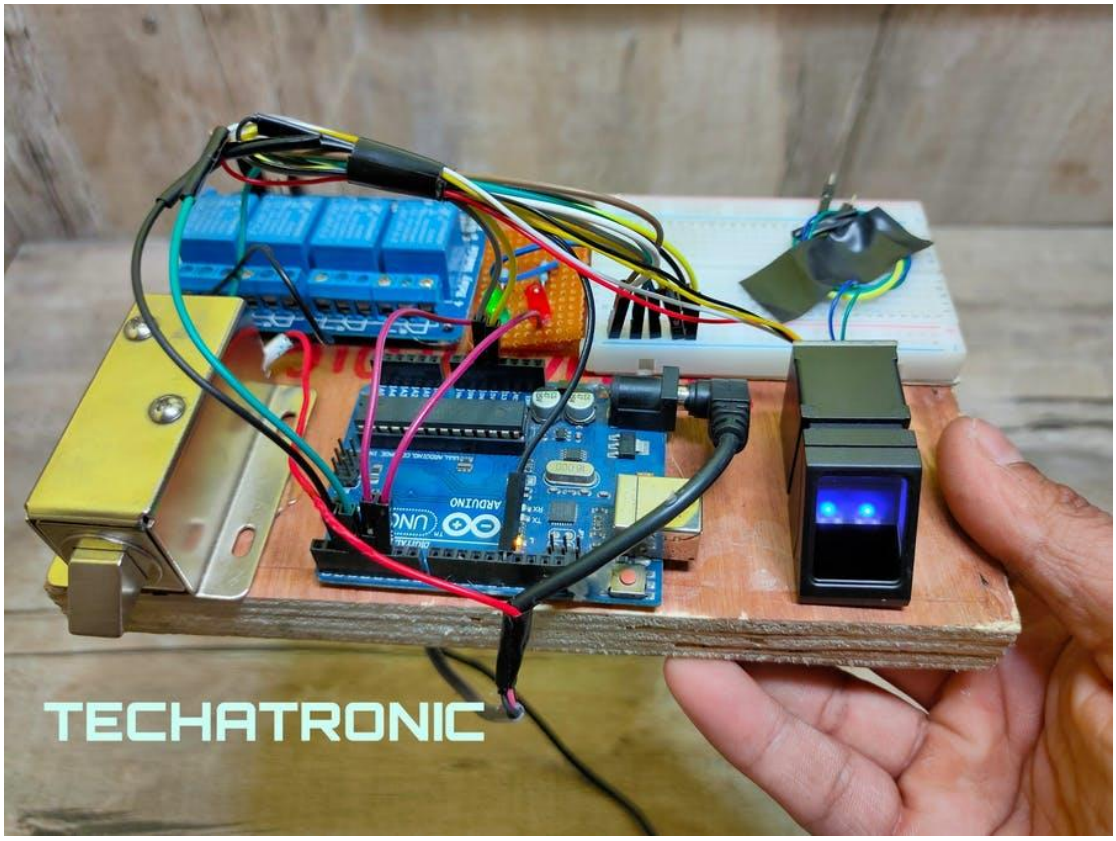


Figure 3: System Figure

Pros:

- Real-time notifications via IoT mobile platform.
- MPU5060 fall detection sensor is attached for precise detection.

Cons:

- No PCB is manufactured.
- Access security is only supported.

Reference:

https://create.arduino.cc/projecthub/electronicprojects/arduino-fingerprint-security-lock-c4e5b3?ref=user&ref_id=1012943&offset=0

2.3.3 Third System: IoT Wireless Smart Home Security System.

Illustrative Image:

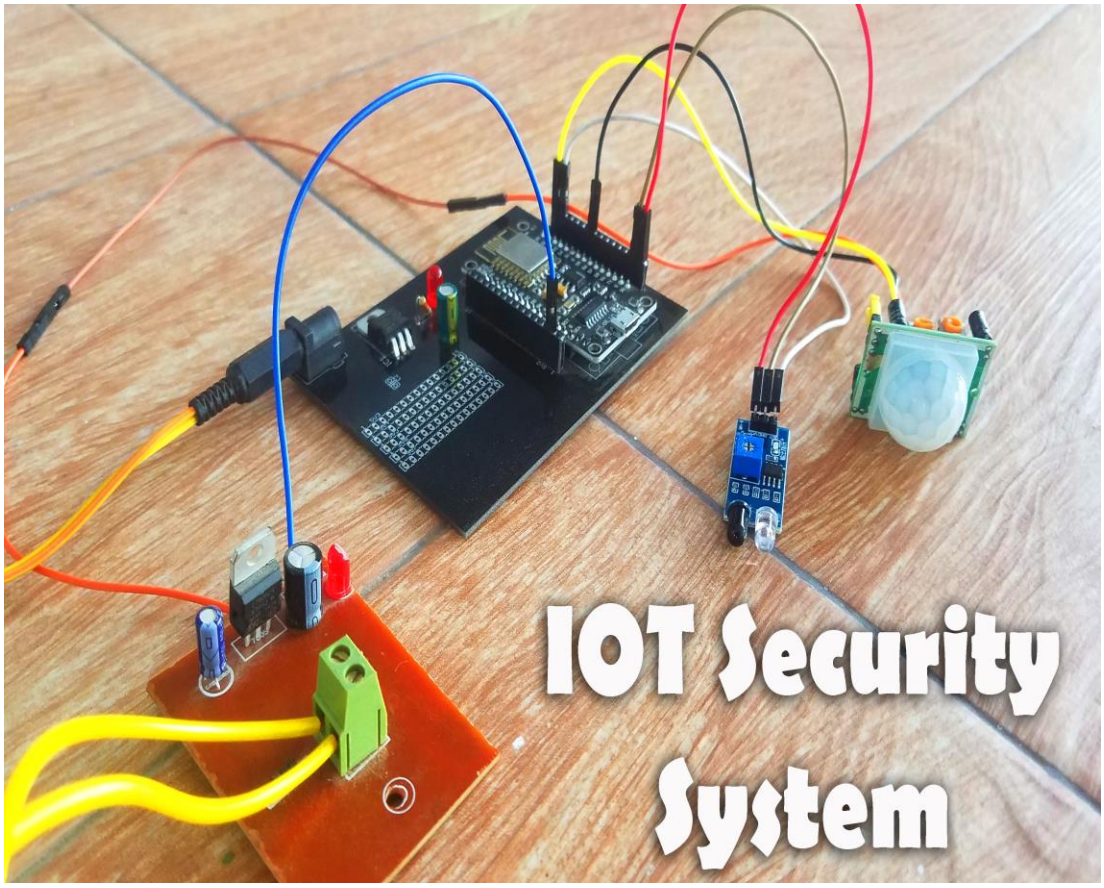


Figure 4: System Figure

Pros:

- Wi-Fi communication technology is supported.
- Easy to use and deploy.

Cons:

- Not all sensors and detectors are supported.
- No alarm siren is attached.

Reference:

https://www.electroniclinic.com/iot-security-system-iot-home-alarm-iot-home-security-system-project/

Chapter 3:

3. Requirements & Analysis:

3.1 Project Lifecycle Model:

Iterative Model.

Iterative software development is a software development process that is performed in small steps, during which the obtained intermediate results are analysed, new requirements are set, and the previous work stages are corrected.

Each of the iterations includes all software development processes: requirements acquisition and analysis, specification preparation, the implementation itself, testing, and launch. However, within a single iteration, only a separate component or version is developed, but not the entire project.

The next iteration results either in a new functionality or an improved existing functionality of the product. The full set of requirements fixed by the project boundaries becomes implemented after the final iteration is complete.

Advantages:

- **Quick project launch.** You start your project in a shorter time (even if it does not feature full functionality), and you are already earning money on it.
- **Risk reduction.** Issues are identified and resolved during iterations.
- **Flexibility to modifications.** If at a certain stage you understand that a particular function has become a priority, you can start implementing it in the next iteration without waiting for the entire project to be finished.
- **Regular release of new versions.** Using the iterative development approach, new versions are released regularly and the project is constantly advancing.
- **Efficient feedback.** Development teams actively communicate with customers, creating a product that meets their needs and business goals.
- **Prompt release of MVP.** This model allows bringing the product to the market and starting its use much earlier than in the case of a waterfall model.
- **Higher quality.** An iterative approach allows for creating a more robust architecture since all errors are fixed during several iterations.

Disadvantages:

- **No fixed budget or deadlines.** With an iterative approach, especially in the case of a complex project, deadlines and a budget depend on functional features and may change throughout the development process.
- **Strong customer involvement in the process.** Customers have to actively join the work on the project and discuss and approve modifications to the project. Some customers may feel uncomfortable about it, and the habitual waterfall development model will probably work better for them.
- **Possible problems with the architecture.** With no strict requirements and well-developed global plan, the software product architecture may suffer, and to bring it back to a reasonable condition, you may need additional resources.

Illustrative Figure:

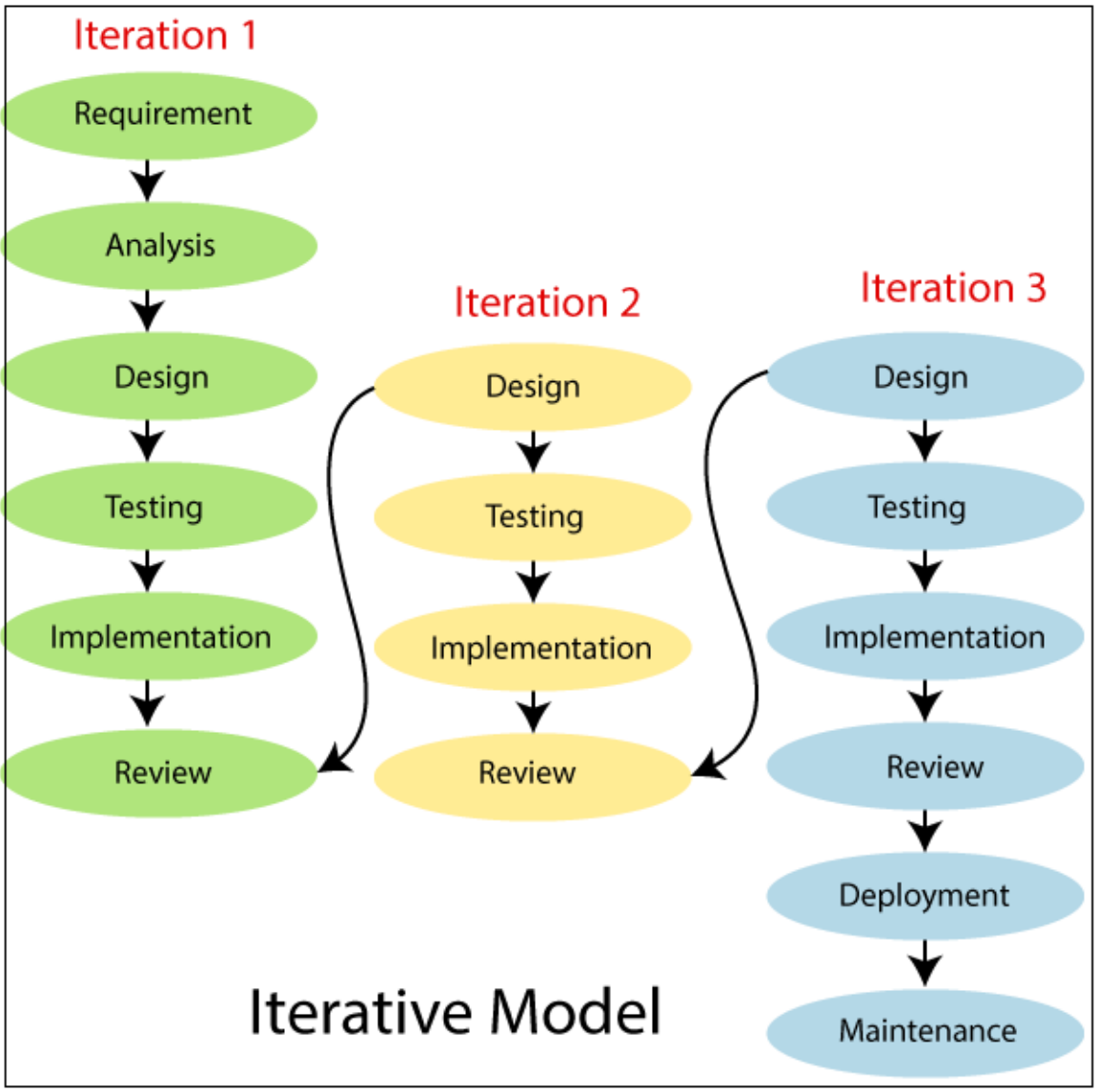


Figure 5: Iterative Model

3.2 Project Requirements:

Here are complete lists for my project requirements specification.

3.2.1 Functional Requirements:

- Connecting to Internet.

- Disconnecting from Internet.
- Connecting to Blynk IoT Cloud server.
- Disconnecting from Blynk IoT Cloud server.
- Reconnecting to Blynk IoT Cloud server.
- Read IR sensor.
- Read smoke sensor.
- Read flame sensor.
- Read motion sensor.
- Calibrate read data from sensors.
- Send data to software system.
- Visualize incoming data.
- Send mobile notifications.
- Send email notifications.
- Create user account.
- Login with existent user account.
- Reset user account password.
- Receive authentication codes.

3.2.2 Non-Functional Requirements:

- Connection to Internet should not exceed 10 seconds.
- Disconnection from Internet should be checked every 5 seconds.

- Connection to Blynk IoT server should not exceed 15 seconds.
- Disconnection from Blynk IoT server should be ignored for just 5 seconds.
- Real-time communication should be maintained.
- Timed data sending should be maintained.
- Each sensor data should be calibrated before sending.
- Mean time between failures should not exceed 2 in a month.
- Mean time to failures should not exceed 1 in a week.
- Mean time to repair should not exceed a quarter hour.
- Periodic maintenance should be performed every month.
- User email should be restricted on alphanumeric, hyphens and dots only.
- User password length should be between 6 and 15.
- User password should not contain spaces.
- User password can be uncovered by only the entering user.

3.2.3 Hardware Requirements:

- ATMEL AVR ATmega328P Microcontroller.
- GSM SIM800L SIM Module.
- Printed Circuit Board “PCB”.
- Motion Sensor.
- Smoke Sensor “MQ-2”.

- Flame Sensor.
- IR Sensor.
- DC Socket.
- Relay Module “Solid-State Relay”.
- Siren.
- Diodes.
- Resistors.
- Capacitors.
- USB-To-Serial IC “CH340”.
- Voltage Regulator “7805”.
- LEDs.

3.2.4 Hardware Tools :

- OrCad
- Arduino
- Arduino-C
- Blynk Application

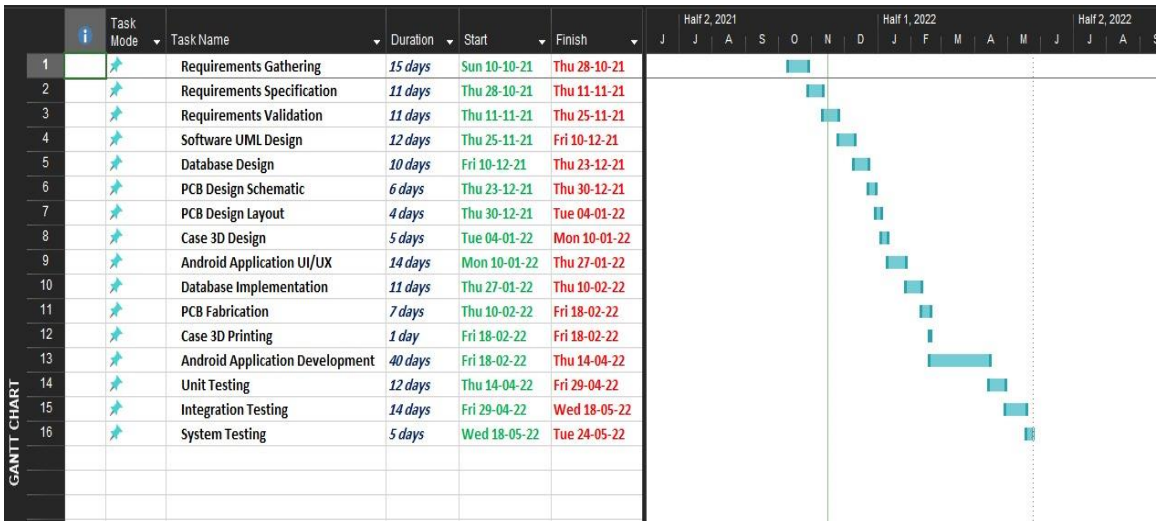
3.2.5 Software Requirements:

- IoT Blynk Console.
- Cadence OrCAD
- Altium Designer.
- GerbTool.

3.2.4 Software Tools (for diagrams) :

- Microsoft Office Project
- Office Visio

3.2.6 GanttChart:



	Task Mode	Task Name	Duration	Start	Finish
1		Requirements Gathering	15 days	Sun 10-10-21	Thu 28-10-21
2		Requirements Specification	11 days	Thu 28-10-21	Thu 11-11-21
3		Requirements Validation	11 days	Thu 11-11-21	Thu 25-11-21
4		Software UML Design	12 days	Thu 25-11-21	Fri 10-12-21
5		Database Design	10 days	Fri 10-12-21	Thu 23-12-21
6		PCB Design Schematic	6 days	Thu 23-12-21	Thu 30-12-21
7		PCB Design Layout	4 days	Thu 30-12-21	Tue 04-01-22
8		Case 3D Design	5 days	Tue 04-01-22	Mon 10-01-22
9		Android Application UI/UX	14 days	Mon 10-01-22	Thu 27-01-22
10		Database Implementation	11 days	Thu 27-01-22	Thu 10-02-22
11		PCB Fabrication	7 days	Thu 10-02-22	Fri 18-02-22
12		Case 3D Printing	1 day	Fri 18-02-22	Fri 18-02-22
13		Android Application Development	40 days	Fri 18-02-22	Thu 14-04-22
14		Unit Testing	12 days	Thu 14-04-22	Fri 29-04-22
15		Integration Testing	14 days	Fri 29-04-22	Wed 18-05-22
16		System Testing	5 days	Wed 18-05-22	Tue 24-05-22

3.3 Use Case Scenarios:

Parameter	Description
Function Name:	Connect to Internet.
Precondition:	Router or access point is enabled.
Trigger	System
Normal Flow	1. Enter router or access point SSID and password. 2. Assure that credentials pass router registration. 3. Connect to access point.
Postcondition:	System is online.

Parameter	Description
Function Name:	Connect to Blynk Server.
Precondition:	System must be connected to Internet.
Trigger	System
Normal Flow	1. Validate Internet connectivity. 2. Establish connection to MQTT server. 3. Authenticate with MQTT credentials. 4. Connect to Blynk dashboard.
Postcondition:	System is connected

3.4 Use Case Diagram:

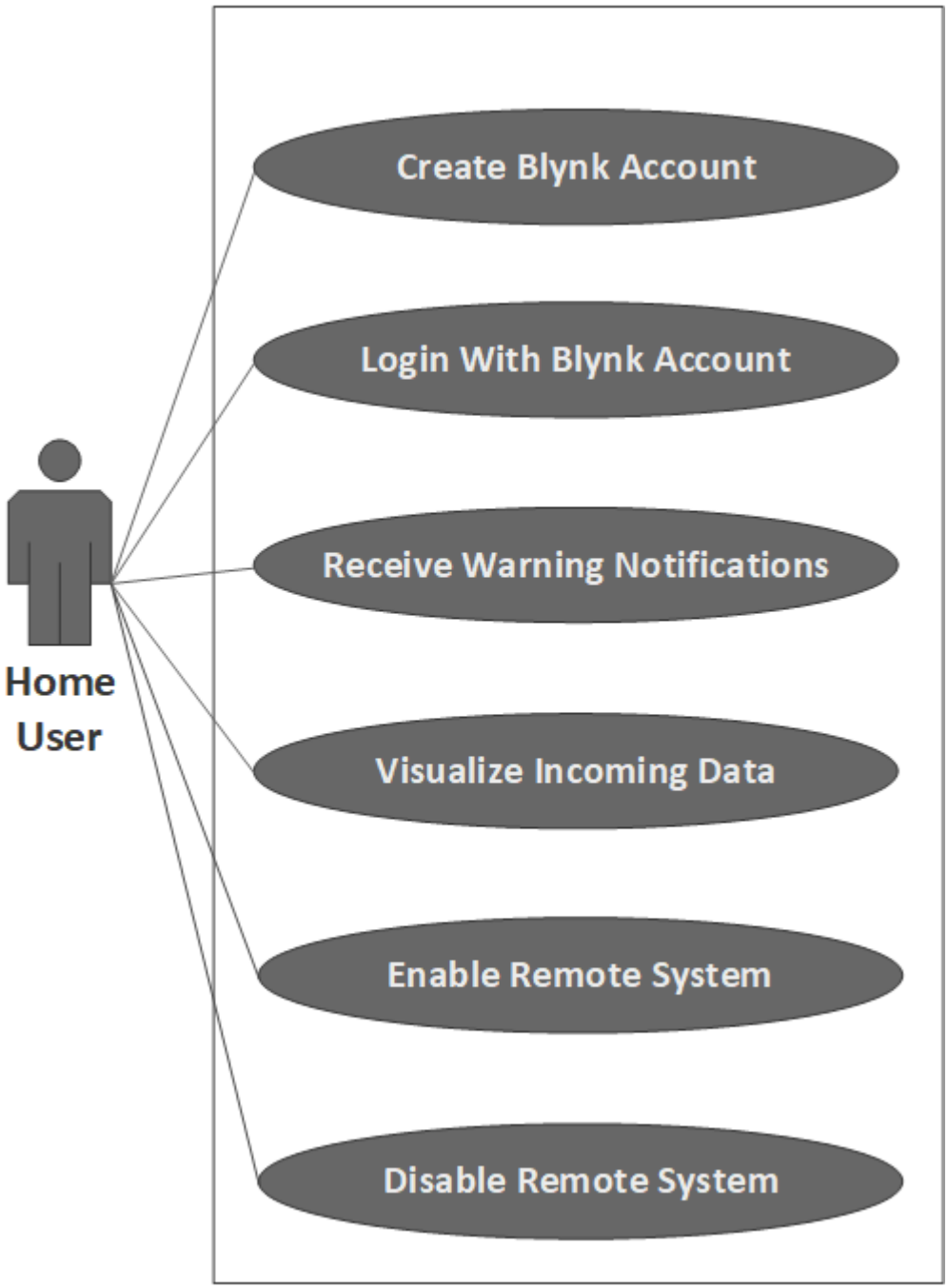


Figure 6: Use Case Diagram

3.5 Data Flow Diagram:

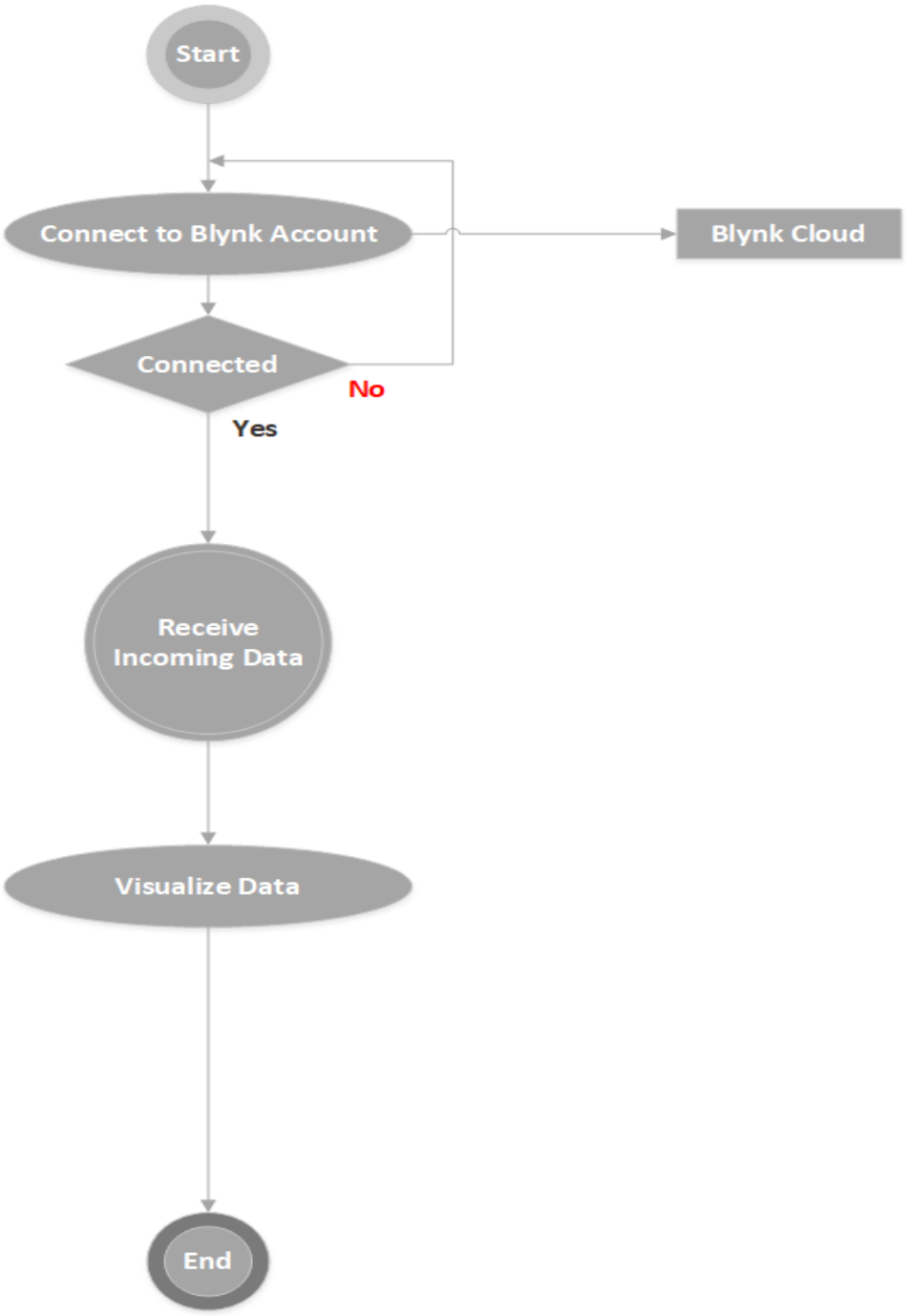


Figure 7: Data Flow Diagram

Chapter 4:

4. Design, Implementation & Testing:

4.1. Design:

4.1.1. System Model:

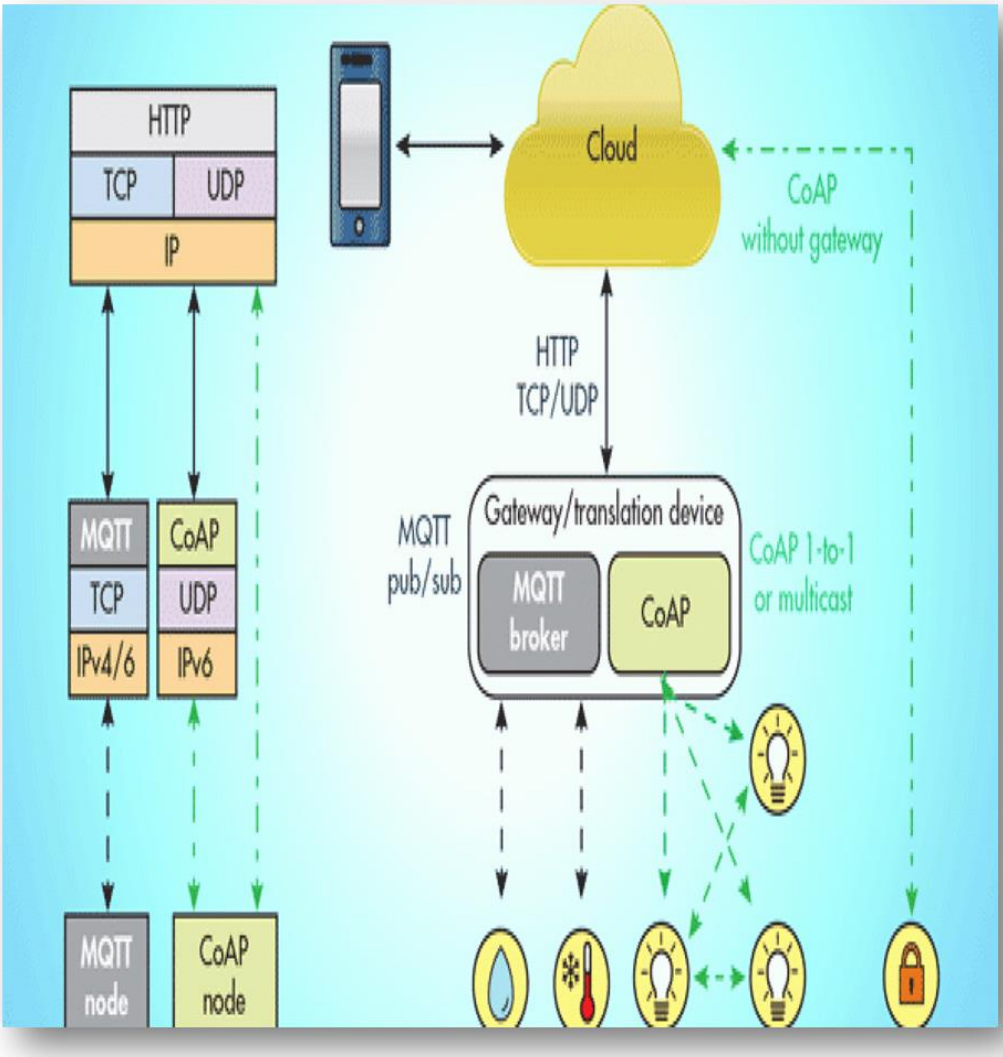


Figure 8: System Model

4.1.2. Main Approach:

Lean Software Development (Agile Methodology).

Description:

Lean Software Development (LSD) is an agile framework based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs. The Lean approach is also often referred to as the Minimum Viable Product (MVP) strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don't like and want to be added, and then iterates based on this feedback.

LSD actually borrows its philosophy from the manufacturing industry, which originated the lean development process as a way to optimize production and assembly lines to minimize waste and maximize customer value. In fact, it was originally called the Toyota Production System, because automaker Toyota invented this approach in the middle of the twentieth century as a way to streamline its production of cars and eliminate wasted time and resources. (Any action that did not impact the functionality of the car being built and delivered was considered a waste under this system, and therefore removed from the process.)

LSD depends mainly on the following principles:

1. **Eliminate Waste:**

One of the key elements of practicing Lean is to eliminate anything that does not add value to the customer.

2. **Build Quality In:**

It might seem self-evident -- every team wants to build quality into their work. But unless this is part of a disciplined practice, it's far easier said than done.

3. **Create Knowledge:**

The Lean development principle of Create Knowledge is another one that seems simple, but requires discipline and focus to implement. This principle encourages

Lean teams to provide the infrastructure to properly document and retain valuable learning.

4. Defer Commitment:

This Lean principle encourages team to demonstrate responsibility by keeping their options open and continuously collecting information, rather than making decisions without the necessary data.

Advantages:

- Streamlined approach allows more functionality to be delivered in less time.
- Eliminates unnecessary activity, and as a result can reduce costs.
- Empowers the development team to make decisions, which can also boost morale.

Disadvantages:

- Heavily depends on the team involved, making it not as scalable as other frameworks.
- Depends on strong documentation, and failure to do so can result in development mistakes.

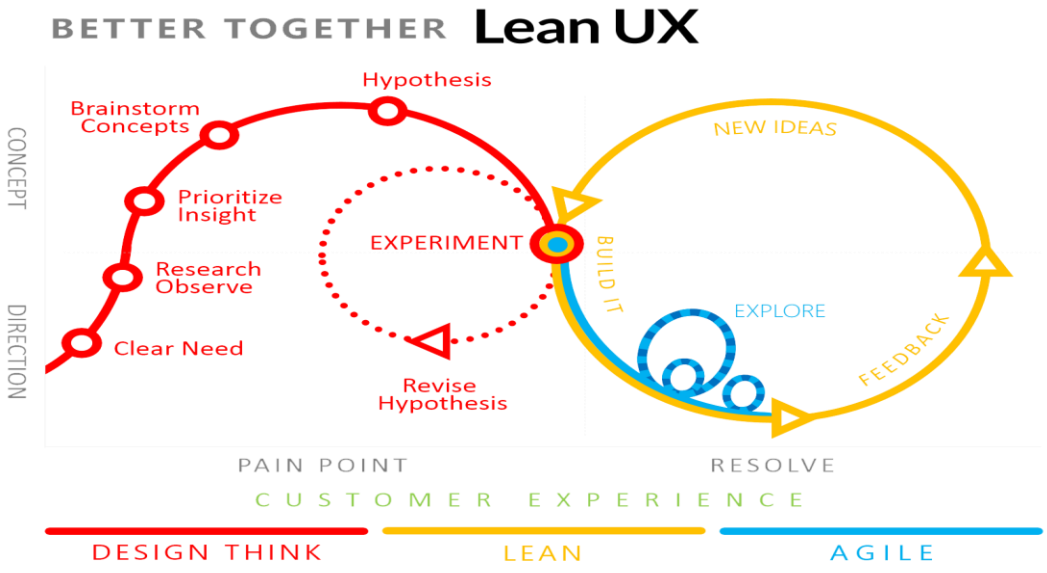


Figure 9: LSD Software Approach

Adoption Justification:

This project required deep design thinking sessions with expertise personnel in the field of shops security and physical safety. This model helped me clarify exact needs to perfectly tackle and accomplish, including project rapid delivery.

4.1.3. Alternative Model:

Future Driven Development (Agile Methodology).

Description:

Feature Driven Development (FDD) is an agile framework that, as its name suggests, organizes software development around making progress on features. Features in the FDD context, though, are not necessarily product features in the commonly understood sense. They are, rather, more akin to user stories in Scrum. In other words, "complete the login process" might be considered a feature in the Feature Driven Development (FDD) methodology.

The first real-world application of the Feature Driven Development methodology was on a 50-person software-development project for a Singapore-based financial institution, and the first public discussion of the methodology was in the 1999 book Java Modelling in Colour with UML.

FDD was designed to follow a five-step development process, built largely around discrete "feature" projects. That project lifecycle looks like this:

- Develop an overall model.
- Build a features list.
- Plan by feature.
- Design by feature.
- Build by feature.

Advantages:

- Simple five-step process allows for more rapid development.
- Allows larger teams to move products forward with continuous success.
- Leverages pre-defined development standards, so teams are able to move quickly.

Disadvantages:

- Does not work efficiently for smaller projects.
- Less written documentation, which can lead to confusion.
- Highly dependent on lead developers or programmers.

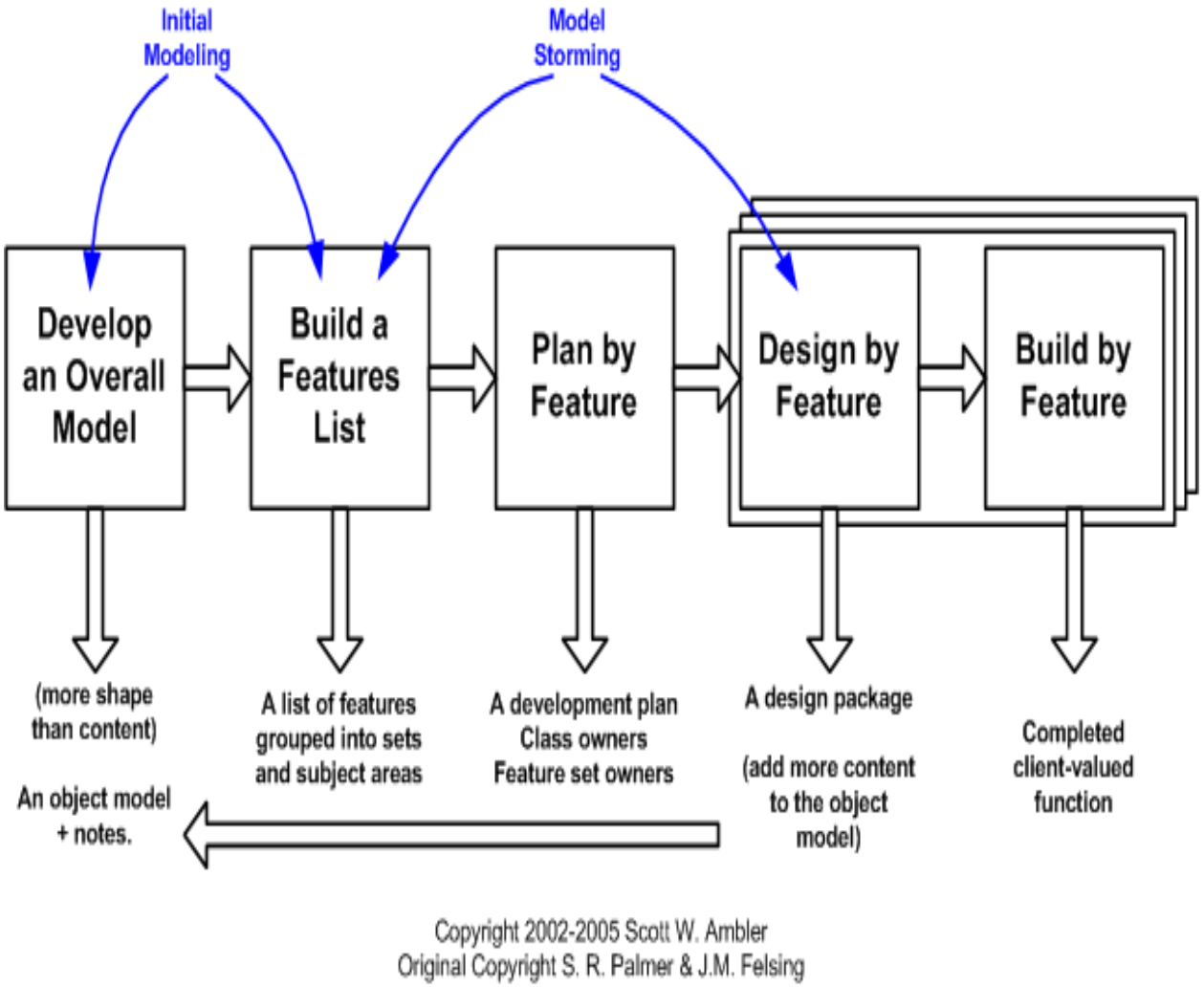
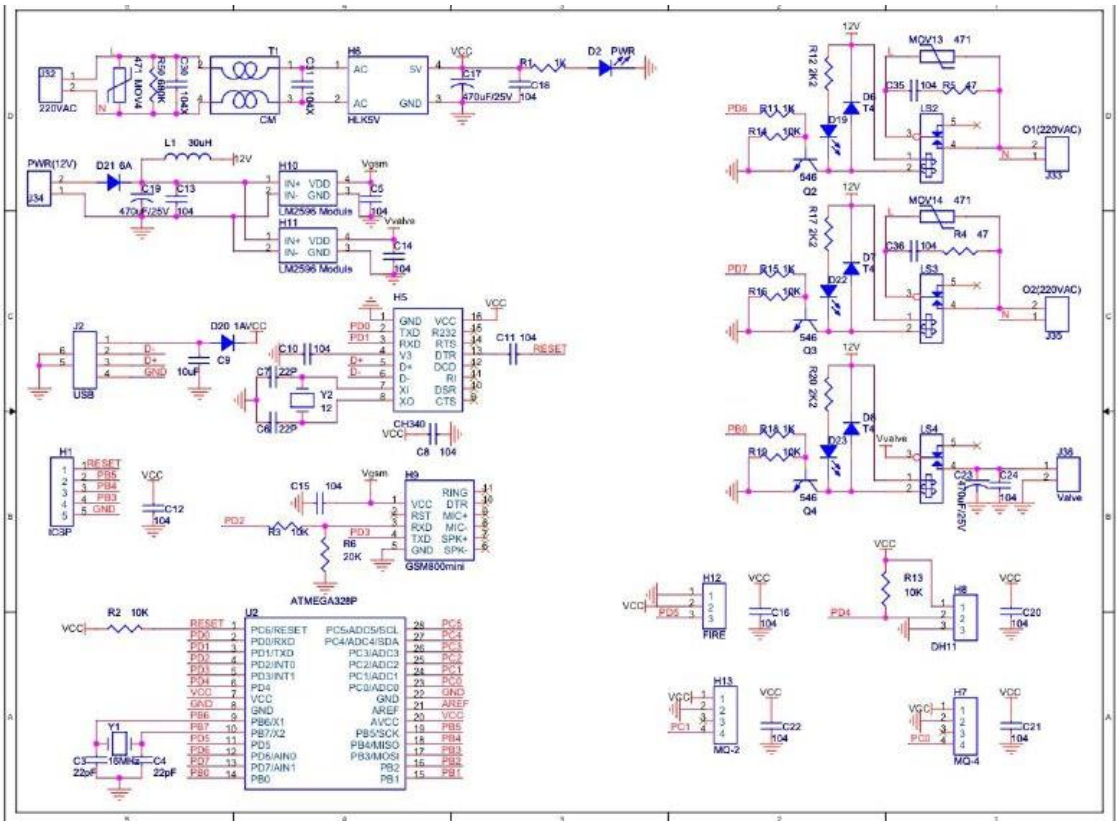


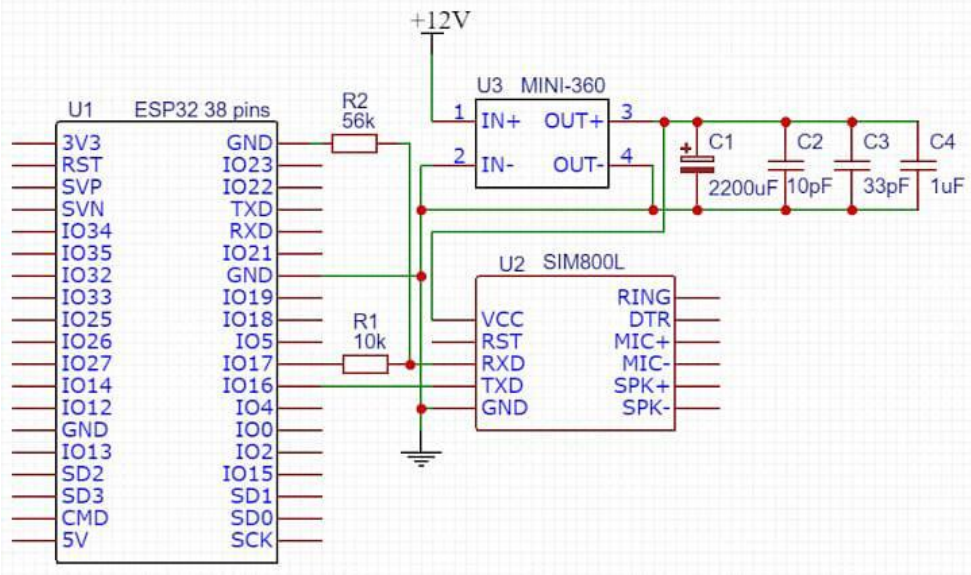
Figure 10: FDD Approach

Adoption Justification:

My project actively depends on adding new features during the whole period of project implementation. This model rapidly helped me manage the process of adding new features by means of applying its simple five steps procedure.

4.1.4. PCB Design Schematic:





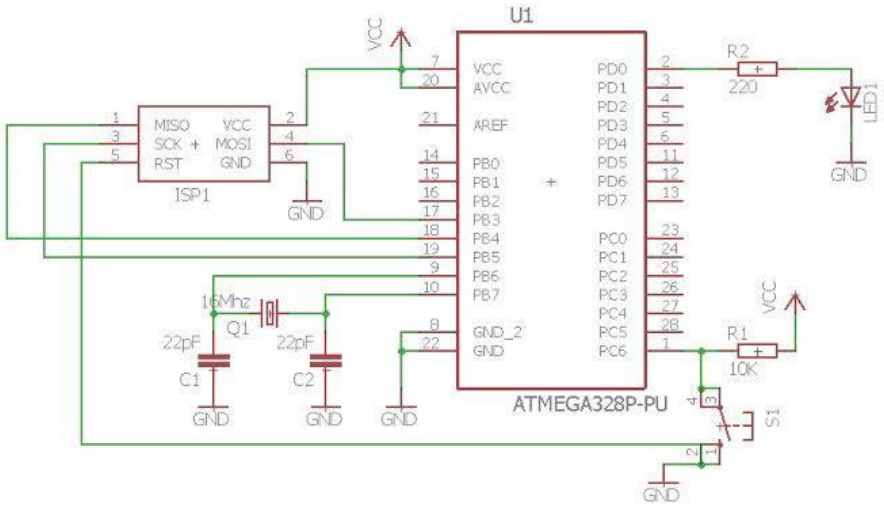


Figure 11: PCB Design Schematic

4.1.5. PCB Design Layout:

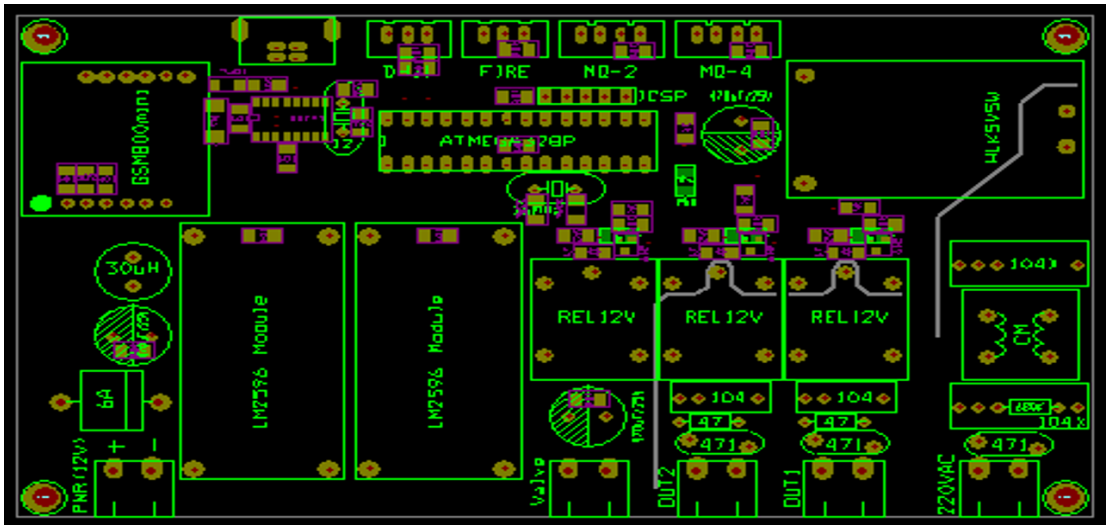


Figure 12: PCB Design Layout

3.6 Implementation:

PCB (Top Layer):

4.2.1. PCB (Bottom Layer):

4.2.2. Arduino-C Circuit 1 Code:

Firmware

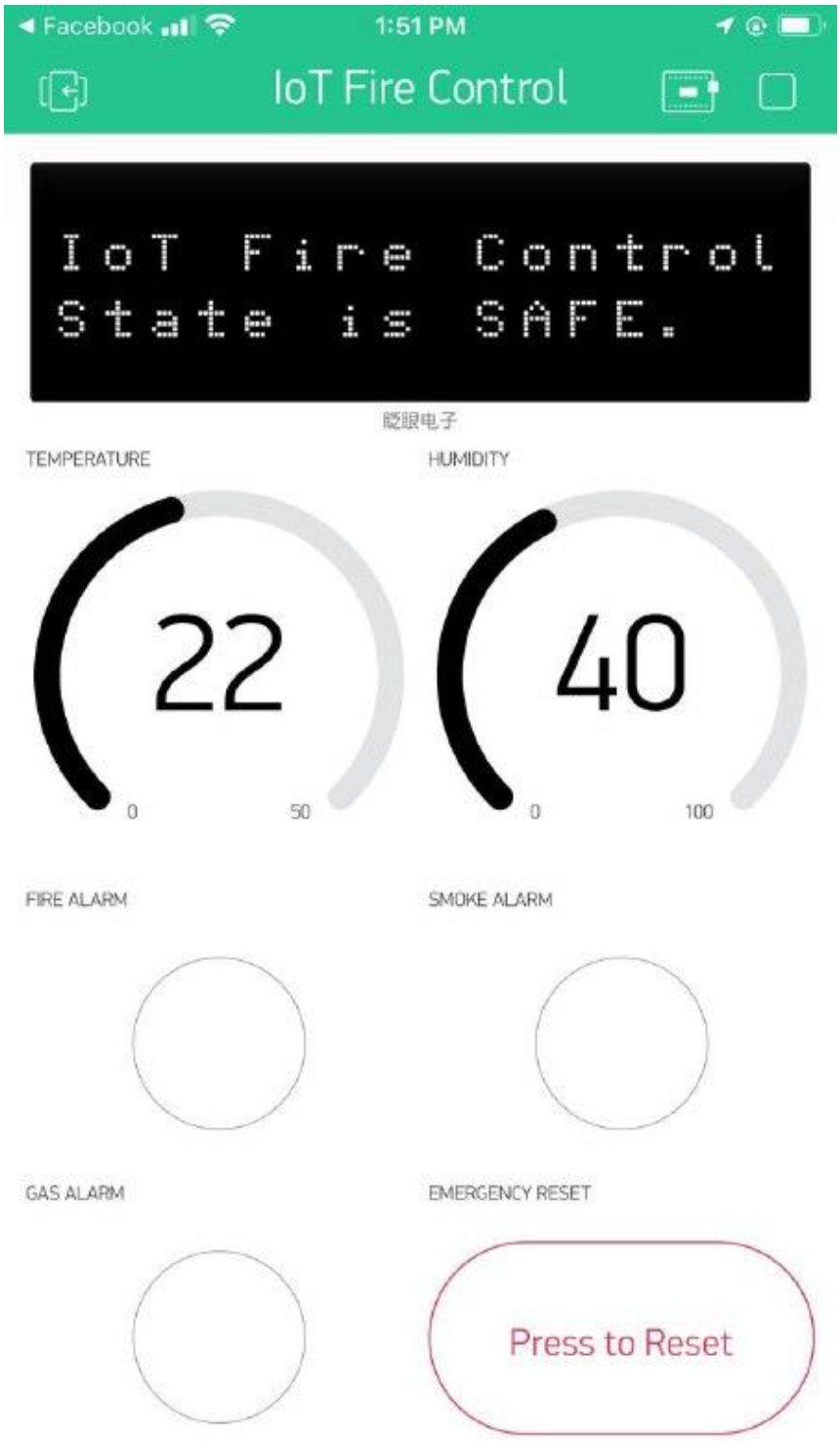
```
#define BLYNK_PRINT Serial
#define TINY_GSM_MODEM_SIM800
#define BLYNK_HEARTBEAT 8

#include <SoftwareSerial.h>
#include <TinyGsmClient.h>
#include <BlynkSimpleTinyGSM.h>
#include <DHT.h>

char auth[] = "ke_-uSE5Xvw7YTAlHe2WZv6tKW8g7nMS";
char apn[]  = "orangeinternet";
char user[] = "";
char pass[] = "";

#define temperatureHumiditySensor 4
#define fireSensor 5
#define smokeSensor A1
```

4.2.3. Arduino-C Circuit 2 Code:

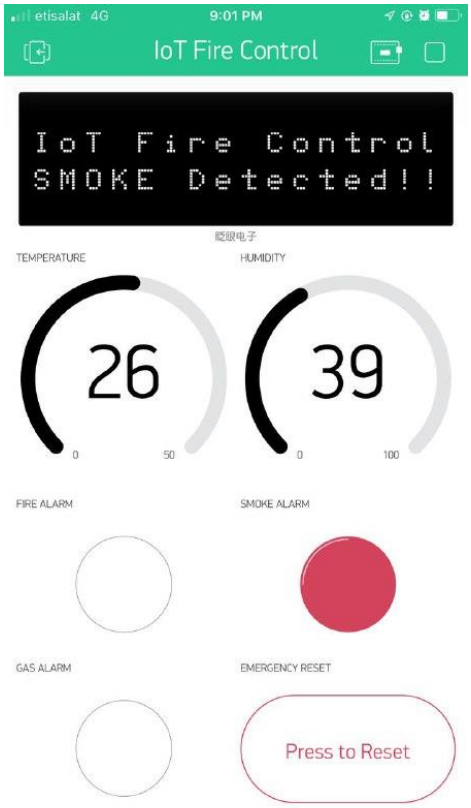
```
pinMode(fireSensor, INPUT);
pinMode(gasValve, OUTPUT);
pinMode(acDevice1, OUTPUT);
pinMode(siren, OUTPUT);
digitalWrite(gasValve, HIGH);
digitalWrite(acDevice1, HIGH);
digitalWrite(siren, LOW);
Serial.begin(115200);
delay(10);
SerialAT.begin(9600);
delay(3000);
modem.restart();
Blynk.begin(auth, modem, apn, user, pass);
dht.begin();
timer.setInterval(4000L, readTemperatureHumidity);
Blynk.notify("System Operated ->");
```

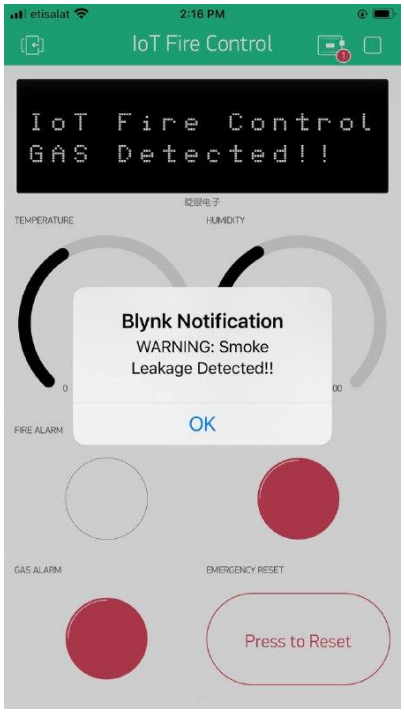
```
fireValue = digitalRead(fireSensor);
if (fireValue == 1)
{
  safetyStatus = true;
  resetStatus = true;
  fireLED.on();
  Blynk.notify("WARNING: Fire Incident Detected!!");
  Blynk.email("IoT Fire Control", "WARNING: Fire Incident Detected!!");
  digitalWrite(gasValve, LOW);
  digitalWrite(acDevice1, LOW);
  digitalWrite(siren, HIGH);
  stateLCD.clear();
  stateLCD.print(0, 0, "IoT Fire Control");
  stateLCD.print(0, 1, "FIRE Detected!!");
}

smokeValue = analogRead(smokeSensor);
```

4.2.4. IoT Blynk Application:

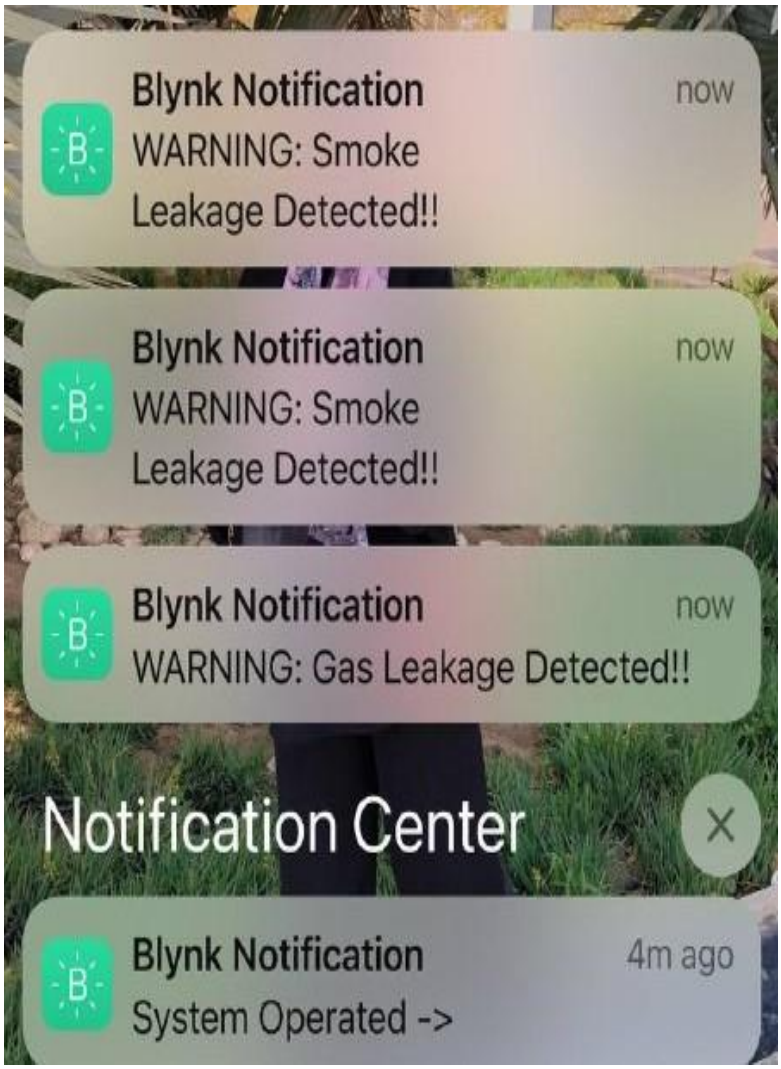


4.2.5. IoT Blynk Application Configuration:

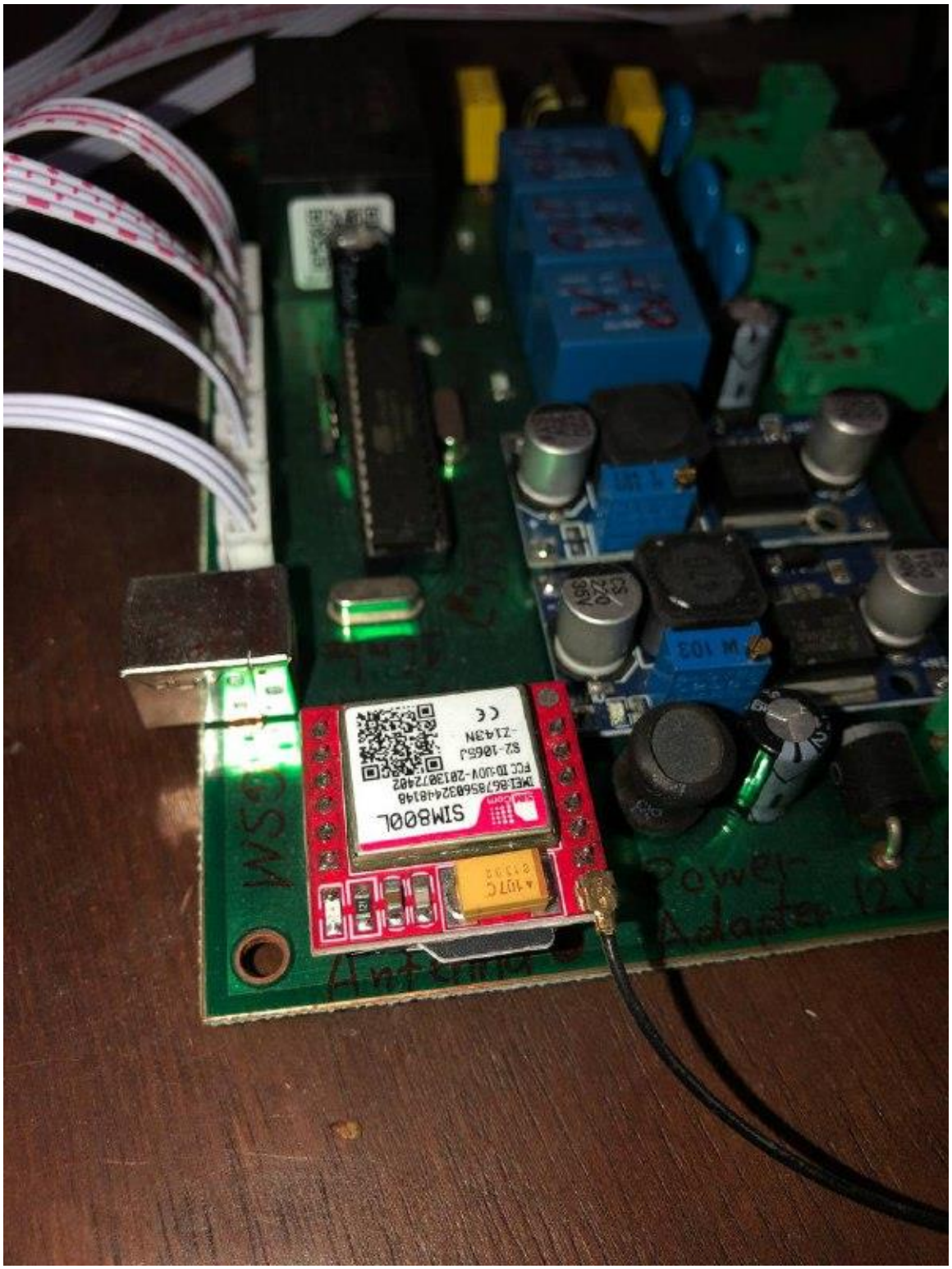




Mobile Notification Warning:

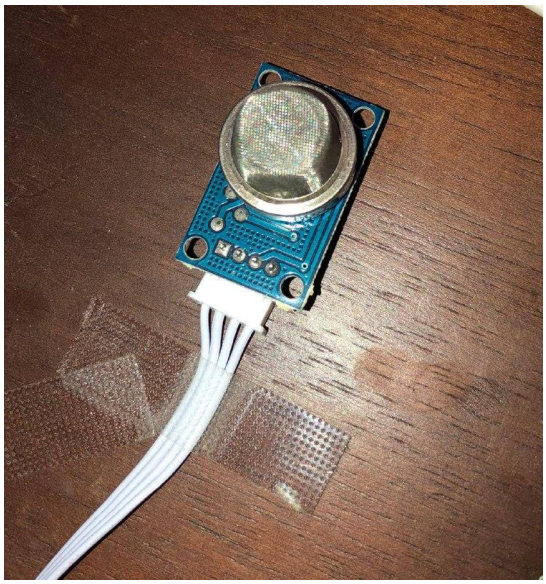


Gmail Notifications Warning:

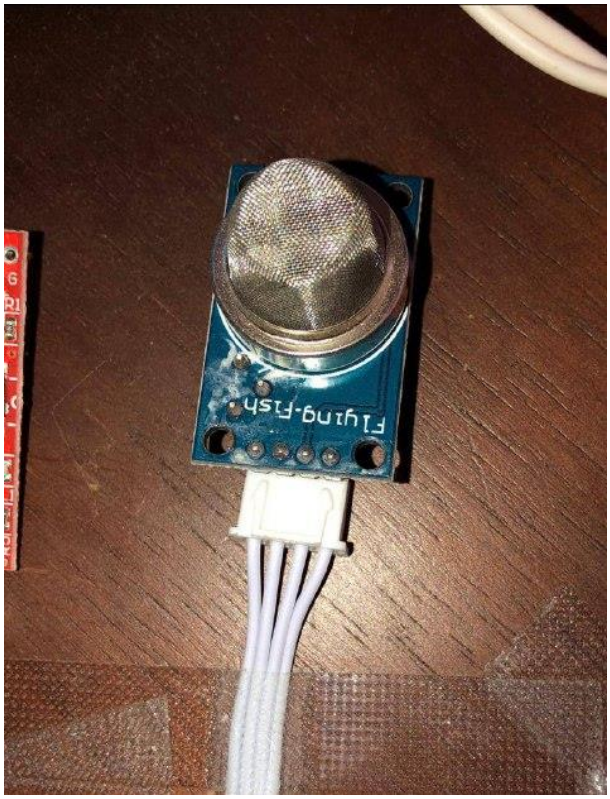
- **Knowledge of main components:**
- **GSM (SIM 800 L) with Orange Internet:**
- 

- **Sensors:**

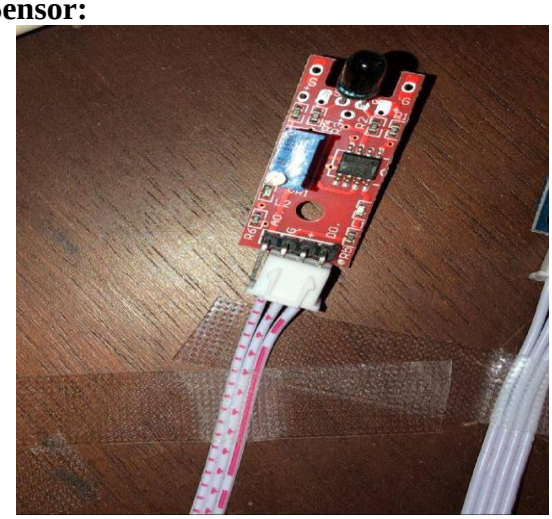
  Gas Sensor:



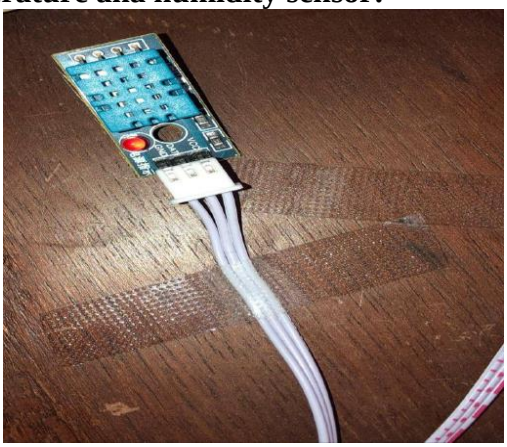
- **Smoke sensor:**



- **Fire Sensor:**



- **Temperature ana humidity sensor:**

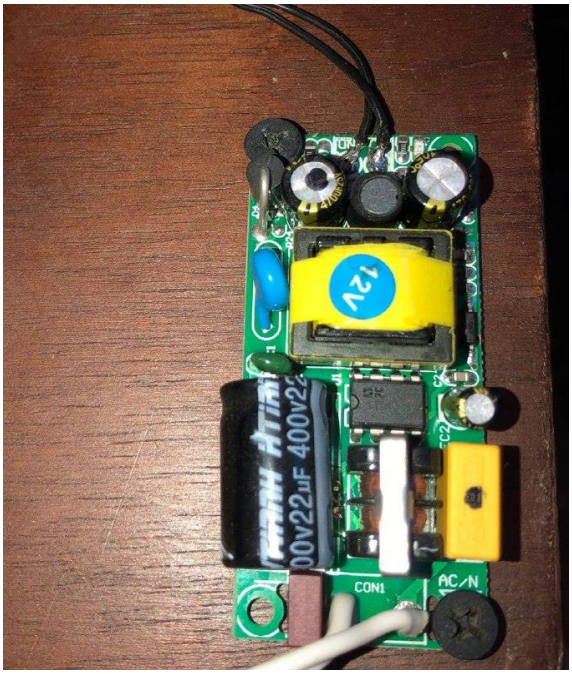


- **Socket:**

To prove disconnecting of the power or electricity when a danger happen like Gas leak or the temperature or humidity is higher than normal or smoke leak higher than normal or fire.

- **Module:**

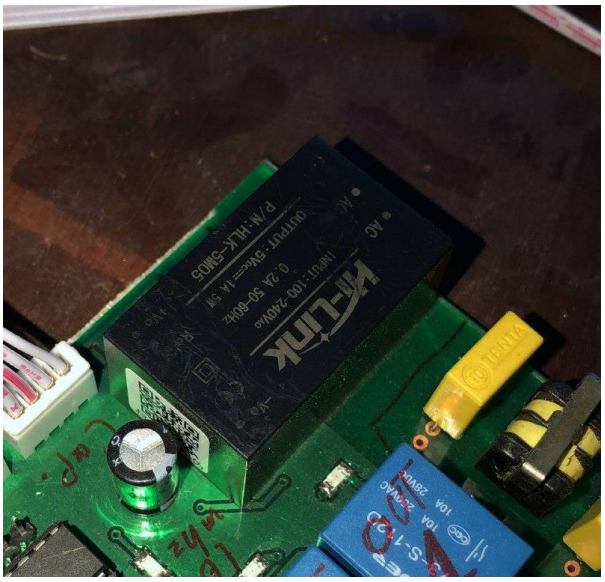
Work to supply the siren alarm with the necessary electricity.



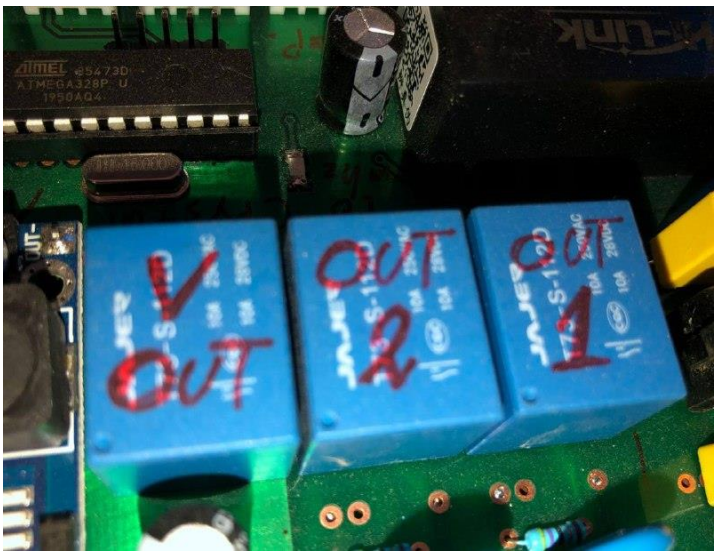
- **Gas Valve:**

Works on control the opening and closing of gas automatically by relay V out.

- **Switch mode power supply:**
  To Save and keep PCB from heating up.



- **Relays :**
  V Out : To control Gas Valve .
  OUT 2 : For siren alarm.
  OUT 1: For Electricity.



3.7 Testing:

4.2.6. Embedded System Testing:

Embedded systems are the electronically controlled devices where software and hardware are tightly coupled. Embedded systems may contain a variety of computing devices. These are PCs incorporated in other devices to operate application-specific functions. The end user usually is not even aware of their existence.

Embedded Testing is checking the functional and non-functional attributes of both software and hardware in an embedded system. The purpose of Embedded test is to verify and validate the Embedded software as well as hardware against client requirement.

Embedded Software testing checks and ensure the concerned software is of good quality and complies with all the requirements it should meet. Embedded software testing is an excellent approach to guarantee security in critical applications like medical equipment, railways, aviation, vehicle industry, etc. Strict and careful testing is crucial to grant software certification.

In general, we test for four reasons:

- **To find bugs in software.**
- **Helps to reduce risk to both users and the company.**
- **Cut down development and maintenance costs.**
- **To improve performance.**

The activities to performed are:

1. **The software is provided with some inputs.**
2. **A Piece of the software is executed.**
3. **The software state is observed, and the outputs are checked for expected properties like whether the output matches the expected outcome, conformance to the requirements and absence of system crashes.**

The testing types are:
Fundamentally, there are five levels of testing that can be applied to embedded software.

Software Unit Testing:
The unit module is either a function or class. Unit Testing is performed by the development team, primarily the developer and is usually carried out in a peer-review model. Based on the specification of the module test cases are developed.

Integration Testing:
Integration testing can be classified into two segments:

1. **Software integration testing.**
2. **Software/hardware integration testing.**

In the end, the interaction of the hardware domain and software components are tested. This can incorporate examining the interaction between built-in peripheral devices and software.

Embedded software development has a unique characteristic which focuses on the actual environment, in which the software is run, is generally created in parallel with the software. This causes inconvenience for testing since comprehensive testing cannot be performed in a simulated condition.

System Unit Testing:
Now the module to be tested is a full framework that consists of complete software code additionally all real-time operating system (RTOS) and platform-related pieces such as interrupts, tasking mechanisms, communications and so on. The Point of Control protocol is not anymore, a call to a function or a method invocation, but rather a message sent/got utilizing the RTOS message queues.

System resources are observed to evaluate the system's ability to support embedded system execution. For this aspect, gray-box testing is the favoured testing method. Depending on the organization, system unit testing is either the duty of the developer or a dedicated system integration team.

System Integration Testing:

The module to be tested begins from a set of components within a single node. The Points of Control and Observations (PCOs) are a mix of network related communication protocols and RTOS, such as network messages and RTOS events. Additionally, to a component, a Virtual Tester can likewise play the role of a node.

System Validation Testing:

The module to be tested is a subsystem with a complete implementation or the complete embedded system. The objective of this final test is to meet external entity functional requirements. Note that an external entity either be a person, or a device in a telecom network, or both.

Embedded Software Testing Challenges:

Some of the challenges that I have faced during Embedded software testing:

Hardware Dependency:

Hardware dependency is among the main difficulties faced during embedded software testing because of limited access to hardware. However, Emulators and Simulators may not precisely represent the behaviour of the actual device and could give a wrong sense of system performance and application's usability.

Open-Source Software:

The majority of the embedded software components are open source in nature, not created in-house and absence of complete test available for it. There is a wide range of test combinations and resulting scenarios.

Software vs. Hardware Defects:

Another aspect is when software is being developed for a freshly created hardware, during this process high ratio of hardware defects can be identified. The found defect is just not limited to software. It may be related to hardware also.

Reproducible Defects:

Defects are harder to reproduce/recreate in the case of the embedded system. That enforces the embedded testing procedure to value every defect occurrence substantially higher than in a standard case, other than to gather as much data as could sensibly be required to alter the system to find the foundation of the defect.

Continuous Software Updates:

Embedded systems require regular software updates like the kernel upgrade, security fixes, different device drivers, etc. Constraints identified with the software updates influence makes bug identification difficult. Additionally, it increases the significance of build and deployment procedure.

Finally, there are some difficulties in testing embedded software testing that makes it more difficult than regular software testing. The most fundamental issue is the tight reliance on the hardware environment that is prepared simultaneously with the software, and that is regularly required to perform reliable software testing. Sometimes it is even difficult to test the software without custom tools, which effortlessly makes concentrating on testing in late stages exceptionally enticing. One of the most important things is that you should think about is the fact that you should often opt for automated software testing. The embedded automated testing is a quicker process which would take some hours to complete, and in this way, the issue of your software is settled.

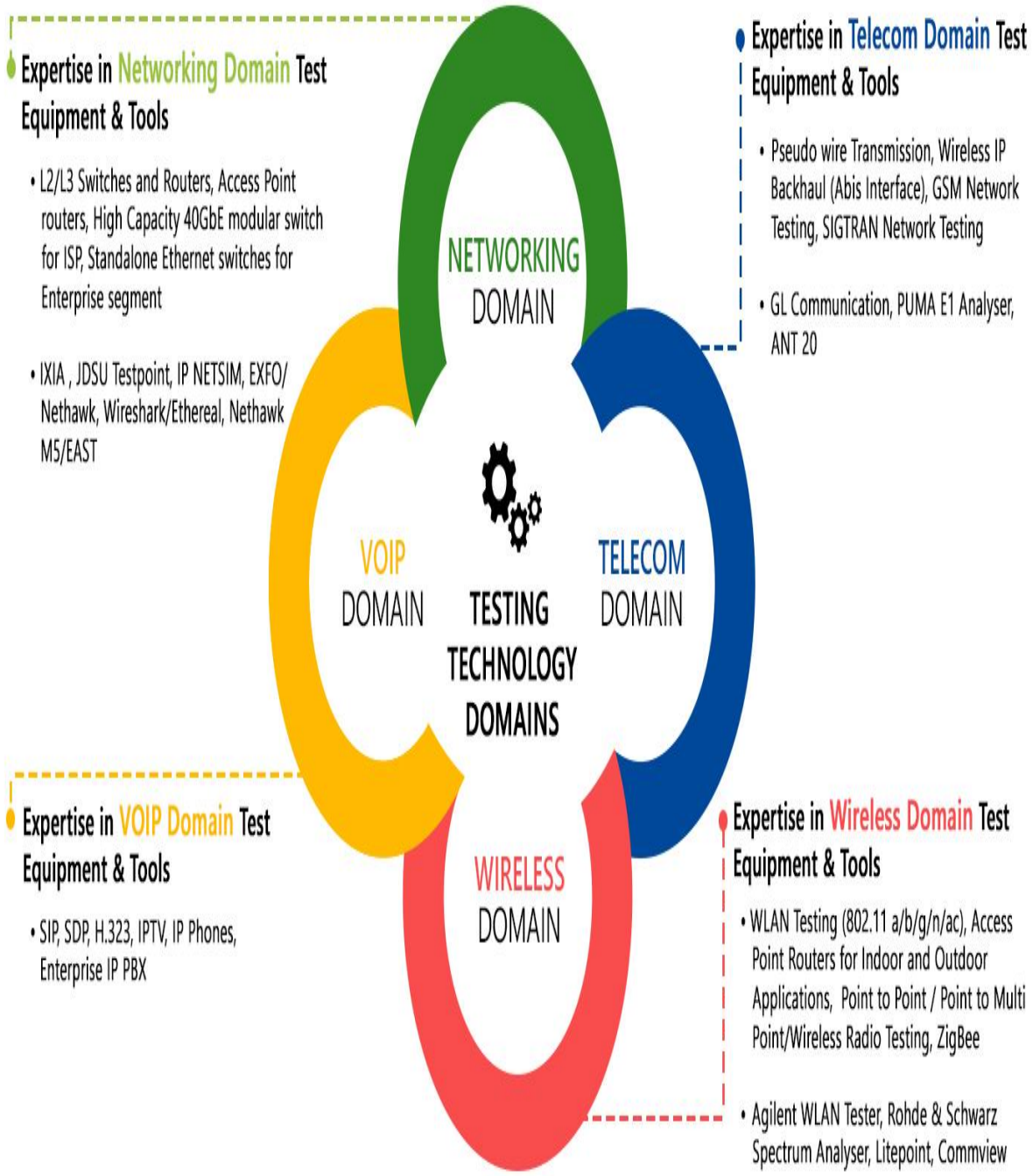


Figure 13: Testing Technology Domains

Code :

```
#define BLYNK_PRINT Serial
#define TINY_GSM_MODEM_SIM800
#define BLYNK_HEARTBEAT 8

#include <SoftwareSerial.h>
#include <TinyGsmClient.h>
#include <BlynkSimpleTinyGSM.h>
#include <DHT.h>

char auth[] = "ke_-uSE5Xvw7YTAlHe2WZv6tKW8g7nMS";
char apn[]  = "orangeinternet";
char user[] = "";
char pass[] = "";

#define temperatureHumiditySensor 4
#define fireSensor 5
#define smokeSensor A1
#define gasSensor A0
#define gasValve 8
#define acDevice1 6
#define siren 7

#define DHTTYPE DHT11
SoftwareSerial SerialAT(3, 2); //RX, TX
TinyGsm modem(SerialAT);
DHT dht(temperatureHumiditySensor, DHTTYPE);

WidgetLCD stateLCD(V0);
WidgetLED fireLED(V3);
WidgetLED smokeLED(V4);
WidgetLED gasLED(V5);
BlynkTimer timer;

bool safetyStatus = false, resetStatus = false;
int fireValue, smokeValue, gasValue;

void setup()
{
  pinMode(fireSensor, INPUT);
  pinMode(gasValve, OUTPUT);
  pinMode(acDevice1, OUTPUT);
  pinMode(siren, OUTPUT);
  digitalWrite(gasValve, HIGH);
  digitalWrite(acDevice1, HIGH);
```

```
  digitalWrite(siren, LOW);
  Serial.begin(115200);
  delay(10);
  SerialAT.begin(9600);
  delay(3000);
  modem.restart();
  Blynk.begin(auth, modem, apn, user, pass);
  dht.begin();
  timer.setInterval(4000L, readTemperatureHumidity);
  Blynk.notify("System Operated ->");
  stateLCD.clear();
  stateLCD.print(0, 0, "IoT Fire Control");
  stateLCD.print(0, 1, "State is SAFE.");
  fireLED.off();
  smokeLED.off();
  gasLED.off();
}

void loop()
{
  Blynk.run();
  timer.run();
  readSensors();
}

void readTemperatureHumidity()
{
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  if (isnan(h) || isnan(t))
  {
    //Serial.println("Failed to read from DHT sensor!");
    return;
  }
  Blynk.virtualWrite(V1, t);
  Blynk.virtualWrite(V2, h);
}

void readSensors()
{
  fireValue = digitalRead(fireSensor);
  if (fireValue == 1)
  {
    safetyStatus = true;
    resetStatus = true;
    fireLED.on();
```

```
  Blynk.notify("WARNING: Fire Incident Detected!!");
  Blynk.email("IoT Fire Control", "WARNING: Fire Incident Detected!!");
  digitalWrite(gasValve, LOW);
  digitalWrite(acDevice1, LOW);
  digitalWrite(siren, HIGH);
  stateLCD.clear();
  stateLCD.print(0, 0, "IoT Fire Control");
  stateLCD.print(0, 1, "FIRE Detected!!");
 }

 smokeValue = analogRead(smokeSensor);
 Serial.println(smokeValue);
 if (smokeValue > 400)
 {
  safetyStatus = true;
  resetStatus = true;
  smokeLED.on();
  Blynk.notify("WARNING: Smoke Leakage Detected!!");
  Blynk.email("IoT Fire Control", "WARNING: Smoke Leakage Detected!!");
  digitalWrite(gasValve, LOW);
  digitalWrite(acDevice1, LOW);
  digitalWrite(siren, HIGH);
  stateLCD.clear();
  stateLCD.print(0, 0, "IoT Fire Control");
  stateLCD.print(0, 1, "SMOKE Detected!!");
 }

 gasValue = analogRead(gasValue);
 Serial.println(gasValue);
 if (gasValue < 200)
 {
  safetyStatus = true;
  resetStatus = true;
  gasLED.on();
  Blynk.notify("WARNING: Gas Leakage Detected!!");
  Blynk.email("IoT Fire Control", "WARNING: Gas Leakage Detected!!");
  digitalWrite(gasValve, LOW);
  digitalWrite(acDevice1, LOW);
  //digitalWrite(acDevice2, LOW);
  digitalWrite(siren, HIGH);
  stateLCD.clear();
  stateLCD.print(0, 0, "IoT Fire Control");
  stateLCD.print(0, 1, "GAS Detected!!");
 }*/
}
```

```
BLYNK_WRITE(V6)
{
  int pinValue = param.asInt();
  if (pinValue == 1)
  {
    if (safetyStatus == true)
    {
      safetyStatus = false;
      resetStatus = false;
      fireLED.off();
      smokeLED.off();
      gasLED.off();
      digitalWrite(gasValve, HIGH);
      digitalWrite(acDevice1, HIGH);
      digitalWrite(siren, LOW);
      stateLCD.clear();
      stateLCD.print(0, 0, "IoT Fire Control");
      stateLCD.print(0, 1, "State is SAFE.");
    }
  }
}
```

Chapter 5:

4. Results & Discussion:

4.1 Findings:

5.1.1. Blynk IoT:

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things.

There are three major components in the platform:

Blynk App:

It allows to you create amazing interfaces for your projects using various widgets we provide.

Blynk Server:

It is responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It's open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.

Blynk Libraries:

There are for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

Now imagine: every time you press a Button in the Blynk app, the message travels to space the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a Blynk of an eye.

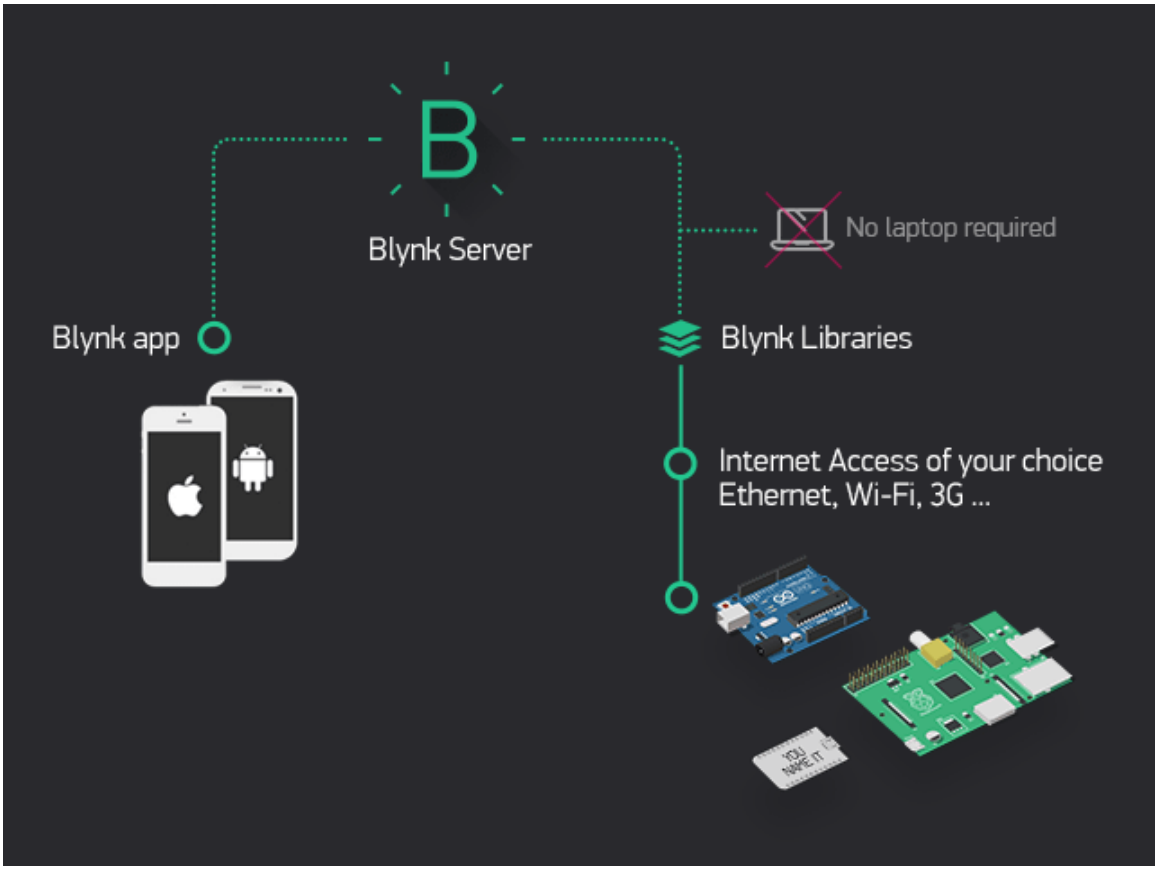


Figure 14: Blynk Architecture

Blynk IoT has the following features:

Similar API & UI for all supported hardware & devices.

Connection to the cloud using:

1. Wi-Fi.
2. Bluetooth and BLE.
3. Ethernet.
4. USB (Serial).
5. GSM.

- Set of easy-to-use Widgets.
- Direct pin manipulation with no code writing.
- Easy to integrate and add new functionality using virtual pins.

- History data monitoring via SuperChart widget.
- Device-to-Device communication using Bridge Widget.
- Sending emails, tweets, push notifications, etc.

You can find example sketches covering basic Blynk Features. They are included in the library. All the sketches are designed to be easily combined with each other.

Blynk IoT has the following ecosystem:

1. Hardware:

An Arduino, Raspberry Pi, or a similar development kit.
Blynk works over the Internet. This means that the hardware you choose should be able to connect to the internet. Some of the boards, like Arduino Uno will need an Ethernet or Wi-Fi Shield to communicate, others are already Internet-enabled: like the ESP8266, Raspberry Pi with Wi-Fi dongle, Particle Photon or SparkFun Blynk Board. But even if you don't have a shield, you can connect it over USB to your laptop or desktop (it's a bit more complicated for newbies, but we got you covered). What's cool, is that the list of hardware that works with Blynk is huge and will keep on growing.

2. A Smartphone:

The Blynk App is a well-designed interface builder. It works on both iOS and Android.

5.1.2. PCB Manufacturing Process:

The printed circuit board (PCB) acts as the linchpin for almost all of today's modern electronics. If the device needs to do some sort of computation — such as is the case even with simple items like a digital clock — chances are there's a PCB inside of it. PCBs bring electronics to life by routing electrical signals where they need to go to satisfy all the device's electronic requirements. For this to happen, PCBs are laid with

a network of paths outlined in the traces. It's these copper pathways that allow PCBs to direct electrical currents around their surface.

There are three main types of circuit boards that get manufactured on a consistent basis, and it's important to understand the differences between each so you can decide the right circuit board for your requirements. The three main types of circuit boards in current manufacture are:

Single-Sided Circuit Boards:

These boards when made with a FR4 base have rigid laminate of woven glass epoxy material, which is then covered on one side with a copper coating that is applied in varying thicknesses depending on the application.

Double-Sided Circuit Boards:

Double-sided boards have the same woven glass epoxy base as single-sided boards — however, in the case of a double-sided board, there is copper coating on both sides of the board, also to varying thicknesses depending on the application.

Multi-Layer Boards:

These use the same base material as single and double-sided boards, but are made with copper foil instead of copper coating — the copper foil is used to make "layers," alternating between base material and copper foil until the number of desired layers is reached.

Manufacturing Process:

Step 1: Design and Output.

Circuit boards should be rigorously compatible with, a PCB layout created by the designer using PCB design software. Commonly-used PCB design software includes Altium Designer, OrCAD, Pads, KiCad, Eagle etc. NOTE: Before PCB fabrication, designers should inform their contract manufacturer about the PCB design software version used to design the circuit since it helps avoid issues caused by discrepancies.

Once the PCB design is approved for production, designers export the design into format their manufacturers support. The most frequently used program is called extended Gerber. The 1980's baby food ad campaign sought beautiful babies, and this software creates some beautifully designed offspring. Gerber also goes by the name IX274X.

The PCB industry birthed extended Gerber as the perfect output format. Different PCB design software possibly calls for different Gerber file generation steps, they all encode comprehensive vital information including copper tracking layers, drill drawing, apertures, component notations and other options. All aspects of the PCB design undergo checks at this point. The software performs oversight algorithms on the design to ensure that no errors go undetected. Designers also examine the plan with regard to elements relating to track width, board edge spacing, trace and hole spacing and hole size.

After a thorough examination, designers forward PCB file to PC Board Houses for production. To ensure the design fulfils requirements for the minimum tolerances during manufacturing process, almost all PCB Fab Houses run Design for Manufacture (DFM) check before circuit boards fabrication.

Step 2: From File to Film.

PCB printing begins after designers output the PCB schematic files and manufacturers conduct a DFM check. Manufacturers use a special printer called a plotter, which makes photo films of the PCBs, to print circuit boards. Manufacturers will use the films to image the PCBs. Although it's a laser printer, it isn't a standard laser jet printer. Plotters use incredibly precise printing technology to provide a highly detailed film of the PCB design.

The final product results in a plastic sheet with a photo negative of the PCB in black ink. For the inner layers of PCB, black ink represents the conductive copper parts of the PCB. The remaining clear portion of the image denotes the areas of non-

conductive material. The outer layers follow the opposite pattern: clear for copper, but black refers to the area that'll be etched away. The plotter automatically develops the film, and the film is securely stored to prevent any unwanted contact.
Each layer of PCB and solder mask receives its own clear and black film sheet. In total, a two-layer PCB needs four sheets: two for the layers and two for the solder mask. Significantly, all the films have to correspond perfectly to each other. When used in harmony, they map out the PCB alignment.

To achieve perfect alignment of all films, registration holes should be punched through all films. The exactness of the hole occurs by adjusting the table on which the film sits. When the tiny calibrations of the table lead to an optimal match, the hole is punched. The holes will fit into the registration pins in the next step of the imaging process.

Step 3: Printing the Inner layers.

The creation of films in previous step aims to map out a figure of copper path. Now it's time to print the figure on the film onto a copper foil.

This step in PCB manufacturing prepares to make actual PCB. The basic form of PCB comprises a laminate board whose core material is epoxy resin and glass fibre that are also called substrate material. Laminate serves as an ideal body for receiving the copper that structures the PCB. Substrate material provides a sturdy and dust-resistant starting point for the PCB. Copper is pre-bonded on both sides. The process involves whittling away the copper to reveal the design from the films.

In PCB construction, cleanliness does matter. The copper-sided laminate is cleaned and passed into a decontaminated environment. During this stage, it's vital that no dust particles settle on the laminate. An errant speck of dirt might otherwise cause a circuit to be short or remain open.

Next, the clean panel receives a layer of photo-sensitive film called photo resist. The photo resist comprises a layer of photo reactive chemicals that harden after exposure to ultra violet light. This ensures an exact match from the photo films to the photo resist. The films fit onto pins that hold them in place over the laminate panel.
The film and board line up and receive a blast of UV light. The light passes through the clear parts of the film, hardening the photo resist on the copper underneath. The black ink from the plotter prevents the light from reaching the areas not meant to harden, and they are slated for removal.

After the board becomes prepared, it is washed with an alkaline solution that removes any photo resist left unhardened. A final pressure wash removes anything else left on the surface. The board is then dried.

The product emerges with resist properly covering the copper areas meant to remain in the final form. A technician examines the boards to ensure that no errors occur during this stage. All the resist present at this point denotes the copper that will emerge in the finished PCB.

This step only applies to boards with more than two layers. Simple two-layer boards skip ahead to drilling. Multiple-layer boards require more steps.

Step 4: Removing the Unwanted Copper.

With the photo resist removed and the hardened resist covering the copper we wish to keep, the board proceeds to the next stage: unwanted copper removal. Just as the alkaline solution removed the resist, a more powerful chemical preparation eats away the excess copper. The copper solvent solution bath removes all of the exposed copper. Meanwhile, the desired copper remains fully protected beneath the hardened layer of photo resist.

Not all copper boards are created equal. Some heavier boards require larger amounts of copper solvent and varying lengths of exposure. As a side note, heavier copper

boards require additional attention for track spacing. Most standard PCBs rely on similar specification.

Now that the solvent removed the unwanted copper, the hardened resist protecting the preferred copper needs washing off. Another solvent accomplishes this task. The board now glistens with only the copper substrate necessary for the PCB.

Step 5: Layer Alignment and Optical Inspection.

With all the layers clean and ready, the layers require alignment punches to ensure they all line up. The registration holes align the inner layers to the outer ones. The technician places the layers into a machine called the optical punch, which permits an exact correspondence so the registration holes are accurately punched.

Once the layers are placed together, it's impossible to correct any errors occurring on the inner layers. Another machine performs an automatic optical inspection of the panels to confirm a total absence of defects. The original design from Gerber, which the manufacturer received, serves as the model. The machine scans the layers using laser sensor and proceeds to electronically compare the digital image with the original Gerber file.

If the machine finds inconsistency, the comparison is displayed on a monitor for the technician to assess. Once the layer passes inspection, it moves to the final stages of PCB production.

Step 6: Layer-up and Bond.

In this stage, the circuit board takes shape. All the separate layers await their union. With the layers ready and confirmed, they simply need to fuse together. Outer layers must join with the substrate. The process happens in two steps: layer-up and bonding. The outer layer material consists of sheets of fibre glass, pre-impregnated with epoxy resin. The shorthand for this is called prepreg. A thin copper foil also covers the top

and bottom of the original substrate, which contains the copper trace etchings. Now, it's time to sandwich them together.

The bonding occurs on a heavy steel table with metal clamps. The layers securely fit into pins attached to the table. Everything must fit snugly to prevent shifting during the alignment.

A technician begins by placing a prepreg layer over alignment basin. The substrate layer fits over the prepreg before the copper sheet is placed. Further sheets of prepreg sit on top of the copper layer. Finally, an aluminium foil and copper press plate complete the stack. Now it's prepped for pressing.

The entire operation undergoes an automatic routine run by the bonding press computer. The computer orchestrates the process of heating up the stack, the point in which to apply pressure, and when to allow the stack to cool at a controlled rate. Next, a certain amount of unpacking occurs. With all the layers molded together in a super sandwich of PCB glory, the technician simply unpacks the multi-layer PCB product. It's a simple matter of removing the restraining pins and discarding the top pressure plate. The PCB goodness emerges victorious from within its shell of aluminium press plates. The copper foil, included in the process, remains to comprise the outer layers of the PCB.

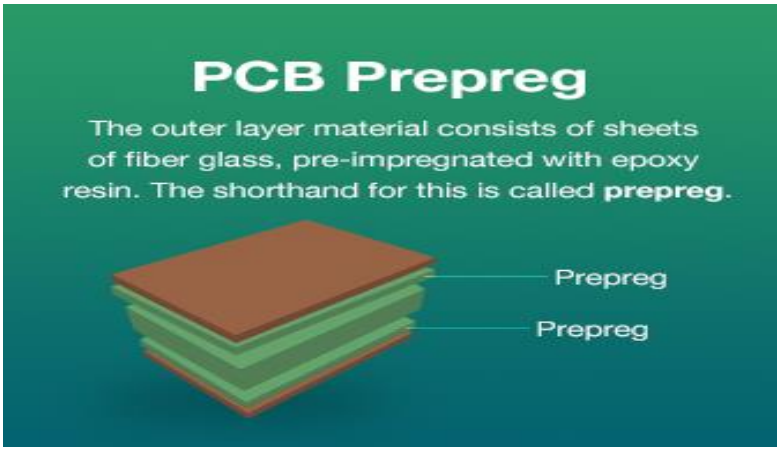


Figure 15: PCB Prepreg

Step 7: Drill.

Finally, holes are bored into the stack board. All components slated to come later, such as copper-linking via holes and leaded aspects, rely on the exactness of precision drill holes. The holes are drilled to a hairs-width - the drill achieves 100 microns in diameter, while hair averages at 150 microns.

To find the location of the drill targets, an x-ray locator identifies the proper drill target spots. Then, proper registration holes are bored to secure the stack for the series of more specific holes.

Before drilling, the technician places a board of buffer material beneath the drill target to ensure a clean bore is enacted. The exit-material prevents any unnecessary tearing upon the drill's exits.

A computer controls every micro-movement of the drill - it's only natural that a product that determines the behaviour of machines would rely on computers. The computer-driven machine uses the drilling file from the original design to identify the proper spots to bore.

The drills use air-driven spindles that turn at 150,000 rpm. At this speed, you might think that drilling happens in a flash, but there are many holes to bore. An average PCB contains well over one hundred bore intact points. During drilling, each needs its own special moment with the drill, so it takes time. The holes later house the vias and mechanical mounting holes for the PCB. The final affixation of these parts occurs later, after plating.

After the drilling completes itself, the additional copper that lines the edges of the production panel undergoes removal by a profiling tool.

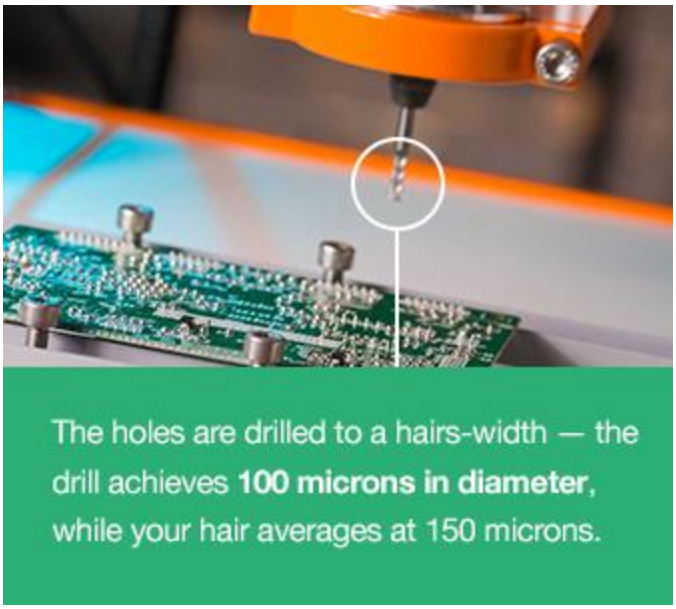


Figure 16: PCB Drilling

Step 8: Plating and Copper Deposition.

After drilling, the panel moves onto plating. The process fuses the different layers together using chemical deposition. After a thorough cleaning, the panel undergoes a series of chemical baths. During the baths, a chemical deposition process deposits a thin layer - about one micron thick - of copper over the surface of the panel. The copper goes into the recently drilled holes.

Prior to this step, the interior surface of the holes simply exposes the fibre glass material that comprises the interior of the panel. The copper baths completely cover, or plate, the walls of the holes. Incidentally, the entire panel receives a new layer of copper. Most importantly, the new holes are covered. Computers control the entire process of dipping, removal and procession.

Step 9: Outer Layer Imaging.

In Step 3, we applied photo resist to the panel. In this step, we do it again - except this time, we image the outer layers of the panel with PCB design. We begin with the layers in a sterile room to prevent any contaminants from sticking to the layer surface,

then apply a layer of photo resist to the panel. The prepped panel passes into the yellow room. UV lights affect photo resist. Yellow light wavelengths don't carry UV levels sufficient to affect the photo resist.

Black ink transparencies are secured by pins to prevent misalignment with the panel. With panel and stencil in contact, a generator blasts them with high UV light, which hardens the photo resist. The panel then passes into a machine that removes the unhardened resist, protected by the black ink opacity.
The process stands as an inversion to that of the inner layers. Finally, the outer plates undergo inspection to ensure all of the undesired photo resist was removed during the previous stage.

Step 10: Plating.
We return to the plating room. As we did in Step 8, we electroplate the panel with a thin layer of copper. The exposed sections of the panel from the outer layer photo resist stage receive the copper electro-plating. Following the initial copper plating baths, the panel usually receives tin plating, which permits the removal of all the copper left on the board slated for removal. The tin guards the section of the panel meant to remain covered with copper during the next etching stage. Etching removes the unwanted copper foil from the panel.

Step 11: Final Etching.
The tin protects the desired copper during this stage. The unwanted exposed copper and copper beneath the remaining resist layer undergo removal. Again, chemical solutions are applied to remove the excess copper. Meanwhile, the tin protects the valued copper during this stage.
The conducting areas and connections are now properly established.

Step 12: Solder Mask Application.
Before the solder mask is applied to both sides of the board, the panels are cleaned and covered with an epoxy solder mask ink. The boards receive a blast of UV light,

which passes through a solder mask photo film. The covered portions remain unhardened and will undergo removal.
Finally, the board passes into an oven to cure the solder mask.

Step 13: Surface Finish.

To add extra solder-ability to the PCB, we chemically plate them with gold or silver. Some PCBs also receive hot air-levelled pads during this stage. The hot air levelling results in uniform pads. That process leads to the generation of surface finish. PCBCart can process multiple types of surface finish according to customers' specific demands.

Step 14: Silkscreen.

The nearly completed board receives ink-jet writing on its surface, used to indicate all vital information pertaining to the PCB. The PCB finally passes onto the last coating and curing stage.

Step 15: Electrical Test.

As a final precaution, a technician performs electrical tests on the PCB. The automated procedure confirms the functionality of the PCB and its conformity to the original design. At PCBCart, we offer an advanced version of electrical testing called Flying Probe Testing, which depends on moving probes to test electrical performance of each net on a bare circuit board.

Figure 17: PCB Electrical Test

Step 16: Profiling and V-Scoring.

Now we've come to the last step: cutting. Different boards are cut from the original panel. The method employed either centers on using a router or a v-groove. A router leaves small tabs along the board edges while the v-groove cuts diagonal channels along both sides of the board. Both ways permit the boards to easily pop out from the panel.

4.2 Goals Achieved:

All the following goals have been successfully achieved during the project implementation period, as follows:

- Remotely securing houses from theft and breach incidents.
- Easily monitoring the safety and security statues of houses in a real-time manner.
- Speeding up the reaction process in case of susceptible security breaches or safety problems.
- Ensuring that all safety and security precautions are strict and valid.
- Enabling wireless ubiquitous communication for remote supervision.

4.3 Future Work:

- Supporting web platform.
- Deploying LoRa technology for long-range wireless communication.
- Adding smart surveillance Wi-Fi camera.
- Sending the reason of fire to FireFighting.
- Calling the user automatically.

4.4 Ethical Issues:

- **Social:**
  Addictive design
  All developers are anxious to create applications that people want to use. This is a good UX design. The problem is that some teams are developing apps that people love too much. There are ethical concerns about the role of digital platforms like social media.
  Suspicious possession of personal data
  As devices and software evolved, AI-based processing of biometrics and other contextual data about customers has grown exponentially. The software can profile users and predict behavior at terrifying levels of detail.

- **Specialist:**
  Impacts on privacy, accuracy, ownership, accessibility, and quality of life are all issues to consider when designing and deploying computer software systems. Choosing a specific approach to system development can hinder or facilitate the ethical treatment of these issues.

- **legal:**
  Code ownership In addition to performance issues, intellectual property ownership is an important issue that needs to be addressed to avoid legal issues. Nowadays, neither software developers nor customers really know who owns what. Many app development projects believe that both the client and the developer own the code, but the other person only has the permission (license) to use the application. Complicating ownership issues is the frequent use of open source (and even public domain) code as part of software development.

- In the future, work when it is implemented in companies, houses, or others, will be under licences from all parties, and I'll not accept the existence of ethical issues in my project.

Chapter 6:

Conclusion:

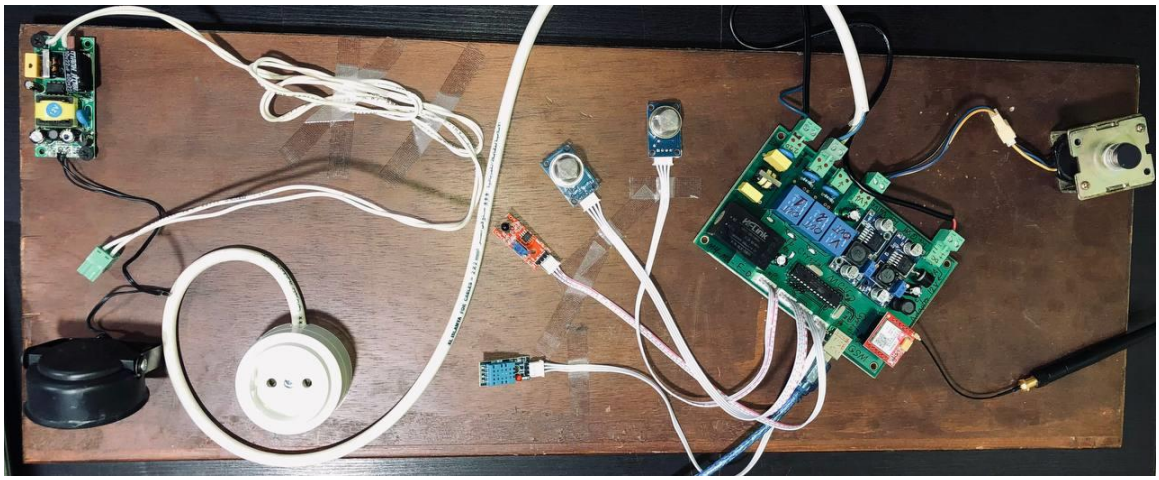
My project was completely different from the projects that existed in the labor market, as I collected all the means of security and safety in it. Most of the market projects, and perhaps 90% of them contain only certain parts of my project, not all together, most of them contain only gas sensors or only fire Or smoke only, and not all of them contain gas control, but a few of them, but in my project, I added the gas sensor, fire, heat, humidity and steam, in addition to the alarm device, with the addition of the most important element of the project, which is GSM, which gives me the opportunity to send mail and text messages through the Blynk application and through Gmail also makes the system available all the time. I did not use the Wi-Fi network because it has a certain distance, but the GSM device is not like that because it gives me the opportunity to use an internet chip. I also purchased a device that automatically controls gas, which is one of the most important components of the project and what makes it different from the existing projects in the Job market, several sensors were used in the project that sense most of the things that cause a fire to be created. Each sensor senses the danger instantaneously and disconnect the gas Valve, automatically disconnects the electricity from the place from the sensor that causes an outbreak fire or emergency event. I faced several problems during the implementation of my project and tried to fix it. It was difficult to get the gas controller from the gas company, but I communicated a lot with Town Gas and I also got it during the design of the electronic circuit. It was difficult to design it before implementation because it contains many details. I took a great effort, someone from the NTI factory helped me learn how to design it, in addition to a course to learn how to design and print an electronic circuit, and finally, measuring and estimating the extent of the smoke and gas sensor was somewhat confusing, but I learned a lot from this project.

Eco-friendly shop security system developed by combining two systems, namely home-security system based on microcontrollers and monitoring system based on Android/iOS

smartphone. The home security system is developed by using six types of security modules while monitoring system is developed by using advanced sensors, GSM technology “SIM800L mini” module, Blynk IoT and Android/iOS smartphone. This integrated system can detect both internal and external disturbances with excellent performance. This is proven by the excellent results of security module performance test and network connectivity test. The monitoring system also works well which is indicated by the success rate of notification sending and successful monitoring application run on Android/iOS smartphone. The next system will be developed for home security systems on cluster housing using LoRaWAN technology.

Also I’ll add some features like sending location to FireFighting, Web platform and make more safety and protection to save people from any danger or all of fire problems .

Finally, I hope my own work to be liked by all of you and hope my project to be benefit to our society and country and save people souls and help them to can control most of fire problems .



7- References:

- Al-Sarawi, S., Anbar, M., Alieyan, K. and Alzubaidi, M., 2017, May. Internet of Things (IoT) communication protocols. In 2017 8th International conference on information technology (ICIT) (pp. 685-690). IEEE.
- Yokotani, T. and Sasaki, Y., 2016, September. Comparison with HTTP and MQTT on required network resources for IoT. In 2016 international conference on control, electronics, renewable energy and communications (ICCEREC) (pp. 1-6). IEEE.
- Upadhyay, Y., Borole, A. and Dileepan, D., 2016, March. MQTT based secured home automation system. In 2016 Symposium on Colossal Data Analysis and Networking (CDAN) (pp. 1-4). IEEE.
- ATmega328P, M., 3.1. Microcontroller ATmega328P.
- THINKER, A., 2017. ESP-01 WiFi Module.
- Doshi, H.S., Shah, M.S. and Shaikh, U.S.A., 2017. Internet Of Things (Iot): Integration Of Blynk For Domestic Usability. VJER-Vishwakarma Journal of Engineering Research, 1(4).
- Singh, K.J. and Kapoor, D.S., 2017. Create Your Own Internet of Things: A survey of IoT platforms. IEEE Consumer Electronics Magazine, 6(2), pp.57-68.
- Gill, M., Bilby, C. and Turbin, V., 1999. Retail security: Understanding what deters shop thieves. Journal of Security Administration, 22(1), p.29.
- Ekblom, P., 1986. The prevention of shop theft: an approach through crime analysis (p. 5). London: Home Office.

- Lemke, J., 2004. Microsoft Office Visio 2003 step by step. Microsoft Press.
- Phillips, J., 2013. PMP, Project Management Professional. Schwartz, M., 2016. Internet of Things with ESP8266. Packt Publishing Ltd.
- Al-Sarawi, S., Anbar, M., Alieyan, K. and Alzubaidi, M., 2017, May. Internet of Things (IoT) communication protocols. In 2017 8th International conference on information technology (ICIT) (pp. 685-690). IEEE.
- Yokotani, T. and Sasaki, Y., 2016, September. Comparison with HTTP and MQTT on required network resources for IoT. In 2016 international conference on control, electronics, renewable energy and communications (ICCEREC) (pp. 1-6). IEEE.
- Upadhyay, Y., Borole, A. and Dileepan, D., 2016, March. MQTT based secured home automation system. In 2016 Symposium on Colossal Data Analysis and Networking (CDAN) (pp. 1-4). IEEE.
- ATmega328P, M., 3.1. Microcontroller ATmega328P.
- THINKER, A., 2017. ESP-01 WiFi Module.
- Doshi, H.S., Shah, M.S. and Shaikh, U.S.A., 2017. Internet Of Things (Iot): Integration Of Blynk For Domestic Usability. VJER-Vishwakarma Journal of Engineering Research, 1(4).
- Singh, K.J. and Kapoor, D.S., 2017. Create Your Own Internet of Things: A survey of IoT platforms. IEEE Consumer Electronics Magazine, 6(2), pp.57-68.

- Gill, M., Bilby, C. and Turbin, V., 1999. Retail security: Understanding what deters shop thieves. Journal of Security Administration, 22(1), p.29.
- Ekblom, P., 1986. The prevention of shop theft: an approach through crime analysis (p. 5). London: Home Office.
- Lemke, J., 2004. Microsoft Office Visio 2003 step by step. Microsoft Press.
- Phillips, J., 2013. PMP, Project Management Professional. Schwartz, M., 2016. Internet of Things with ESP8266. Packt Publishing Ltd.