VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgaum - 590018

A project report on

"IOT Based Smart Door Lock System"

submitted in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

by

1CR18IS081 Kushagra Goyal

1CR18IS090 Mohammed Akbar Awais

Under the Guidance of

Dr. S. Seetha

Associate Professor

Department of ISE, CMRIT, Bengaluru

CMR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

#132, AECS Layout, IT Park Road, Bengaluru - 560037

2021-22

CMR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

#132, AECS Layout, IT Park Road, Bengaluru - 560037

Certificate

This is to Certified that the project work entitled **"IoT Based Smart Door Lock System"** carried out by **Mr.Kushagra Goyal (1CR18IS081), Mr.Mohammed Akbar Awais (1CR18IS090)** in partial fulfillment for the award of Bachelor of Engineering in **Information Science & Engineering** of the Visveswaraiah Technological University, Belgaum during the year **2021-22**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

----------	----------	----------
Signature of Guide	Signature of HoD	Signature of Principal
Dr. S. Seetha	**Dr. Farida Begam**	**Dr. Sanjay Jain**
Associate Professor	Professor & Head	Principal
Department of ISE	Department of ISE	CMRIT,
CMRIT	CMRIT	Bengaluru - 37

External Viva

	Name of the Examiners	Institution	Signature with Date
1.	----------	----------	----------
2.	----------	----------	----------

CMR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

#132, AECS Layout, IT Park Road, Bengaluru - 560037

Declaration

We, **Mohammed Akbar Awais(1CR18IS090) And Kushagra Goyal (1CR 18IS081)**,bonafide students of **CMR Institute of Technology**, Bangalore, hereby declare that the dissertation entitled, **"IoT Based Smart Door Lock System"** has been carried out by us under the guidance of **Dr. S. Seetha**, Associate Professor, CMRIT, Bangalore, in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science & Engineering, of the Visvesvaraya Technological University, Belgaum during the academic year 2021-2022. The work done in this dissertation report is original and it has not been submitted for any other degree in any university.

Kushagra Goyal

Mohammed Akbar Awais

Acknowledgement

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible. Success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So, it is with gratitude that we acknowledge all those whose guidance and encouragement served as beacon of light and crowned our effort with success.

We would like to thank Dr. Sanjay Jain, Principal, CMRIT, Bangalore, for providing an excellent academic environment in the college and his never-ending support for the B.E program.

We would like to express our gratitude towards Dr. Farida Begam, Professor and HOD, Department of Information Science & Engineering CMRIT, Bangalore, who provided guidance and gave valuable suggestions regarding the project.

We consider it a privilege and honour to express our sincere gratitude to our internal guide Dr. S. Seetha, Associate Professor, Department of Information Science & Engineering, CMRIT, Bangalore, for their valuable guidance throughout the tenure of this project work.

We would like to thank all the faculty members who have always been very cooperative and generous. Conclusively, we also thank all the non-teaching staff and all others who have done immense help directly or indirectly during our project.

Kushagra Goyal
Mohammed Akbar Awais

Abstract

An ESP 8266 is Wi-Fi enabled wireless micro controller. It has been implemented in this scenario in a door mounted security system with the help of a sensor. The basic components that are used here are a reed switch, an alarm, an ESP 8266 module and a e power supply. 3 interfaces have been used. First an ESP 8266, which is connected to a buzzer and a reed switch. These three components have been mounted on the door frame. The second interface is an application which will run on either a stationary workstation or a mobile workstation. The third component is a server, which will handle traffic routing and synchronization tasks when a scenario with multiple doors is involved. These three interfaces will be connected and synchronization will be carried out with the help of the MQTT(Message Queuing Telemetry Transport) protocol. Two of these three interfaces i.e the application and the door mounted system will serve as clients whereas the server will be the broker. The system will use a messaging mechanism to ensure two step verification either via email or via a text message. This system will be fault tolerant and scalable because it will be easy to facilitate addition of new devices.

Keywords: *:IoT (Internet of Things), ESP8266, MQTT Protocol.*

Table of Contents

List of Figures

Chapter 1

Preamble

1.1 Introduction

Application services based on information and communication technology has been actively investigated in the knowledge information society. In particular, the most rapid growth can be observed in convergence services which combines more than two elements for the same purpose. Convergence services prove to represent Internet of Things (IoT) technology, as it enables all objects to provide intelligent service and interactive communication through wired or wireless networks.

Furthermore, the IoT industry is deemed the core industrial field of the future. IoT provides convenient and effective services in any place at any time, beyond the technical and economical restrictions, as well as the temporal and spatial limits by providing services required in various kinds of fields. It also aids the distribution of intelligent terminals which includes smart phones, in conjunction with the advancement of information and communication technology.

Meanwhile, the demand on convenience and speed has increased in the economic sectors of modern society. The financial sector, amongst other fields, require IoT technology as mentioned above. Financial institutes have increased the distribution of unmanned and automated machines to strengthen competitiveness by advancing financial services, streamlining the business processes, automating the system, and ultimately reducing costs.

While increasing the distribution of unmanned automated systems satisfies the social demand, as replacing the human operators for financial services, it also generates

incidental side effects. In each bank, the physical keys used to manage the unmanned automated systems are kept by a cash transportation staff, a maintenance staff, a security staff, and a bank teller. Accordingly, it is highly possible that the physical keys can be lost, stolen, or reproduced. It is thereby required to strengthen the security for the system in its administrative aspects.

With respect to the security of the system, security risks related to the locking device of automated teller machines has been pointed out in a presentation entitled 'Jackpotting' in 'BlackHat', the largest security conference in the world held in July of 2010. Thus, there is urgent need to develop the security system applicable to IoT technology in the financial sector.

Accordingly, this project proposes an approach to strengthen the security in the administrative aspects of the locking device applied to the unmanned automated system used in the financial sector, by applying the server authentication method using a smart phone as the repeater on its technological aspects.

Home automation is being used pervasively in today's world. Nowadays almost everything which uses computers or microprocessors is being automated in some or the other way. This project concentrates more on the automation of home security. Automated home security has many advantages, although it provides lesser security than actual physical security. It is primarily flexible because we can change it in our accordance. It is also simpler to maintain. It is considerably more economical and it also provides cost effectiveness to an already flexible price. The application which uns on home security can be made as efficient as the programmer wants it to be. If the entities being protected are important, then more funds can be put in to make a more stable and efficient application. The application used in this paper, however, is one which is economical and can be used by anyone wanting basic security measures but not wanting to spend a lot of money. It is a lightweight, cost effective, application specific system.

1.2 Existing System

Healthcare: Patients can be monitored in real-time using IoT.

Public Sector: Government authorities can notify people of a city of outages or interruptions in the supply of water or electricity. Parking spaces can be well maintained using data obtained by optical or infrared sensors.

Manufacturing: Manufacturers can deploy IoT to monitor the manufacturing processes and avoid any irregularities. The quality of the products can also be confirmed using IoT.

Automobiles: IoT can be implemented to supervise the production of automobiles. Even after the car has been sold, the IoT can enhance the user experience on the road (kind of like what Tesla does).

Retail: Retailers can benefit a lot if they implement IoT. They can use IoT to optimize the supply chain, reduce operational costs and improve customer experience. They can use smart shelves to notify them if they run out of products, for instance.

Transportations and logistics: Transport vehicles can be equipped with IoT sensors to tell drivers the optimal driving route according to weather, road condition, and traffic. This can save a lot of costs.

Wearables: Smart wearables like smartwatches can track everything from calories burned to heartbeat rate and blood oxygen levels.

Smart homes: Smart homes are the most popular application of IoT. Smart surveillance systems, smart air conditioners, smart lighting systems, smart speaker systems, and smart fridges are some features of smart homes.

1.2.1 Drawbacks

1. Data breach : The physical objects in Iot connect to the internet to transmit and receive information. So the user data is now available on the internet and hackers can hack this private and sensitive information. Data breach occurs when outsiders can access the user data without the awareness of the user. This data can be used to manipulate the user.

2. Security and privacy concerns :Iot systems are connected to a number of devices and communicate within a network. This means that there is little to no privacy between devices on the network.

3. Security and privacy concerns Iot systems are connected to a number of devices and communicate within a network. This means that there is little to no privacy between devices on the network.

4. Corruption of the entire system Any bug in one of the devices leads to corruption in the entire system.

5. Increase in unemployment IoT robots replace manual forces in various industries to a large extent. This causes the unemployment of factory labourers and sometimes leads to lesser wages for more skilled labourers.

1.3 Proposed System

This model has been designed with the current home security applications scenarios and the purpose of a low budget but efficient system in mind. MQTT has 3 QOS levels; fire and target, delivered at least once and delivered exactly once. Out of these three, this project will be employing the delivered at least once i.e the second level for quality of service to ensure optimal communication.

The model uses a central subscriber program because this application might be employed to control multiple doors and hence multiple ESP8266 door mounted systems. This ensures that the system built will be scalable i.e new processors can be added very easily

1.4 Plan of Implementation

Components which is required to implement this project include solenoid lock, 12v power supply, relay module, NodeMCU ESP8266 and a buzzer in which all will be connected through a breadboard and inject the code in the esp8266 to run as per the requirement.

1.5 Problem Statement

Most of the people always in the urgency situation. This is probably inflicting the issues inclusive of forgetting to lock the door in their home. sometimes, they may lose the key of the door. Apart from that, there may be less protection on the door lock which the burglar can break the door.

1.6 Objective of the Project

The objectives of the "IoT Based Smart Door Lock System" can be stated as follows:

1. Read the LastRead value from the MQTT Server.
2. Compare the String of the LastRead with Close and Open.
3. If it is close The door will be unlocked else locked.

Chapter 2

Literature Survey

2.1 Internet of Things

Today, we are living in the era of smart technologies which represent a "ubiquitous computing" or "web 0.3" . The use of the internet in the future can dominate all work done and defeat human computing capabilities such as controlling electronic equipment remotely using internet media. All objects have an IP as their address and can be tracked. Technology trends that will affect the IT field in the next five years will be laid out by Gartner and among them are the Internet of Things. Internet of Things is a concept or paradigm that considers the existence of various things/objects/applications/other services in the environment to create something new and achieve common goals. All objects have an IP as their address and can be traced. Internet of Things is defined as an invention that can solve existing problems by combining technology and social impact, meanwhile if viewed from the standardization of IoT techniques it can be described as a global infrastructure to meet the needs of the community . Every equipment used in life from transportation, household equipment, and industry can be connected to the internet and monitoring can be easier and can be done anywhere. According to Betts , there will be many technology towers referring to IoT, such as communication machines, cellular and telecommunications networks, applications, smart devices, and security. However, all that cannot be predicted for future development.

2.2 ESP8266

The ESP8266 is a low-cost Wi-Fi microchip, with built-in TCP/IP networking software, and microcontroller capability, produced by Espressif Systems in Shanghai, China.

The chip was popularized in the English-speaking maker community in August 2014 via the ESP-01 module, made by a third-party manufacturer Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at first, there was almost no English-language documentation on the chip and the commands it accepted. The very low price and the fact that there were very few external components on the module, which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, the chip, and the software on it, as well as to translate the Chinese documentation.

The ESP8285 is a similar chip with a built-in 1 MiB flash memory, allowing the design of single-chip devices capable of connecting via Wi-Fi.

These microcontroller chips have been succeeded by the ESP32 family of devices.

Chapter 3

Theoretical Background

Theoretical background highlighting some topics related to the project work is given below. The description contains several topics which are worth to discuss and also highlight some of their limitation that encourage going on finding solution as well as highlights some of their advantages for which reason these topics and their features are used in this project.

3.1 Solenoid lock

A solenoid is a type of electromagnet formed by a helical coil of wire whose length is substantially greater than its diameter, which generates a controlled magnetic field. The coil can produce a uniform magnetic field in a volume of space when an electric current is passed through it. The term solenoid was coined in 1823 by André-Marie Ampère.

The helical coil of a solenoid does not necessarily need to revolve around a straight-line axis; for example, William Sturgeon's electromagnet of 1824 consisted of a solenoid bent into a horseshoe shape (not unlike an arc spring).

Solenoids provide magnetic focusing of electrons in vacuums, notably in television camera tubes such as vidicons and image orthicons. Electrons take helical paths within the magnetic field. These solenoids, focus coils, surround nearly the whole length of the tube.

In engineering, the term "solenoid" refers not only to the electromagnet but to a complete apparatus providing an actuator that converts electrical energy to mechanical energy.

3.1.1 Advantages of Solenoid Lock

- Excellent for releasing sticky latches with minimal noise
- Safer than manual lock systems because they lock automatically when the remote is pressed.
- Advanced design minimizes the size and weight of the device.
- Consumes small amounts of energy; no power is utilized when the door is locked
- Durable and shockproof
- Fully scalable and can be designed in various sizes to fit different applications.

3.1.2 Disadvantages of Solenoid Lock

- The potential for the coil to need replacing during its lifetime
- The need for the control signal to remain during its operation
- Sensitivity to voltage fluctuations or changes
- Unintended partial closure of the valve if the magnetic field isn't properly set up
- The effect of the flow on the valve fluid.

3.2 Relay Module

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal. Relays were first used in long-distance telegraph circuits as signal repeaters: they refresh the signal coming in from one circuit by transmitting it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

The traditional form of a relay uses an electromagnet to close or open the contacts,

but relays using other operating principles have also been invented, such as in solid-state relays which use semiconductor properties for control without relying on moving parts. Relays with calibrated operating characteristics and sometimes multiple

operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called protective relays.

Latching relays require only a single pulse of control power to operate the switch persistently. Another pulse applied to a second set of control terminals, or a pulse with opposite polarity, resets the switch, while repeated pulses of the same kind have no effects. Magnetic latching relays are useful in applications when interrupted power should not affect the circuits that the relay is controlling.

3.2.1 Advantages of Relay Module

- Simple and effective operation
- Circuit multiplication
- Galvanic isolation
- Voltage conversion
- Compact size and low cost
- Easy to install and troubleshoot
- High resistance
- Maintenance-free and long lifespan

3.2.2 Disadvantages of Relay Module

- Used only low current applications.
- Low speed of operation.
- Poor performance in high inrush currents and microelectronic circuits.
- Change in characteristics due to aging.
- Low performance in vibrated environments.

- Contact wear.

3.3 NodeMCU ESP8266

NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit).The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits.

Both the firmware and prototyping board designs are open source

The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects).

The ESP8266 is a low-cost Wi-Fi microchip, with built-in TCP/IP networking software, and microcontroller capability, produced by Espressif Systems in Shanghai, China.

The chip was popularized in the English-speaking maker community in August 2014 via the ESP-01 module, made by a third-party manufacturer Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at first, there was almost no English-language documentation on the chip and the commands it accepted. The very low price and the fact that there were very few external components on the module, which suggested that it could eventually be very inexpensive in volume,

attracted many hackers to explore the module, the chip, and the software on it, as well as to translate the Chinese documentation.

The ESP8285 is a similar chip with a built-in 1 MiB flash memory, allowing the design of single-chip devices capable of connecting via Wi-Fi.

These microcontroller chips have been succeeded by the ESP32 family of devices.

3.4 Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, train and confirmation of user input such as a mouse click or keystroke.

Chapter 4

System Requirements Specification

A System Requirement Specification (SRS) is basically an organization's understanding of a customer or potential client's system requirements and dependencies at a particular pointprior to any actual design or development work. The information gathered during the analysis is translated into a document that defines a set of requirements. It gives the brief description of the services that the system should provide and also the constraints under which, the system should operate. Generally, SRS is a document that completely describes what the proposed software should do without describing how the software will do it. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time.

SRS document itself states in precise and explicit language those functions and capabilities a software system (i.e., a software application, an e-commerce website and so on) must provide, as well as states any required constraints by which the system must abide. SRS also functions as a blueprint for completing a project with as little cost growth as possible. SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it.

Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable attributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources

4.1 Functional Requirement

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

Following are the functional requirements on the system:

- The entire control model set must be translated to Embedded C output Code.
- Inputs must be taken from the ESP8266 standard input output pins which is defined in the code itslef
- Multiple input must be processed and the result must be combined to obtain a single output file.

4.2 Non Functional Requirements

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

- Product Requirements
- Organizational Requirements
- User Requirements
- Basic Operational Requirements

4.2.1 Product Requirements

Platform Independency: Standalone executables for embedded systems can be created so the algorithm developed using available products could be downloaded on the actual hardware and executed without any dependency to the development and modeling platform

Correctness: It followed a well-defined set of procedures and rules to compute and also rigorous testing is performed to confirm the correctness of the data.

Ease of Use: Model Coder provides an interface which allows the user to interact in an easy manner

Modularity: The complete product is broken up into many modules and well-defined interfaces are developed to explore the benefit of flexibility of the product.

Robustness: This software is being developed in such a way that the overall performance is optimized and the user can expect the results within a limited time with utmost relevancy and correctness.

Non functional requirements are also called the qualities of a system. These qualities can be divided into execution quality evolution quality. Execution qualities are security usability of the system which are observed during run time, whereas evolution quality involves testability, maintainability, extensibility or scalability

4.2.2 Organizational Requirements

Design Methods: Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

4.2.3 User Requirements

- The Computer Must be able to recognize the ESP8266 through USB.
- The IDE Must be able to detect correct board.
- The IDE Must be able to export code into ESP8266.
- The ESP8266 pins must have output functionality working properly.

4.2.4 Basic Operational Requirements

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

Mission profile or scenario: It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.

Performance and related parameters: It points out the critical system parameters to accomplish the mission

Utilization environments: It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.

Operational life cycle: It defines the system lifetime.

4.2.5 System Configuration

H/W System Configuration:

Processor	- Pentium –IV
Speed	- 1.1 Ghz
RAM	- 256 MB(min)
Hard Disk	- 600 MB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA

S/W System Configuration:

Operating System :Windows/Linux/Ubuntu.
Coding Language : C/C++
Tools : Arduino Ide

Chapter 5

System Analysis

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

5.1 Feasibility Study

All systems are feasible when provided with unlimited resource and infinite time. But unfortunately this condition does not prevail in practical world. So it is both necessary and prudent to evaluate the feasibility of the system at the earliest possible time. Months or years of effort, thousands of rupees and untold professional embarrassment can be averted if an illconceived system is recognized early in the definition phase. Feasibility risk analysis are related in many ways. If project risk is great, the feasibility of producing quality software is reduced.

5.1.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

5.1.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

5.1.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

5.2 Analysis

5.2.1 Performance Analysis

For the complete functionality of the project work, the project is run with the help of healthy networking environment. Performance analysis is done to find out whether the proposed system. It is essential that the process of performance analysis and definition must be conducted in parallel.

5.2.2 Technical Analysis

System is only beneficial only if it can be turned into information systems that will meet the organization's technical requirement. Simply stated this test of feasibility asks whether the system will work or not when developed installed, whether there are any major barriers to implementation. Regarding all these issues in technical analysis there are several points to focus on:

Changes to bring in the system: All changes should be in positive direction, there will be increased level of efficiency and better customer service

Required skills: Platforms tools used in this project are widely used. So the skilled manpower is readily available in the industry

Acceptability: The structure of the system is kept feasible enough so that there should not be any problem from the user's point of view.

5.2.3 Economical Analysis

Economic analysis is performed to evaluate the development cost weighed against the ultimate income or benefits derived from the developed system. For running this system, we need not have any routers which are highly economical. So the system is economically feasible enough

Chapter 6

System Design

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translated into a representation of software. Design is a place where design is fostered in software Engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. Design is the perfect way to accurately translate a customer's requirement in the finished software product. Design creates a representation or model, provides details about software data structure, architecture, interfaces and components that are necessary to implement a system. The logical system design arrived at as a result of systems analysis is converted into physical system design.

6.1 System Development Methodology

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

6.1.1 Model phases

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design, Implementation, Testing and maintenance.

Requirement Analysis: This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.

System Design: Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on:-algorithm, data structure, software architecture etc.

Coding: In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in to machine readable compute code.

Implementation: The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation

Testing: In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

Maintenance: The maintenance phase is the longest phase in which the software is updated to fulfill the changing customer need, adapt to accommodate change in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

6.2 Data Flow Diagram

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system,various processing carried out on these data, and the output data is generated by the system.

The data flow diagram shows how the data control flows from one module to another. unless the board is connected to the WiFi the program cannot proceed to the next module once the WiFi is connected to then it will connect to the MQTT Server and then get the commands from the server compare it with reference string and assign GPIO pins high/low and continue to flow in the loop unless and until the server have no commands to receive

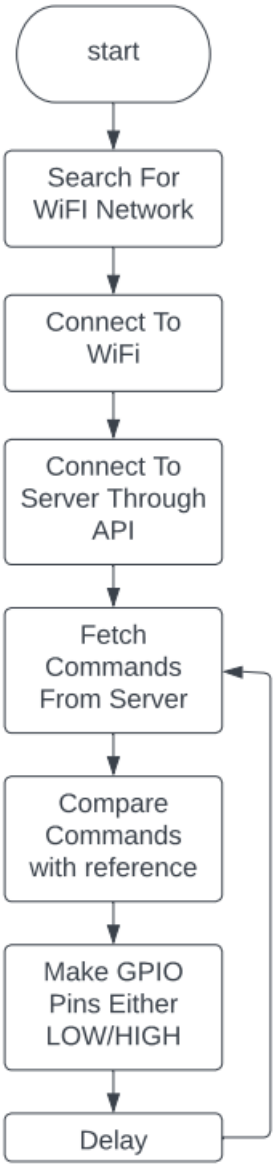

Figure 6.1: Data Flow Diagram

6.3 Component Diagram

Connections for this IoT Smart Door Lock are very simple as we are only connecting a solenoid lock, relay module, and a buzzer with NodeMCU ESP8266. The input pin of the relay is connected to the D5 pin of NodeMCU while VCC and Ground pins are connected to Vin and GND pin of NodeMCU. The positive pin of the buzzer is connected to the D6 pin of NodeMCU, and the GND pin is connected to the GND of NodeMCU.

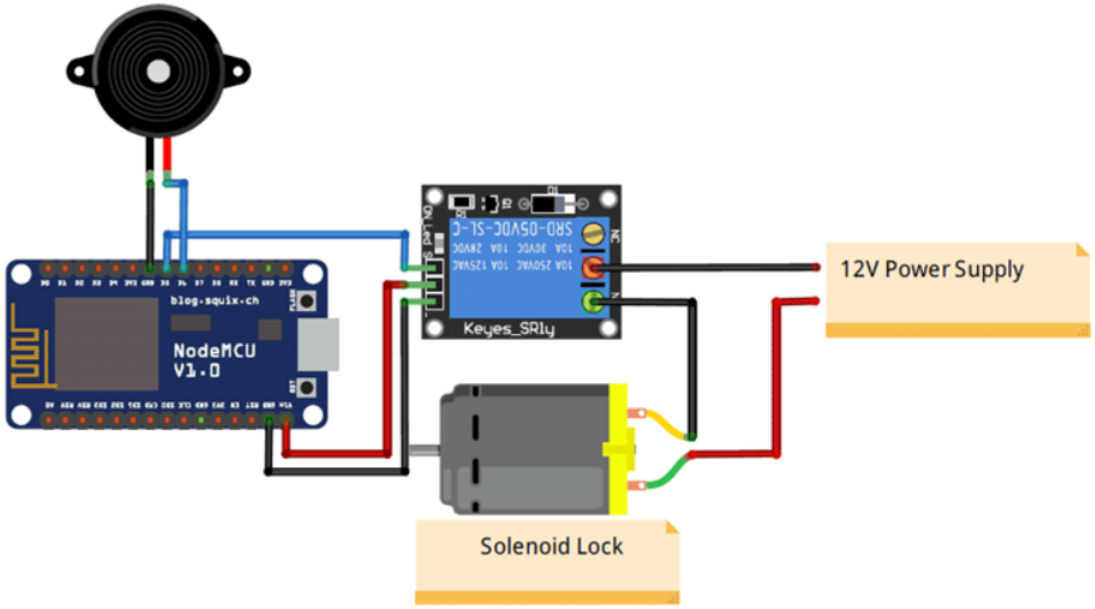

Figure 6.2: Block Diagram

6.4 Use Case Diagram

A use case defines a goal-oriented set of interactions between external entities and the system under consideration. The external entities which interact with the system are its actors. A set of use cases describe the complete functionality of the system at a particular level of detail and it can be graphically denoted by the use case diagram.

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

In software and systems engineering, a use case is a list of steps, typically defining interactions between a role (known in Unified Modeling Language (UML) as an "actor") and a system, to achieve a goal. The actor can be a human, an external system, or time.

In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in Systems Modeling Language (SysML) or as contractual

statements.

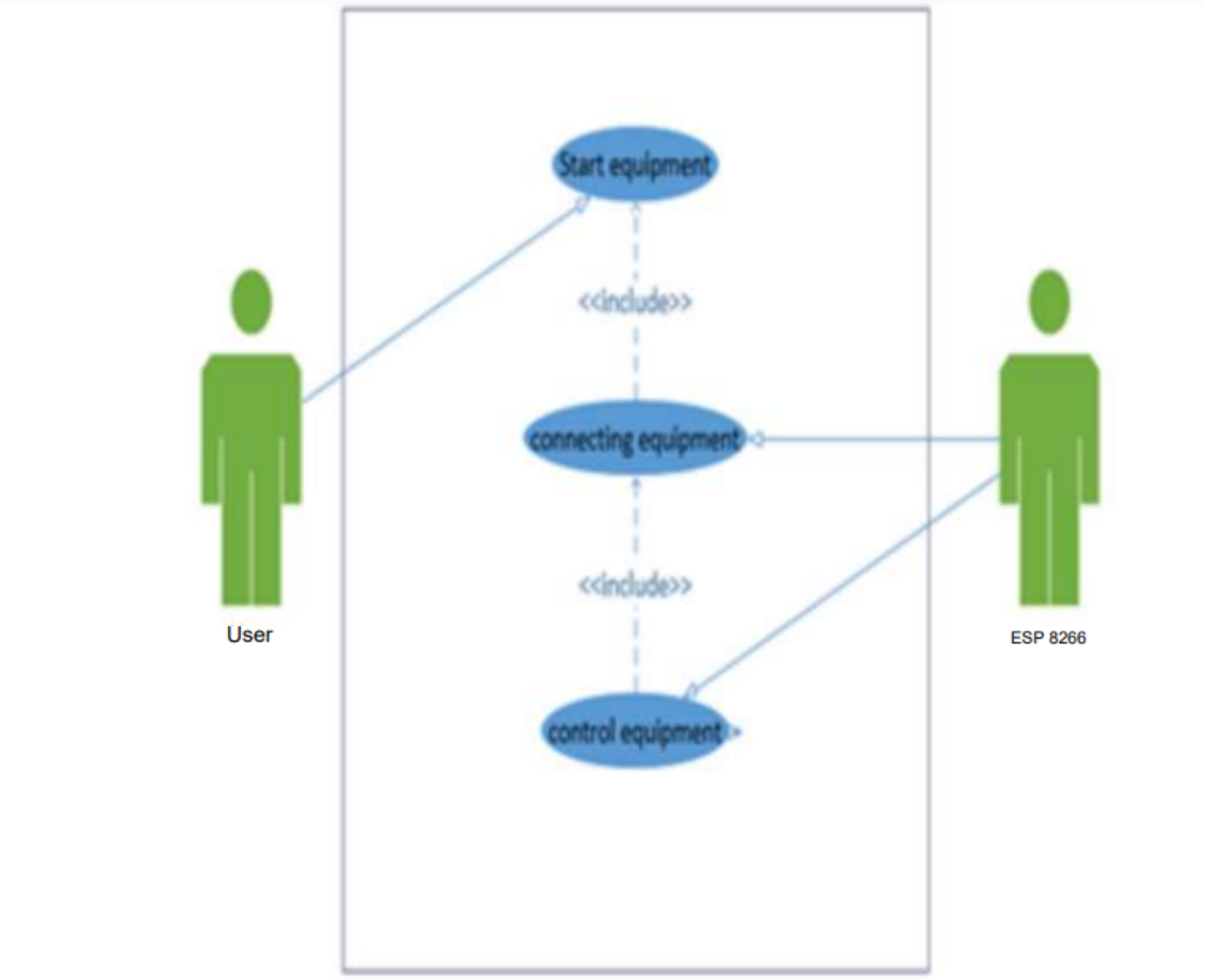

Figure 6.3: Use Case Diagram

The Seqeunce of activities that are carried out are the same as the other diagrams .Use case for this module indicates the users interaction with the system as a whole rather than individual modules .All the encryption mechanisms are carried out via the login page that redirects the user to the particular functionality that he or she wishes to implement .

6.5 Sequence Diagram

Sequence diagram are an easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram

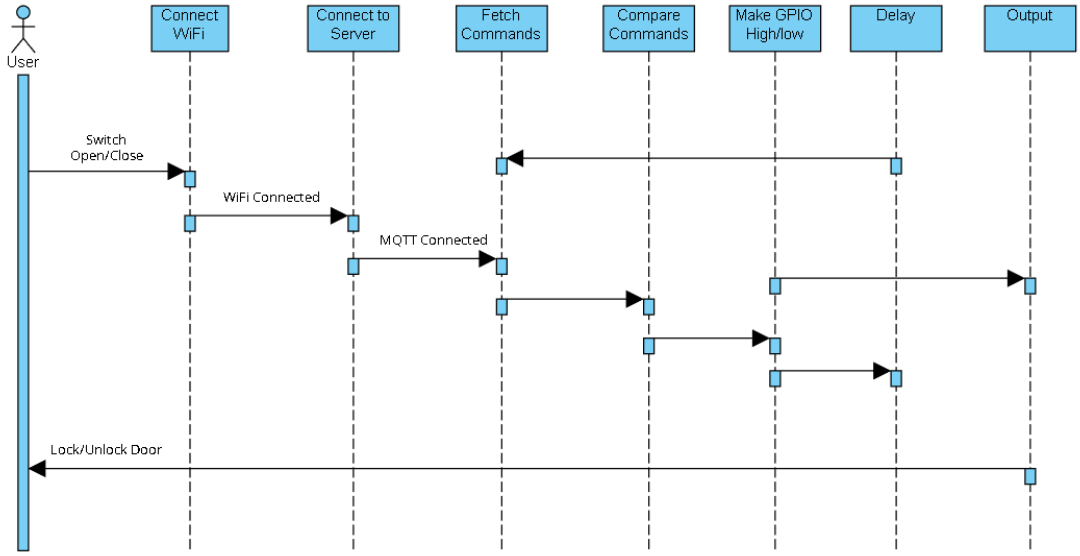

Figure 6.4: Sequence Diagram

has two dimensions: vertical dimension represents time, the horizontal dimension represents the objects existence during the interaction.

A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios

Chapter 7

Implementation

7.1 Adafruit IO Setup for IoT Door Lock

Adafruit IO is an open data platform that allows you to aggregate, visualize, and analyze live data on the cloud. Using Adafruit IO, you can upload, display, and monitor your data over the internet, and make your project IoT enabled. You can control motors, read sensor data, and make cool IoT applications over the internet using Adafruit IO. For test and try, with some limitation, Adafruit IO is free to use. We have also used Adafruit IO with Raspberry Pi, Arduino and ESP32 previously.

- To use Adafruit IO, first, you have to create an account on Adafruit IO. To do this, go to the Adafruit IO website and click on 'Get started for Free' on the top right of the screen.
- After finishing the account creation process, log in to your account and click on 'AIO Key' on the top right corner to get your account username and AIO key.When you click on 'AIO Key,' a window will pop up with your Adafruit IO AIO Key and username. Copy this key and username, it will be needed later in the code.
- Now, after this, you need to create a feed. To create a feed, click on 'Feed.' Then click on 'Actions,' and then click on 'Create a New Feed'.
- After this, a new window will open to enter the Name and Description of the feed. The writing description is optional.
- Click on 'Create,' after this; you will be redirected to your newly created feed.After creating the feed, now create an Adafruit IO dashboard to add a toggle button to open and close the door lockTo create a dashboard, click on the

Dashboard option and then click on the 'Action,' and after this, click on 'Create

a New Dashboard.'. For that, first, create a dashboard and then add your feed in this dashboard.To create a dashboard, click on the Dashboard option and

then click on the 'Action,' and after this, click on 'Create a New Dashboard.'In the next window, enter the name for your dashboard and click on 'Create.'

- As the dashboard is created, now, we will add our feeds to the dashboard. To add a feed, click on the '+' in the top right corner.
- Here we will add a toggle button blocks to turn open and close the Wi-Fi door lock. To add a button on the dashboard, click on the Toggle block.In the next window, it will ask you to choose the feed, so click on your feed.After adding the toggle button, my dashboard looks like below. You can edit the dashboard by clicking on the settings buttons.

7.1.1 Code Explanation

First, include all the required libraries. In this program, only Adafruit MQTT and ESP8266WiFi.h libraries are used.

Then include the WiFi and Adafruit IO credentials that you copied from the Adafruit IO server. These will include the MQTT server, Port No, User Name, and AIO Key

```
const char *ssid =  "Wi-Fi Name"; // Enter your WiFi Name
const char *pass =  "Password"; // Enter your WiFi Password
#define MQTT_SERV "io.adafruit.com"
#define MQTT_PORT 1883
#define MQTT_NAME "Adafruit IO Username"
#define MQTT_PASS "Your API Ley"
```

Then set up an MQTT connection using the credentials that you provided earlier.

```
Adafruit_MQTT_Client mqtt(&client ,MQTT_SERV,MQTT_PORT,
MQTT_NAME, MQTT_PASS);
```

Set up the feeds you're subscribing to. Here 'Lock' is the Feed name.

```
Adafruit_MQTT_Subscribe Lock =
Adafruit_MQTT_Subscribe(&mqtt, MQTT_NAME "/f/Lock");
```

Inside the void setup function, define the buzzer pin and relay pin as an output.

```
pinMode(relay, OUTPUT);
pinMode(buzzer, OUTPUT);
```

Now, the void loop function will continuously check your subscribed feed for data, and it will compare that data with the defined string. If the received and defined string both are the same, then it will perform further operations according to the data.

```
Adafruit_MQTT_Subscribe * subscription;
while ((subscription = mqtt.readSubscription(5000)))
    {
  if (subscription == &Lock)
    {
     //Print the new value to the serial monitor
     Serial.println((char*) Lock.lastread);
  if (!strcmp((char*) Lock.lastread, "OFF"))
     {
       digitalWrite(relay, LOW);
       Serial.print("Door Unlocked");
       digitalWrite(buzzer, HIGH);
       delay(2000);
       digitalWrite(buzzer, LOW);
   }
   if (!strcmp((char*) Lock.lastread, "ON"))
     {
       digitalWrite(relay, HIGH);
       Serial.print("Door Closed");
       digitalWrite(buzzer, HIGH);
       delay(2000);
       digitalWrite(buzzer, LOW);
   }
```

A complete working video and full code are given below.

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
```

```
const char *ssid =  "XXXX";     // Enter your WiFi Name
const char *pass =  "XXXX"; // Enter your WiFi Password
WiFiClient client;
#define MQTT_SERV "io.adafruit.com"
#define MQTT_PORT 1883
#define MQTT_NAME "XXXX"
#define MQTT_PASS "XXXX"// Enter the API key
#define relay D5
#define buzzer D6
Adafruit_MQTT_Client mqtt(&client, MQTT_SERV, MQTT_PORT,
MQTT_NAME, MQTT_PASS);
//Set up the feed you're subscribing to
Adafruit_MQTT_Subscribe Lock = Adafruit_MQTT_Subscribe(&mqtt,
MQTT_NAME "/f/Lock");
void setup()
{
  Serial.begin(115200);
  delay(10);
  mqtt.subscribe(&Lock);
  pinMode(relay, OUTPUT);
  pinMode(buzzer, OUTPUT);
  digitalWrite(relay, LOW);  // keep motor off initally
  Serial.println("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print("."); // print ... till not connected
  }
  Serial.println("");
  Serial.println("WiFi connected");
}
void loop()
{
   MQTT_connect();
Adafruit_MQTT_Subscribe * subscription;
while ((subscription = mqtt.readSubscription(5000)))
    {
```

```
    if (subscription == &Lock)
      {
       //Print the new value to the serial monitor
       Serial.println((char*) Lock.lastread);
    if (!strcmp((char*) Lock.lastread, "OFF"))
       {
         digitalWrite(relay, LOW);
         Serial.print("Door Unlocked");
         digitalWrite(buzzer, HIGH);
         delay(2000);
         digitalWrite(buzzer, LOW);
     }
     if (!strcmp((char*) Lock.lastread, "ON"))
       {
         digitalWrite(relay, HIGH);
         Serial.print("Door Closed");
         digitalWrite(buzzer, HIGH);
         delay(2000);
         digitalWrite(buzzer, LOW);
     }
 }
      }
}
void MQTT_connect()
{
  int8_t ret;
  // Stop if already connected.
  if (mqtt.connected())
  {
    return;
  }
  uint8_t retries=3;
  while ((ret = mqtt.connect()) != 0)
  {
       mqtt.disconnect();
       delay(5000);  // wait 5 seconds
       retries--;
       if (retries == 0)
       {
```

```
            // basically die and wait for WDT to reset me
            while (1);
        }
    }
}
```

Chapter 8

Testing

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all the system elements have been properly integrated and perform allocated functions. The testing process is actually carried out to make sure that the product exactly does the same thing what is supposed to do. In the testing stage following goals are tried to achieve:-

- To affirm the quality of the project.
- To find and eliminate any residual errors from previous stages.
- To validate the software as a solution to the original problem.
- To provide operational reliability of the system.

8.1 Testing Methodologies

There are many different types of testing methods or techniques used as part of the software testing methodology. Some of the important testing methodologies are:

8.1.1 White box testing

White box testing (clear box testing, glass box testing, and transparent box testing or structural testing) uses an internal perspective of the system to design test cases based on internal structure. It requires programming skills to identify all paths through the software. The tester chooses test case inputs to exercise paths through the code and determines the appropriate outputs. While white box testing is applicable at the unit, integration and system levels of the software testing process, it is typically applied to the unit. While it normally tests paths within a unit, it can also test paths between units during integration, and between subsystems during a system level test. Though

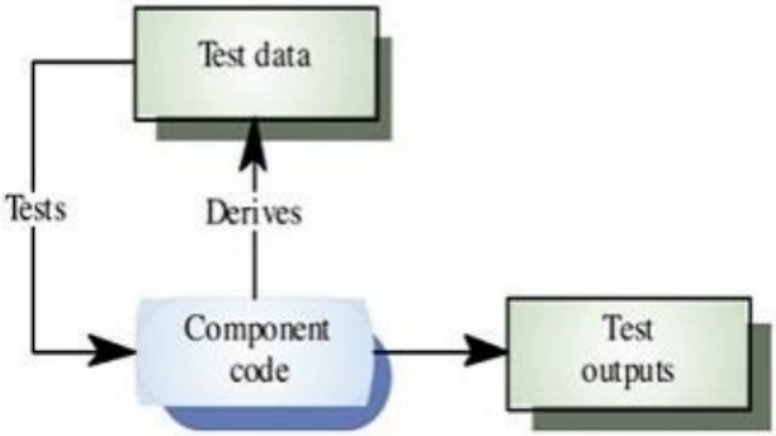

Figure 8.1: White Box Testing

this method of test design can uncover an overwhelming number of test cases, it might not detect unimplemented parts of the specification or missing requirements, but one can be sure that all paths through the test object are executed. Using white box testing we can derive test cases that:

- Guarantee that all independent paths within a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Execute internal data structure to assure their validity.

8.1.1.1 Advantages of White Box Testing

- To start the white box testing of the desired application there is no need to wait for user face (UI) to be completed. It covers all possible paths of code which will ensure a thorough testing.
- It helps in checking coding standards.
- Tester can ask about implementation of each section, so it might be possible to remove unused/deadlines of codes helps in reducing the number of test cases to be executed during the black box testing.
- As the tester is aware of internal coding structure, then it is helpful to derive which type of input data is needed to test the software application effectively.

- White box testing allows you to help in code optimization

8.1.1.2 Disadvantages of White Box Testing

- To test the software application a highly skilled resource is required to carry out testing who has good knowledge of internal structure of the code which will increase the cost.
- Updating the test script is required if there is change in requirement too frequently.
- If the application to be tested is large in size, then exhaustive testing is impossible.
- It is not possible for testing each and every path/condition of software program, which might miss the defects in code.
- White box testing is a very expensive type of testing.
- To test each paths or conditions may require different input conditions, so in order to test full application, the tester need to create range of inputs which may be a time consuming.

8.1.2 Black box testing

Black box testing focuses on the functional requirements of the software. It is also known as functional testing. It is a software testing technique whereby the internal workings of the item being tested are not known by the tester. For example, in a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs.

The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specifications. It enables us to derive sets of inputs that will fully exercise all functional requirements for a program

Black box testing is an alternative to white box technique. Rather it is a complementary approach that is likely to uncover a different class of errors in the following categories:-

- Incorrect or missing function.
- Interface errors.

Figure 8.2: Black Box Testing

- Performance errors.
- Initialization and termination errors.
- Errors in objects.

8.1.2.1 Advantages of Black Box Testing

- The test is unbiased as the designer and the tester are independent of each other.
- The tester does not need knowledge of any specific programming languages.
- The test is done from the point of view of the user, not the designer.
- Test cases can be designed as soon as the specifications are complete.

8.1.2.2 Disadvantages of Black Box Testing

- The test inputs need to be from large sample space. That is, from a huge set of data this will take time.
- Also it is difficult to identify all possible inputs in limited testing time. So writing test cases is slow and difficult.
- Chances are more that there will be unidentified paths during this testing.

8.2 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual

software units of the application. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.3 System Testing

This information contributes towards reducing the ambiguity about the system. For example, when deciding whether to release a product, the decision makers would need to know the state of the product including aspects such as the conformance of the product to requirements, the usability of the product, any known risks, the product's compliance to any applicable regulations, Software testing enables making objective assessments regarding the degree of conformance of the system to stated requirements and specifications.

System testing checks complete end-end scenarios, as a user would exercise the system. The system has to be tested for correctness of the functionality by setting it up in a controlled environment. System testing includes testing of functional and nonfunctional requirements. It helps to verify and validate the system. All components of system should have been successfully unit tested and then checked for any errors after integration.

8.4 Quality Assurance

Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confident that the product quality is meeting its goals. This is an "umbrella activity" that is applied throughout the engineering process. Software quality assurance encompasses:-

- Analysis, design, coding and testing methods and tools
- Formal technical reviews that are applied during each software engineering
- Multi-tiered testing strategy
- Control of software documentation and the change made to it.
- A procedure to ensure compliance with software development standards.

- Measurement and reporting mechanisms

8.4.1 Quality Factors

An important objective of quality assurance is to track the software quality and assess the impact of methodological and procedural changes on improved software quality. The factors that affect the quality can be categorized into two broad groups:

- Factors that can be directly measured.
- Factors that can be indirectly measured

These factors focus on three important aspects of a software product

- Its operational characteristics
- Its ability to undergo changes
- Its adaptability to a new environment.
- Effectiveness or efficiency in performing its mission
- Duration of its use by its customer.

8.5 Functional Test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures: Interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined

Chapter 9

Results and performance analysis

9.1 Snapshots Of Output

In this section snapshots showing the performance of the final code that is generated is shown. Also different scenarios where the WiFi is not connected to the board also if MQTT Server is not connected to the server is shown.

The Figure 9.1 represents the output that is generated on the serial monitor from the ESP8266 board which shows all the analytic such as if the WiFi is connected to the internet and MQTT protocol is connected to its server. and on altering the switches on the server the microcontroller acts and performs the actions such as close door and open door and also prints it to the serial monitor.

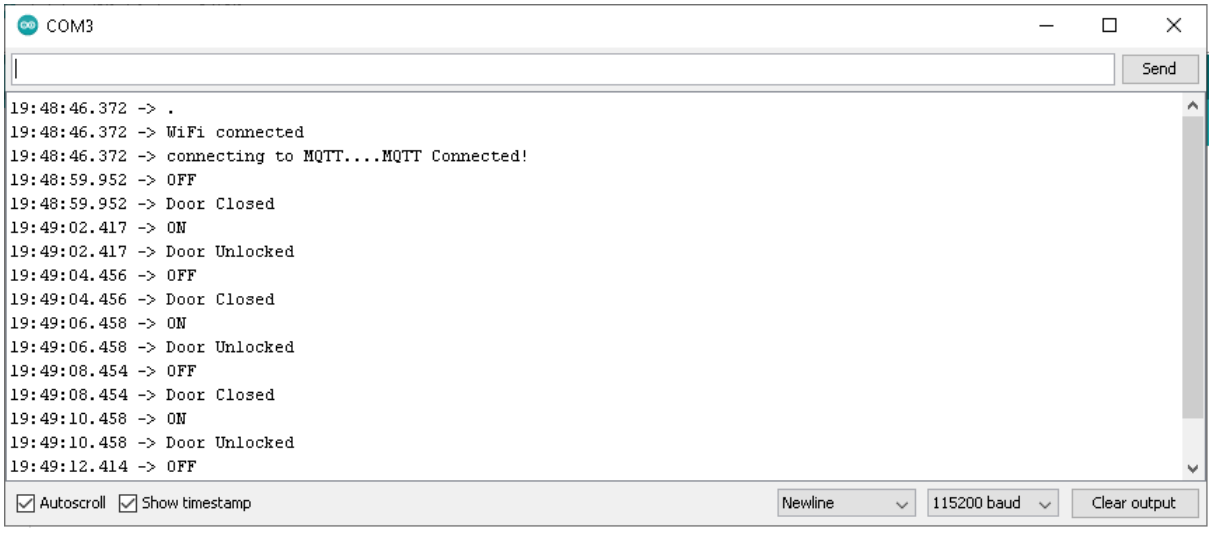

Figure 9.1: Serial Monitor

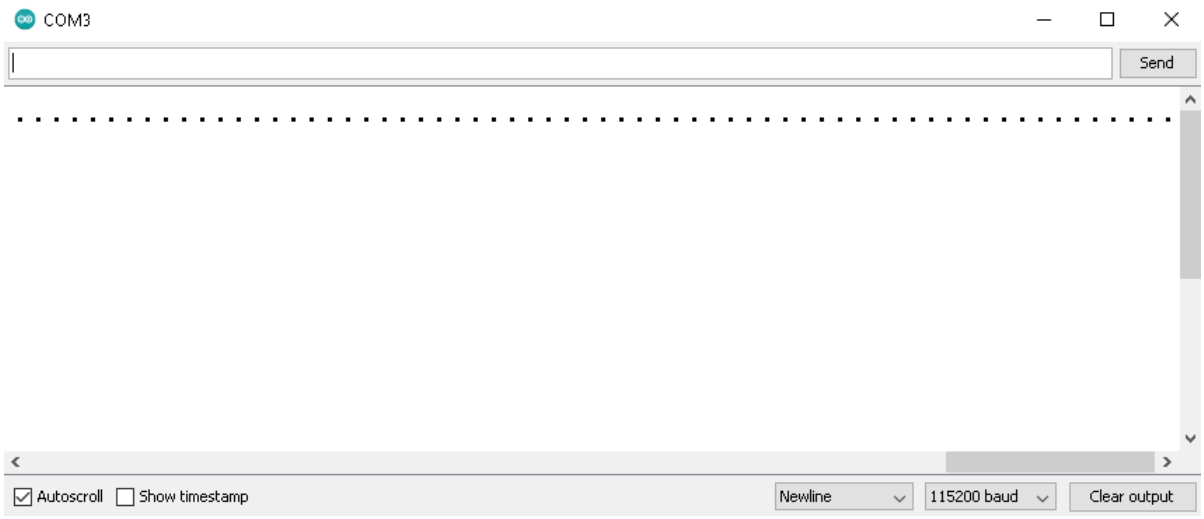

Figure 9.2: WiFi Not Connected

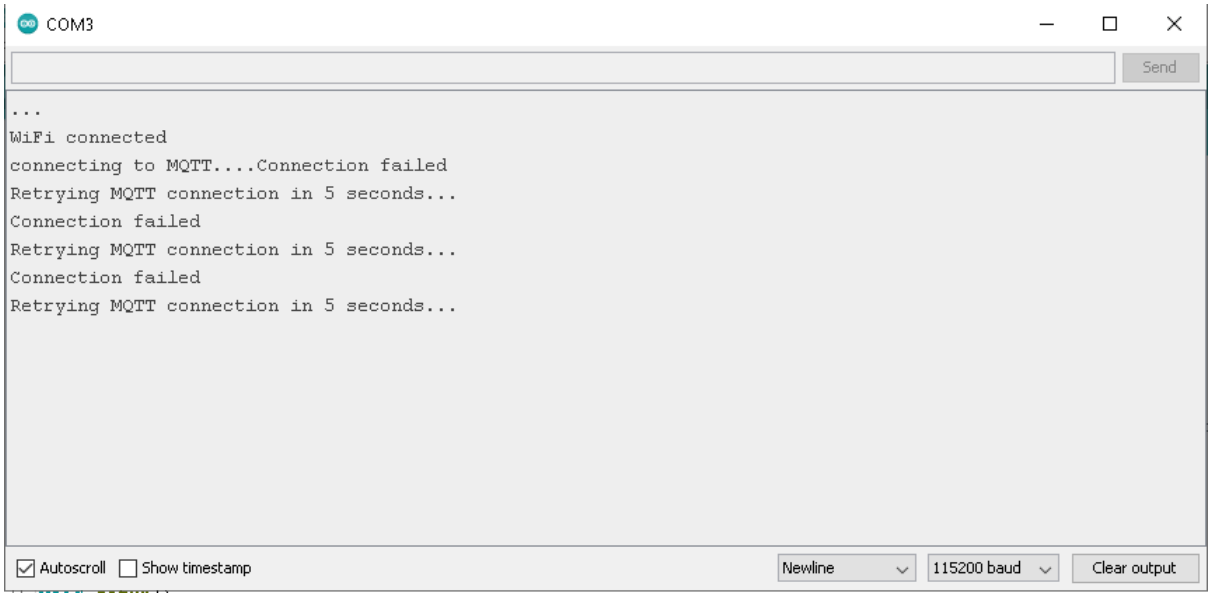

Figure 9.3: MQTT Not Connected

The Figure 9.2 is the case where the ESP8266 is not connected to the WiFi it displays certain message and retries until it is connected to the WiFi.

The Figure 9.3 is the case when the ESP8266 Is Connected to the WiFi But could not be able to connect to the MQTT Server, where if the name and Api of the Server are not initialized properly it keeps Connecting Again After each 5 seconds of time till it is connected else it will display a message of connection Failed.

Chapter 10

Conclusion

This project introduces a technology that will make human lives more versatile with increased security, ease, and to lead an upper class lifestyle inevitably creating our life's much simpler, finer, accessible and more steady. It represents innovativeness for operating a house replacing manual keys with digital codes and knocks for a door lock at home, thereby keeping in view of the growing Security trends in years to come and resolving the use hi-tech manual locks for our existing doors with a Digital Smart Lock. Moreover, the cost of execution of the developed system has been kept at low cost making it reasonable to all who seeks Security for home. As differentiated with most of the used technologies as narrated in the literature review, the Digital Smart home is convenient to execute and manage. It assists IoT, cellular technology and a non-proprietary open-source android system platform. The proposed standpoint registers to operate with Wi-Fi technology for integration, android based App for client ingress, and client testimonial for reliability and authentication, perceive the desired architecture. Hereafter, the sketched structure can be broadened to embrace other characteristics of home modernization and reliability. In addition, these experimentation assignments are often supplemented to reinforce other OS platforms distant from Android.

References

[1] B. P. Statistik, "Statistik Kriminal 2018," in Badan Pusat Statistik, 2018, p. 27.

[2] T. Adiono, S. Fuada, S. F. Anindya, I. G. Purwanda, and M. Y. Fathany, "IoT-enabled door lock system," Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 5, pp. 445–449, 2019.

[3] R. Manjunatha and R. Nagaraja, "Home Security System and Door Access Control Based on Face Recognition," International Research Journal of Engingeering and Technology, vol. 4, no. 3, pp. 437-442, 2017.

[4] P. Wibowo, S. A. Lubis, . Hermansyah, . Hamdani, and Z. Tharo, "Smart Home Security System Design Sensor Based on Pir and Microcontroller," Int. J. Glob. Sustain., vol. 1, no. 1, p. 67, 2017.

[5] Y. T. Park, P. Sthapit, and J. Y. Pyun, "Smart digital door lock for the home automation," IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON, no. February 2009, 2009.

[6] S. Jensen and C. D. Jensen, "Proximity Door Locking," 2016.

[7] Arafat, "Sistem Pengaman Pintu Rumah Otimatis Berbasis Internet Of Things (IoT) Dengan ESP8266," Technologia: Jurnal Ilmiah, vol. 7, no. 4, pp. 262-268, 2016,

[8] N. Kolban, Kolban's Book on ESP32 ESP8266, Canada: Leanpub, 2017.