AUTHENTICATION IN A BODY AREA NETWORK (BAN) USING OPENSSL

by

Josiah Mololuwa Oyinola

Bachelor of Computer Science
Ajayi Crowther University, 2016

A Report Submitted In Partial Fulfillment Of The Requirement For The Degree
Of

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisor(s): Rongxing Lu, Ph.D., Computer Science
Examinining Board: Suprio Ray, Ph.D., Computer Science
Arash Habibi Lashkari, Ph.D., Computer Science

This report is accepted by the Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

December, 2019

© Josiah Mololuwa Oyinola, 2020
Abstract

Internet of Things (IoT), which enable the connection and communication of objects (Things) over the internet, have received considerable attention in recent years. The internet is the main medium (backbone) of communication, while the things are smart devices, machineries, industry level equipment, etc., which generate data to share and process for some intelligent decision making. Ever since the term IoT was coined out and described by Kevin Ashton in 1999, industries have adopted the idea of IoT and began integrating it into their product development process widely.

As a popular application scenario of IoT, body area network (BAN) is one IoT network formed by body sensors, which can sense the health related data, deliver the data to the remote eHealthcare center for a better health monitoring through some gateways over the Internet. However, due to the congestion of the internet and the fast rise in cyberattacks, it is very important to secure packet exchange via advanced level encryption methods to prevent possible session hijacking, Man-In- The-Middle (MITM) attacks, cross site scripting, etc. Fortunately, there are numerous encryption standards available. In this study, in order to address the cyber security issues in body area network, we will be considering the OpenSSL, one popular software library which is open source for user revision.

According to openssl.org, OpenSSL is a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general- purpose cryptography library, and it is licensed under the Apache-style license which makes it free to use by everyone.
Concretely, the report examines the OpenSSL cryptographic architecture, propose and implement a “layer-in-layer” level cryptographic model in a bid to secure our communication while interfacing with generated data between the BAN and IoT. The study involves dissecting the selected authentication process, seeing how it ticks, and why, look into its applications and its setbacks. The ultimate goal of this study is to create an extra layer of security on the same technology and put it to use. It is expected that this strategy will make the authentication algorithm more secure
Acknowledgements

My sincere gratitude goes first to God who out of His loving kindness made it possible for me to complete this report and for his provision since I started my academic pursuit.

I sincerely appreciate my supervisor Dr Rongxing Lu for his intellectual guidance, advice and accessibility. Your height of knowledge and expertise with your students is simply exceptional.

To my Mother Dr Olufunto Olufolake Alonge who gave everything for me to pursue my master’s degree, I thank you so much. I am glad that I am gradually making you proud and seeing you smile at my achievements. My big sister Josiah Motunrayo whose landmarks I have happily followed, I thank you for your assistance both monetarily and academically. I would also like to appreciate my spiritual parents Pastors Segun and Lola Idahor for their prayers and advices, Daddy and Mummy Army—Rtd Major General and Dr US Essien whose inspiration and encouragement has been a positive motivation to my academic pursuit.

I also thank my special friend, motivator and study partner Mr Toluwalase Adebayo Adeniji for encouraging me when I was down, always being there for me when I needed it and leading me academically. You always believed in me even when you didn’t have to and you supported me throughout this project.

I also appreciate Imhoede Jude, Osagie Ayodeji, Nwolisa Zara, Ajetunmobi Tomilola, Ibiyemi Bukunmi, and Isijola Bunmi, for their moral support and for believing in me
# Table of Contents

ABSTRACT.........................................................................................................................II

ACKNOWLEDGEMENTS ........................................................................................................IV

LIST OF TABLES .................................................................................................................... VII

TABLE 1: PATIENT TABLE .............................................................................................. VII

TABLE 2: HISTORY TABLE ............................................................................................. VII

LIST OF FIGURES .............................................................................................................. VIII

CHAPTER 1 INTRODUCTION .............................................................................................. 1

1.1. RESEARCH BACKGROUND..................................................................................... 1

1.2. RESEARCH PROBLEM ............................................................................................ 5

1.3. PURPOSE OF RESEARCH ...................................................................................... 5

1.4. SIGNIFICANCE OF RESEARCH ............................................................................. 6

1.5. RESEARCH OBJECTIVES ....................................................................................... 6

1.6. ORGANIZATION OF REPORT ............................................................................... 7

1.7. DEFINITION OF TERMS ........................................................................................ 8

CHAPTER 2 LITERATURE REVIEW .................................................................................. 10

2.1. INTRODUCTIONS ..................................................................................................... 10

2.2. AUTHENTICATION METHODS [8] ........................................................................ 12

2.2.1 PASSWORD AUTHENTICATION [8] .................................................................... 12

2.2.2 AUTHENTICATION TOKEN [8] .......................................................................... 13

2.2.3 SYMMETRIC- KEY AUTHENTICATION [8] ........................................................ 14

2.2.4 PUBLIC- KEY AUTHENTICATION:THE DIFFIE-HELLMAN AUTHENTICATION [8] . 15

2.2.5 BIOMETRIC AUTHENTICATION [8] .................................................................... 16

2.3. IOT ARCHITECTURE .............................................................................................. 19

2.3.1 PERCEPTION LAYER [22] ............................................................................... 22

2.3.2 NETWORK LAYER [22] ..................................................................................... 22

2.3.3 APPLICATION LAYER [22] ............................................................................... 23

2.3.4 BUSINESS LAYER [23] .................................................................................... 23

2.3.5 PROCESSING LAYER [23] ............................................................................... 24

2.4. IOT SECURITY .......................................................................................................... 24

2.5. IOT PASSWORD AUTHENTICATION ........................................................................ 25

2.6. CRYPTOGRAPHY BASED AUTHENTICATION [8] .................................................. 28

CHAPTER 3 RESEARCH METHODOLOGY ........................................................................ 30

3.1. INTRODUCTION ....................................................................................................... 30

3.2. SYSTEM METHODOLOGY ..................................................................................... 30

3.2.1 METHODOLOGY ADOPTED ............................................................................. 30

3.3. RESEARCH MODEL ................................................................................................. 32

3.4. RESEARCH REQUIREMENTS .................................................................................. 34

3.4.1 FRONT END TECHNOLOGIES ........................................................................... 34

3.4.1.1 JAVASCRIPT (REACT JS) ............................................................................ 34

3.4.1.2 CASCADING STYLE SHEET (CSS) ............................................................... 35
List of Tables

Table 1: Patient table

Table 2: History table
List of Figures

Figure 1 Illustration of a typical body area network
Figure 1.1 Organization of report
Figure 2.0 Password authentication in a web login page
Figure 2.0.1 RSA SecureID devices
Figure 2.2 Diffie Hellman public key authentication
Figure 2.3 Fingerprint based authentication for attendance
Figure 2.3.1 Smart phone fingerprint biometric authentication
Figure 2.3.2 Face detection system
Figure 2.4 Three-layer IoT Architecture
Figure 2.4.1 Three-Layer model and five-layer model in clear view
Figure 2.5 Wireshark Filtering Showing Clear Text of username and Password
Figure 2.6 Encryption and decryption process
Figure 3.1 Encryption Process
Figure 3.2 Decryption Process
Figure 3.3 A login API
Figure 3.4 JSON data structure
Figure 3.5 Code to implement JSON data structure
Figure 3.6.1 Method in the reducer function
Figure 3.6.2 Method in the reducer function
Figure 3.7 Database model
Figure 4.1 Home page
Figure 4.2 Login interface
Figure 4.3. Patient list

Figure 4.4 Create patient account

Figure 4.5. Patient history
CHAPTER 1 INTRODUCTION

1.1. Research background

In recent years, the use of the Internet has become a necessity in many aspects of our everyday life. Internet of Things (IoT) is an emerging technology and it is considered to be the future of the Internet. The idea behind IoT can be considered as an extension of the existing interaction between the humans and applications communicating via a different concept. Due to advancements in communication, Radio Frequency Identification (RFID) innovation, and Wireless Sensor Networks (WSNs), devices and mechanisms in IoT can communicate with each other regardless of time, place or medium. The major breakthrough of IoT is in the formation of smart environments: smart homes, smart transport, smart items, smart cities, smart health, and smart living to name a few. Literally almost every innovation that has the term “smart” as a prefix has an element of IoT attached to it. Furthermore, in business perspective, IoT has enormous potential for various types of organizations and companies, including IoT applications and service providers, IoT platform providers and integrators, telecom operators and software vendors. IoT will also have a major impact in learning experience; especially in a higher education system. With the rapid increase in IoT application use, several security issues have risen rapidly. As devices are becoming part of the internet infrastructure, therefore these issues need to be considered. When almost everything will be connected on the internet, these issues will become more prominent; with continuous internet global exposure will literally disclose more
security vulnerabilities. Such security flaws will be subsequently exploited by hackers, and later can be misused in uncontrolled environments with billions of IoT devices. That is to say that IoT communication as a whole will become a security risk in itself. A study conducted by Hewlett Packard revealed that 70% of the most commonly used IoT devices contain serious vulnerabilities. IoT devices are vulnerable to security threats due to their design by lacking certain security features such as insecure communication medium, insufficient authentication and authorization configurations. As a matter of fact, as IoT grows, the integrity of the system will continue to be a growing concern for both individuals and companies. This leads to a series of potential risks involving information security and data protection, which should be taken into consideration. While there have been methods and ways to keep the integrity of these risks at bay, at local level, the data delivered from these IoT devices are transported to a dashboard or database for storage intended to be read by the users. These data usually contain private sensitive data that should be confidential to only the designated recipient but in the wake of cyber threats, it is very necessary to account for proper authentication methodologies when designing an IoT communication medium such as the body area network.

With today’s growing population, monitoring healthcare systems depicts an important element of the healthcare issue, especially in developed countries where research into this technology remains progressive. Monitoring the health condition of patients and sharing information with hospitals is in great demand, especially as more patients tend to keep pursuing their day to day activities or occupation. The ability to monitor the medical conditions of a patient living far reduces the pressure on doctors, nurses, and the health care system in terms of cost and logistics. One technology that is expected
to play a key role in the patient monitoring faculty is the body area networks (BANs). BANs are highly localized wireless networks that potentially can support a wide variety of medical applications, from monitoring how an implant is functioning, to tracking the health of patients and or senior citizens.

Body area networks connect different IoT devices and sensors delivering information to specific destination over localized networks or external networks. It is the wireless connection of body sensor mechanisms to deliver certain medical data over a network, given the right gateway, over the internet. The term “Body” implies the literal application of this technology in the medical ecosystem from connecting implants within the body of patients to external sensors monitoring heartbeat rates, sonar readings, etc. A network of sensors is placed close to the human body or implanted in some tissues to enable the acquisition of certain specific physiological data which is transmitted to a remote or local server for storage and processing. This allows the medical personnel to continuously monitor the status data of a patient’s health, regardless of patient’s geographical location.

The sensors harnessed in the BAN require accurate sensing, a certain level of signal processing, as well as some wireless characteristics. These sensors can either be transceivers or receivers only. Sensors used within BANs are classified by two main categories:

- A wearable BAN is located within the body space. It consists of lightweight and miniature sensors that allow long term health monitoring, hence, providing a periodic
update of a patient’s health status. Wearable BANs are mostly used for monitoring physiological health data.

• An implantable BAN is located within the tissues of the human body. Implantable BANs harness biosensors and, unlike wearable BANs, are used for more than just monitoring. Implantable BANs are widely used as drug delivery apparatus through a micro-pump or micro-port, insulin, etc.

Fig. 1 Illustration of a typical body area network [33]

There are few challenges that the BAN faces, such as Form factor or size, consumption factor, which deals with the battery life span of the mechanisms, reliability and intelligence, but in this study, we will be looking at security as a major challenge in the authentication process within the network.
1.2. **Research problem**

Like every conventional wireless network, the BAN experiences security threats, either while data is being transmitted from the sensors or being retrieved by the health care personnel for processing and analysis. The fast rise of innovation within the internet space has exposed numerous vulnerabilities to information stored on the internet.

Due to the congestion of the internet and the fast rise in cyberattacks, it is very important to secure packet exchange via advanced level encryption methods to prevent session hijacking, Man-In-The-Middle (MITM) attacks, cross site scripting, etc.

Our motivation for this study is birthed by the fast rise of cyber-attacks on servers within and beyond network boundaries, namely the internet. In tackling these series of threats, numerous algorithms and password hashing techniques have been developed to ensure proper authentication within the networks and ensure that stored data maintains its integrity. As we know that communication is two-way; Data coming from the IoT devices to the server has to encounter some sort of authentication to determine that the right devices are communicating the right way and not by an imposter, and user attempting to access the interface to read said data is authenticated to ensure it is human and not an auto-bot.

1.3. **Purpose of research**

Based on the above discussion, it is important to secure the exchange of data from sensor to interface, server, and dashboard. This will go a long way in securing the patients and keep them at rest of not being misdiagnosed or administered the wrong
serums or medications. Looking at the big picture, without proper IoT authentication, millions of lives hooked on BANs from a minor node to the majority of the internet links could be at the mercy of an ambitious hacker looking to put lives in jeopardy.

For this study we will be looking into the challenges of a commonly used software library called OpenSSL which is a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols and try to build on the technology in order to produce a much secure authentication model for our study.

1.4. Significance of research

The significance of this research is reflected in a number of benefits that the research is expected to yield at the end of this research. As we have outlined earlier that part of the challenges of the BAN is security. Developing an algorithm that will successfully encrypt data during authentication process in a BAN will go a long way in keeping the patients’ safe and ensuring that the right data is transmitted and received. An error in diagnosis or medication can lead to law suits and fines, above all discrediting certain healthcare facilities, so a foolproof encryption system within the BAN will go a long way in securing both raw and processed data.

1.5. Research objectives

The objectives of this research are:

1. To understand the inner working of the OpenSSL encryption standard and how it performs its encryption and decryption of data within the BAN
2. To discover its drawbacks and build on its existing architecture

3. To implement this newly developed model within an interface dashboard run within a simulated BAN

4. To give recommendations based on the research findings.

1.6. **Organization of Report**

![Diagram of report organization]

**Fig 1.1 Organization of report**

From the above figure (1.1), it can be seen that this report is basically divided into five sections.

**Chapter 1: Introduction**

In this section, the purpose is to define why the research is being conducted in terms of what the research is intended to achieve and how it will be achieved.

**Chapter 2: Literature review**

This chapter is designed to present an analysis of past studies in relation to the research topic. This chapter will also highlight relevant theories and models that will be used to address the whole research as well as aid in the development of variables that will be tested in the primary research.
Chapter 3: Research methodology

Research methodology is the third section and it is designed to present detailed analysis of approaches that will be used to conduct the primary research. Such analysis will include the target platform, frameworks used, and the deployment methods and mechanisms involved.

Chapter 4: System Implementation

This chapter presents the analyzed data gathered from the implementation. The results gathered will be further analyzed in relation of the objectives stated within the first chapter.

Chapter 5: Conclusions and Recommendations

This chapter gives an evaluation of the research findings collected. It also gives recommendations as well as describing the limitations of the research and providing recommendations for further research.

1.7. Definition of terms

- **Internet of Things (IoT)** refers to the connection and communication of devices (Things) over a network, either local or over the internet.

- **Body Area Network (BAN)** is the wireless connection of body sensor mechanisms to deliver certain medical data over a network, given the right gateway, over the internet.

- **OpenSSL** is a robust, commercial-level, toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general-purpose cryptography library for programmers and cryptographers. It is
licensed under the Apache-style license which means it is free to use by everybody.
CHAPTER 2 LITERATURE REVIEW

2.1. Introductions

This chapter aims to highlight relevant theories and models that will be used to address the whole research as well as aid in the development of variables that will be tested in the primary research. Understanding Authentication as a Concept

Imagine someone walks up to your home entrance and rings the door bell, the logical response will be to ask for who is at the other end of the door which prompts a response from the guest. When the guest responds, you try to match the voice to all the other voices you have registered in your mind as either familiar or not, assuming the guest is a familiar voice you then go ahead to get the door open. This is no different from how every system operates, considering they were built by humans with the same natural human instinct, either this natural phenomenon is done differently regardless of race or ethnic persuasion, the idea remains the same.

It is not so different for any system that retains a certain level of sensitive data, just as humans are keen to protect themselves from imposters, systems have to be guarded from different sorts of dangers that lurks in the large public space called the internet. These few actions that constitutes our everyday security measure as humans have evolved over the years and same has happened with systems and as we look into different concepts. In this review, we will be using this analogy to explain few measures in different scenarios.

To put in simple terms, Authentication is the process of validating the user’s identity [5]. Authentication asks the questions of if an entity is who they claim to be, just as
explained in our earlier analogy, authentication is the stage where the house owner listens to know if the guest is someone familiar to them or not. But authentication as a security mechanism is not enough to do the job in the sense of its singular context, it has two other ideas that revolve round it to make it a complete mechanism. These are identification and authorization. Authentication cannot simply occur without the first and the final process.

Identification, authentication and authorization are three interrelated concepts, which form the core of any system that requires security [6]. Identification is the communication of an identity to a system, just as the guest communicates who they are to the house owner [6]. In the technology space, the mode of identification could be in different forms, either by voice, biometrics, certain designed hardware, such as QR coded tags, login parameters, or an email address etc. Authentication provides the answers to the questions of who the guest or user is and if they are who they say they are [6]. Because of this, authentication represents one of the most promising approaches concerning trust and security for most applications [6]. Authorization is a process of delivering access to the system based on the vetted identity [6]. Authorization defines the access privilege the user can be afforded within the system, just as we cannot afford the plumber and our spouse the same level of privilege, it would constitute a total disaster.

On the internet, authentication follows the same parameters but on a much complex scale, this is basically due to the fact that the process might not be taking place in a physical manner unlike our earlier analogy of the visiting guest. There are information systems to be protected, vaults, safe rooms, money boxes, etc, and just as there are
different ways we check our guests before they are welcome into our homes, there are different methods that authentication presents itself.

2.2. Authentication Methods [8]

2.2.1 Password Authentication [8]

A password authentication is an authentication method used in a client-server system format to authenticate users before providing them access to data [8]. Password authentication is the most used methodology in authentication protocols [8]. This system functions in a way such that the user provides their username and password and it is compared with the value stored within the database [8]. This authentication method is widely acceptable by users since passwords can be easily memorized but also the most susceptible to a series of dictionary attacks [8]. This authentication method could be textual as seen in Figure 1.2 or could be graphical.
A graphical password allows users to draw or select their passwords from images, in an order specified by the system [8]. This system usually presents itself via a graphical user interface (GUI) and this approach is commonly known as graphical user authentication (GUA) [9]. Unlike the textual password approach, this is immune to dictionary attacks where the attacker uses a random selected words based on certain patterns to iterate till a permutation works.

**Fig. 2.0.1 Graphic based authentication approach [9]**

2.2.2 Authentication Token [8]

Stolen passwords are displayed as a significant threat to today’s enterprise systems
It is apparent that static usernames and passwords are not sufficient to provide adequate security. As the corporate network is increasingly used to store sensitive data, there is need for user access control [10].

An authentication Token is a portable device used for authentication, and it allows only authorized users into a secured network system [8]. Passwords generated by tokens are different each time the user requests one, so an intercepted password is useless as it is never used again [10]. Aside the normal changing token passwords, the token method is usually supported by a username or a static password or personal identification number (PIN) [10]. Token devices range from key chain, to cookie-sized hardware devices to software programs running on different servers [10].

There are several token systems, among these are the RSA SecureID Token Cryptocards, Challenge Response Token, and Time based Tokens [8].

Fig. 2.1: RSA SecureID devices [8]

2.2.3 Symmetric- Key Authentication [8]

In symmetric key authentication case, the user shares a single secret key that is embedded within a token with an authentication server [11]. The user is authenticated
by sending to the authentication server his/her username together with a randomly generated challenge message that is encrypted by the secret key. Then, the user is considered authenticated user if the server can match the received encrypted message using its shared secret key. [8]

2.2.4 Public-Key Authentication: The Diffie-Hellman Authentication [8]

The primary researchers to find and publish the ideas of Public Key Cryptology were Whitfield Diffie and Martin Hellman from Stanford University, and also Ralph Merkle from the University of California at Berkeley [12]. Coincidentally, the two gatherings were working autonomously on the same issue - Diffie and Hellman were working on public key cryptography while Merkle on public key distribution [12] – When they discovered each other’s work, they acknowledged the collaboration of their methodologies [12].

Diffie-Hellman is a mathematical algorithm that permits two PCs (users) to produce an identically shared secret on both systems, despite the fact that those systems might never have at any point communicated with one another [12]. That shared secret can then be utilized to safely exchange a cryptographic encryption key. That key then encrypts traffic between the two systems. This goes a very long way in protecting data in transit from Man-In-The-Middle (MiTM) attacks as an hacker doesn’t see in plain context what is transmitted but in an unreadable scrambled format. Key exchange makes it possible to read when data is delivered to the intended recipient.
The key exchange is an important method in public-key Cryptography for providing authentication cryptographic service. In DH, keys are exchanged between the users according to Cryptography protocols which are based on the key exchange problem.

![Diffie Hellman Key Exchange](image)

**Fig. 2.2: Diffie Hellman public key authentication [8]**

### 2.2.5 Biometric Authentication [8]

A biometric authentication is a digitizing measurement of a physiological or behavioral feature of a human being [8]. In other words, it can theoretically be used to distinguish one form from another based on certain collected characteristic features. Biometric authentication systems can be categorized as face detection authentication system, fingerprint authentication system, Iris authentication system, and voice authentication system [13].

1) **Fingerprint recognition [8]:**

A fingerprint system uses an electronic device to capture a digital image of the fingerprint pattern. This captured pattern in fingerprint system is called a live scan and
is then digitally processed to create a biometric template (that is mainly consisting of finger features). The biometric features will be later stored and used for matching process [8]. Finger print recognition can be found on mobile devices, attendance systems, door handles, automated teller machines, etc.

Fig. 2.3: Fingerprint based authentication for attendance

Fig. 2.3.1: Smart phone fingerprint biometric authentication

2) Voice biometric authentication [8]

Voice biometric authentication is the use of a spoken vocal pattern to recognize the
identity of a user when authenticating into a system [8]. While the voice biometric authentication is becoming a fast widely deployed method, it presents the concerns when it comes to vocal mimicking. Different people with different voices but certain people can mimic certain vocal patterns that could be consistent with certain vocal patterns previously registered within the system. Voice recognition is categorized into five types namely speaker dependent system, speaker independent system, discrete speech recognition, continuous speech recognition, and natural language.

3) Face detection [8]

Face detection and recognition systems are two complementary scenarios [13]. Face detection uses technology that learns algorithms to allocate human faces in digital images, simply put, allocating human faces certain corresponding values based on facial characteristics. As shown by Figure 1.8, face detection algorithm focuses and determines the facial features and neglects everything else within the digital image [13, 14]. However, many facial detection methods have been presented such as Viola and Jones face detection [15], face detection based Adaboost [16], semi-supervised learning for facial expression recognition [17], and etc.

![Fig. 2.3.2 Face detection system [8]](image_url)
This technology is used to identify any given face image using the main features of the given face [18]. Normally, face recognition process works after face detection process to identify the detected face by comparing the detected faces with the stored facial images. Generally, different artificial neural network algorithms have been proposed such as; feed forward back propagation neural network (FFBPNN), cascade forward back propagation neural network (CFBPNN), function fitting neural network (FitNet), and pattern recognition neural network (PatternNet) algorithms [18]. Volkan [19], Weihua and WeiFu [20] are recognition systems applied on neural algorithms.

4) Iris Authentication [8]

Actually, iris and fingerprints are parallel in their uniqueness technology. Generally, the result of the iris usage in authentication is presents that iris is one of the best ways of meeting high risk situations. Iris recognition software is currently in wide use at airport borders. As well as, it is also widely used at many other industries for doing authentication.

2.3. IoT Architecture

The Internet of things (IoT) is a collection of many interconnected objects, services, humans, and devices that can communicate, interact, share data to achieve a common or designated goal in different areas of applications. Devices in IoT follow an Identity Management methodology to be identified in a collection of similar and heterogeneous devices [22]. While traditional Internet connects people and their general smart devices to a much larger network, IoT follows a different approach in
which it provides Machine-to-Machine (M2M) and Human-to-Machine (H2M) connectivity, for heterogeneous types of machines in order to support variety of applications (e.g., identifying, locating, tracking, monitoring, and controlling) [23]. Like a body area network managing a vast number of IoT sensors and data acquisition technologies and objects providing medical data to a control server(s) residing somewhere in the cloud or a private space, connecting a huge number of heterogeneous machines leads to a massive traffic which in turn means large data management and processing. Therefore, the Transmission Control Protocol over Internet Protocol (TCP/IP) architecture, which has been used for a long time for internetwork connectivity, does not fit the needs of IoT regarding various aspects including privacy and security (e.g., information privacy management, machine’s safety, data confidentiality and integrity, data encryption, and network security, and quality of service. [23]

However, according to few researchers [24], the IoT operates mainly on three layers. These layers are called the Perception, Network, and Application layers. Each layer of IoT has inherent security issues associated with it. Fig. 1.9 shows the basic three layer architectural framework of IoT with respect to the devices and technologies that encompass it [22].
Another set of researchers decided to define the architecture in a much more expansive manner and came up with the Five-Layer architecture model [23]. While this model only gives a much expansive approach to the three-layer architecture model, they are in no way contradicting each other but a way to provide more insight to the deeper concept surrounding the three-layer architecture. Same concept was adopted during the creation of the Open Systems of Interconnection popularly known as the OSI model in internetworking which comprises of seven layers and its latter version only comprising of a much concatenated four-layer model.

Fig. 2.0 provides a diagram of the five-layer architecture and how it maps into the three-layer system.
2.3.1 Perception Layer [22]

The perception layer is also known as the “Sensors” or “physical” layer in IoT [22]. This layer interfaces directly with the environment to acquire data with the help of sensors, actuators, relays, etc. This layer takes care of detection, collection, processing, of raw data and then transmits it to the network layer. Examples of physical properties that could be sensed are temperature, speed, location, humidity, and all these are done using different sensor technologies such as GPS, or NFS [23].

2.3.2 Network Layer [22]

The network layer of the IoT serves the function of data routing and transmission to different IoT hubs, devices, and/or servers over the Internet or local networks [23]. This
is the layer that connects the perception layer and application layer together by getting data from the perception layer and transmitting it to the application layer through various technologies. At this layer, cloud computing platforms, Internet gateways, switching, and routing devices etc. operate by using some of the very recent technologies such as WiFi, LTE, Bluetooth, 3G, 4G, 5G, Zigbee etc [22-23]. This layer is also responsible for data management, and serves as the intermediary between IoT nodes by aggregating, filtering, and transmitting data both ways from different sensors. [22]

### 2.3.3 Application Layer [22]

The application layer is responsible for delivering application-specific services to the user [23]. The importance of this layer is that it has the ability to cover numerous markets and its needs (e.g., smart cities, smart cars, smart homes, health care, building automation, smart metering, etc.). This is the layer that guarantees the confidentiality, authenticity, and integrity of data because of its various applications from this point. This layer also harbors the business layer which we will be looking into in the next sub-sub-topic [22].

### 2.3.4 Business Layer [23]

This layer covers the overall IoT system actions, features, behavior and functionality. The application layer transmits the processed data to the business layer whose major role is to build business models, graphs, and flowcharts to analyze data, in order to assist in business intelligence and decision making about business strategies and road-
maps [23]. The business layer is majorly concerned about how the data is presented on the outside. This layer is born from the five-layer IoT architecture because this model is a Service Oriented Architecture (SOA) [23]. It sees the model from a product point of view.

2.3.5 Processing Layer [23]

This is the engine room arm of the transport layer, in the five-layer model cut out as the processing layer. This layer is responsible for storing, analyzing, processing data that is acquired by the sensors at the perception layer [23]. As earlier said, we could refer to this layer as the “middleware” layer [23].

2.4. IoT Security

Just as data security goals are confidentiality, authentication and integrity, same applies for IoT devices. As the use of IoT devices increase, applied in vehicles, homes, automation outfits, they also present with life-threatening security issues. One single tampered data might be the end of a life or beginning of wrong business decisions for large organizations and factories such as Amazon. Various hacking scenarios presented in the past years [26, 27] illustrate the level of harm that could result from a security breach, especially with the development and large adoption of IoT applications dealing with sensitive information (personal, industrial, governmental, etc.) [23]. In this study we will be focusing more on how to secure the IoT password authentication exchange process and make it more C.I.A compliant.
2.5. **IoT Password Authentication**

As discussed earlier, password authentication system is the most commonly accepted and used method in authentication protocols [8]. An IoT system in a body area network like every other programmed system uses the password authentication method to gain access into a certain back-end server or database system. In order to allow successful transfer of data packets from the perception layer to the application layer and back, it is very important to put in appropriate authentication methodology. This is made possible by the use of an Application Programming Interface (API).

An application programming interface (API) is a specification intended to be used as an interface by software components to communicate with each other [21]. An IoT device cannot physically type in username and passwords into an interface to pull data or post data from the database, but can have the credentials embedded using certain trigger methods. These trigger methods are written within the user end application, the alert systems, and other functional dashboards so data is transmitted in an automated fashion.

API requests are usually in two categories, we have the GET request and the POST request. The GET request queries the system for certain data to be displayed over a dashboard or back to the IoT device. The GET request is a standard prepared method that is plugged into the programming sequence of the system to be triggered at certain points or intervals. Consider the GET request of a given bulk messaging service that allows its users to access their bulk messaging feature from within their own corporate internal application without having to visit the bulk messaging website application to transact their business.
This request can be used to get the user messaging unit balance from the platform database systems without having logged into the web based platform itself. This method can be used within an IoT system to help authenticate against the platform to gain access to the system. These kinds of APIs are usually hard coded within the system from development so as to avoid tamper.

Consider an example of a POST request, where the system sends an auto message out without manual authentication.


API usually includes a description of a set of class definitions, with a set of behaviors associated with those classes [21]. In this case we see definitions of variables such as username, password, sender, message, etc within both the Get and POST request. These variables are defined with how they are to behave when interacting with the core system from another device or system or software.

The downside however, is these data are transmitted in plain text over the transport layer and could be seen by an intentional hacker trying to spoof through the network.
Fig. 2.5 Wireshark Filtering Showing Clear Text of user Name and Password [28].

Fig. 2.1 is an example of extracting a user name and password data in a Wireshark tool by filtering the HTTP protocol, which shows the clear text user name and password as shown in the rectangle highlighted box which shows the user name is "Ibrahim_Diyeb" and password is "yemen_123". The filtering command is "http.request.method=="POST"" [28].

This filtering presents a very critical case of passwords being intercepted during transport within the internet or a network and informs the need for a conscious network administrator to develop algorithms to mask the credentials being transported or encrypt the data being exchanged. To encrypt is to create some form of misguided representation of an actual data that is being transported over a network so as not to be easily discovered by hackers.
2.6. **Cryptography Based Authentication** [8]

Cryptography is the art of transmitting data from one point to another through a medium, in such a way that only the receiver who has the secret key can read the encrypted data, which could be text, images, video, even a live broadcast [8].

As the internet community continues to grow in size, new innovations and initiatives, social media and online advertisement, the need for proper encryption has not been more needed. Cryptography provides information security to our everyday online applications, social media platforms and this method shields our data from intentional hackers and data extractors.

To implement data privacy, the information intended can be encrypted to remain secret by using cryptographic methods. These methods scramble the data so that other users or hackers cannot make of what the information actually means.

![Encryption and decryption process](image)

**Fig. 2.6 Encryption and decryption process**
As seen in Fig. 2.2, the word DEVICE is sent from one end to another, but over the transportation process the text is encrypted with the letter K to give the word a different context, this letter K becomes the key needed to unlock this true text and is shared between both parties. This process of scrambling the plain text is called encryption and the process of rearranging is called decryption. There are numerous encryption methods being developed and implemented within each application and as we continue to move further in this study we shall be exploring and building on top of a common model.
CHAPTER 3 RESEARCH METHODOLOGY

3.1. Introduction

Research methodology refers to the methods and procedures used to gather information or data for the purpose of making meaningful observations. In other words, research methodology is concerned with how a certain research is being carried out. The choice of which research methodology to employ is dependent upon the nature of the research problem in itself. This chapter discusses different research methodologies and explains which methodology was chosen for the study and why.

This chapter also contains the research methods, model, and requirements being employed to carry out the research.

3.2. System Methodology

In computing terms, a methodology is “an organized, documented set of procedures and guidelines for one or more phases of the systems life cycle”. It provides the project with a defined structure and focus. To effectively manage this project, it is important that the methodologies applied are appropriate. This is vital because incorrect methodologies, or a complete lack of them, can cause project management problems such as slippages and wasting of time on tasks that have no purpose.

3.2.1 Methodology Adopted
Fig. 3.1 Encryption Process

The IoT device sends a plain data to the server, however, in between is the OpenSSL layer which encrypt the plain text to a non-readable scrabbled format (this is achieved by using the doctor’s unique non-readable secret key that was generated when creating account for the doctor).
3.2 Decryption Process

From the client system, the doctor makes a request to the server to get the patient medical record, but before the response is sent back to the doctor’s system, our OpenSSL layer takes its course by decrypting the non-readable data into a readable plain text using the doctor’s unique secret key after which the response is sent to the doctor’s system.

3.3. Research model

The concept of IoT authentication in a body area network will be explored by understanding the need for IoT authentication in the first place. The gains of information security cannot be overemphasized and the need for authentication hasn’t been over stressed in many different ways. Just as we purchase locks and barricades to keep us safe, our digital presence requires some level of security. An IoT device such as a patient’s heart sensor that periodically sends data to a remote server which then in turns pings the appropriate doctor in the case of an emergency. A simple tamper within this system can mean the end of a life, hence multiply that by the average number of elderly states men and women using heart or pulse rate sensors to keep the ER close by in a manner of figure of speech.

As IoT adoption grows the public network space will have more to do in the area of guarding user data and accessibility by deploying different kinds of authentication methods to keep imposters from manipulating data. A good way to go about this is to employ the concept of cryptography.
When we are in the midst of well-known friends, we speak in a manner we only understand, a jargon formulated by our group alone. To other people around, we might not make sense but to friends it means a lot, this is just a simplistic way to put what cryptography implies.

Cryptography is a method of transferring private information through a network transport media, so only the receiver who has the secret key can read the encrypted messages which could be documents, spread sheets, presentations, multimedia files or any data based information. [8]

This means that information sent from one point to the other will seem as though not making sense but can only be read by the receiver sharing the same key with the sender. This ensures that data is kept intact and not tampered with.

Highlighted below is how cryptography ensures these services [8]:

- **Authentication**: Authentication is used to provide the identity of an entity.
- **Confidentiality**: Confidentiality is a service used to guarantee information accessibility and right privileges are delivered to the right set of entities.
- **Integrity**: Integrity is a service used that guarantees that the information is not tampered with from after leaving the sender all the way to the receiver.
- **Non-repudiation**: Non-repudiation is a service used to confirm the involvement of an entity in a certain form of communication and prevents any party from denying the sent message.
- **Accessibility**: Accessibility is a service put in place to allow the use of information resources by authorized entities. [8]

The process of employing cryptography methodologies is called encryption and we
shall be looking at a particular encryption method and employing it to our authentication method. Our goal in this study is to implement a secure IoT authentication system that ensures the integrity of the data sent by the IoT device, which in our case will be a heart pulse rate sensor.

3.4. Research Requirements

For this research study we implement an actual cryptographic method for our authentication approach which in turn, ensure that data being transmitted from the sensor to the remote database remains that same and not tampered with during the transfer process.

We will briefly be looking at the few requirements for these implementations:

3.4.1 Front End Technologies

Front end-is a terminology used to describe program interfaces and services relative to the initial users of these interface and services. It usually refers to the client side of an application. This is the side that users tend to interact with on daily basis.

3.4.1.1 JavaScript (React JS)

JavaScript is a scripting language that is browser-based and was developed by Netscape to allow web masters/authors to add interactivity and enhances behavior of websites. A number of the dynamic behavior which can be generated by JavaScript is validating form input, performing specific actions e.g. on selecting a combo box field
or after a mouse hovers on an element, adding timestamps etc. JavaScript is an open language, and anyone can use it. It additionally shares several of the features and structures of the Java programming language, though it is not extremely associated to Java. It was developed independently.

### 3.4.1.2 Cascading Style Sheet (CSS)

CSS is a style sheet language used to describe presentation and layout of HTML tags. CSS is used to enable separation of document content from document presentation. This refers to the separation of document presentation aspects such as colors, layouts and fonts from the actual document content. CSS helps us achieve layout design and control much easier.

### 3.4.1.3 MySQL Workbench

MySQL workbench is a tool for analysis MySQL databases. It is used for generating database model.

### 3.4.2 Authentication Application Programming Interface (API)

An application programming interface is a program with allows components of different (hardware and software) or same type (software and software) to communicate with one another. There is a misconception about API as most people only see it as a web-based API, which takes in request and also returns data, most likely in JSON or XML format. An API is not a database or even a server, it is the code that governs the access point(s) for the server. An API could be language-dependent or language-independent in the sense that when written in a specific
language that is only available to that language, we say it is language-dependent and when other programming languages can call an API, we say it is language independent. For example, when integrating google Firestore API into an application, the developer does not need to know the underlying logic behind the API.

```php
if($data['action'] == "login"){
    $payload = array();
    $obj = json_decode($json);
    $doctor = array();
    foreach($obj as $key => $val){
        array_push($doctor, $key);
    }
    if(in_array($data['email'], $doctor)){
        foreach($obj as $key => $val){
            if($key == ($data['email'])){
                if($val->password == $data['password']){
                    $payload['isAuth'] = true;
                    $payload['msg'] = 'login successful';
                    $payload['user'] = $val;
                } else{
                    $payload['isAuth'] = false;
                    $payload['msg'] = 'Wrong login credentials';
                }
            }
        }
    } else{
        $payload['isAuth'] = false;
        $payload['msg'] = 'Wrong login credentials';
    }
}
echo json_encode($payload);
```

Fig. 3.3 A login API

3.4.3 A Wireless Body Area Network (WBAN)

WBAN is a technology which consist of intelligent bio-medical device that is either worn or implanted in human body. The word was first mentioned by Van Dam et al [29], which also allows several researchers [30-34] carry out more work thereafter.
The device worn or implanted has two main parts - sensor and actuator. The sensor is the one that measures certain parameters of the human body (e.g. heart pulse rate, temperature, blood pressure, etc.) while the actuator does some actions (e.g. inject patient or start pumping blood) based on the sensor’s data. In WBAN, the sensor needs to send their data over a network to a server (mostly medical server) where it can be saved. The doctor then analyze the data received and do what is needful.

Over the past few years there has been a need to develop a system capable of monitoring human vitals and organs remotely, with the use of body sensors. It was necessary to have heart rate and pulse sensors because having a way to monitor patients in real time was necessary. The Wireless Body Area Network (WBAN) is such a network that connects various body sensors together for appropriate transmission of data and can also support transmission and query of real time data remotely depending on how it is implemented. Traffic that can be supported includes data, voice, video depending on the core feature of the body sensor deployed.

The WBAN can be perceived more as a concept birthed based on application. This implies that the Wireless BAN still is subject to media such as bluetooth, wifi, zigbee, etc as its primary means of data transport, its application in the health sector as a means for connection and transmission of data from human vital monitoring sensors makes it what it is.

As stated earlier in this sub-topic, the wireless BAN could be any technology that is not bound by a physical media, most times, it is usually wifi of bluetooth, largely wifi. Sensors possessing the feature to enable them communicate data through a wireless network cluster to a remote server or database for review and monitoring purposes,
connected through few lines of code or in-built configuration from the manufacturer, or easy pairing in the case of bluetooth.

In this report we introduce the concept of a wireless BAN as a means to transmit the patient’s heart rate to a remote server for the doctors on call to monitor and advise on next steps. Doctors access this data via a login interface served from a web server hosted within the network. Body sensors connect via a working microcontroller that carries out this data from the local network to the internet (public network). To make this work there will be the need of a modem circuit on the microcontroller to provide bandwidth to transmit data via the internet to the remote server.

3.4.4 Cryptographic method (Open SSL)

OpenSSL is the most well-liked open-source cryptography library that provides loads of tools, libraries, algorithms concerning cryptography. It is mostly known with Secure Socket Layers (SSL) and Transport Layer Security (TLS) protocols. The main purpose of encryption is to hide sensitive information from certain guests or people (most importantly intruders) as this gives some level of confidentiality to our data. OpenSSL allows us to do all these and this could be done using an algorithm among others. The supported algorithms are listed below

- Rivest Shamir Adleman (RSA)
- Digital Signature Algorithm (DSA)
- Advanced Encryption Standard (AES)
- Data Encryption Standard (DES)
- Triple Data Encryption Standard (3DES)
The library includes tools for generating RSA private keys and Certificate Signing Requests (CSRs), checksums, managing certificates and performing encryption/decryption. OpenSSL is written in C, however wrappers are obtainable for a large set of programming languages.

The general algorithm for achieving OpenSSL in PHP is given below:

```php
openssl_encrypt($string, $method, $key, $options, $iv)
```

- `$string` - The text/data to be encrypted
- `$method` - A cipher method chosen from `openssl_get_cipher_methods()`
- `$key` - Your encryption key (reproducible, but kept private)
- `$options` – `0 || OPENSSL_RAW_DATA || OPENSSL_ZERO_PADDING`
- `$iv` – This is a single-use unique Random Initialization Vector (a.k.a. "IV", or "nonce") OpenSSL takes encryption key which generates a hash from the encryption algorithm chosen by the developer. The hashing is used to create one-way values which cannot be converted back.

This is useful for creating values to identify given data. OpenSSL supports the following hash algorithms (MD4, MD5, SHA, SHA256, RC4, DES, HMAC etc.).

Likewise, to achieve OpenSSL decryption, the settings provided above – during encryption (the cipher method, encryption key and the initialization vector) would be required. This can be achieved using:

```php
openssl_decrypt($crypted_token, $cipher_method, $enc_key, 0, hex2bin($enc_iv))
```

If the key does not match, then the data will be assumed lost because the OpenSSL will not return anything.
Our implementation was achieved by creating a layer on the existing OpenSSL layer for our data encryption. A JSON – which stands for JavaScript Object Notation file was used to keep doctors record. JSON is a lightweight data-interchange format for storing and transporting data most especially data sent from a server to a web page.

```json
"boladebode@gmail.com": {
  "id": 1,
  "name": "Akinniyi Bolade",
  "email": "boladebode@gmail.com",
  "password": "password",
  "discipline": "Heart Surgeon",
  "key": "3Q==mololuwab6834bbdf048c5869dbb835a0f0356fb"
},
"Kavanaugh@gmail.com": {
  "id": 2,
  "name": "Kavanaugh",
  "email": "Kavanaugh@gmail.com",
  "password": "password",
  "discipline": "Brain Surgeon",
  "key": ""
}
```

**Fig. 3.4 JSON data structure**

The code to achieve this is found in the figure below
To get patients records, we call the above methods in our reducer function as follows

```java
public static function encrypt_lot($data, $separator) {
    $cipher_method = 'aes-128-ctr';
    $enc_key = openssl_get_iv($plain_text, 'SHA256', TRUE);
    $enc_iv = openssl_random_pseudo_bytes(openssl_cipher_iv_length($cipher_method));
    $encrypted_token = openssl_encrypt($data, $cipher_method, $enc_key, $enc_iv, $separator); bin2hex($enc_iv);
    unset($data, $cipher_method, $enc_key, $enc_iv);

    return $encrypted_token;
}

public static function decrypt_lot($encrypted_data, $separator) {
    list($encrypted_data, $enc_iv) = explode($separator, $encrypted_data);
    $cipher_method = 'aes-128-ctr';
    $enc_key = openssl_get_iv($plain_text, 'SHA256', TRUE);
    $data = openssl_decrypt($encrypted_data, $cipher_method, $enc_key, 0, hex2bin($enc_iv));
    unset($encrypted_data, $cipher_method, $enc_key, $enc_iv);
    return $data;
}
```

Fig. 3.5 Code to implement JSON data structure

Fig. 3.6.1 Method in the reducer function
With every encryption algorithm lies a hashing system or pattern, which is normally achieved by some standard hashing algorithms (MD5, SHA etc). OpenSSL allows this as well as part of the cryptography process, it is important to remember that the OpenSSL function requires a secret key, which is to be kept away from the public. But in our approach, after adopting one of these algorithms, we generated a key for each doctor upon registration. The doctor’s key now becomes the new secret key for a certain patient or every patient under the doctor.

Each registered doctor on the system has a unique key which is used as a separator for the initialization vector of the OpenSSL algorithm. This means doctor A cannot see Doctor B’s patients’ data.
3.4.5 Application Database Systems

A database is a collection of related data that represent some facet of the real world. A database could be a software or a file that stores data for a certain task. The data stored can be a few entries, or rows, that make up a simple inventory system of products, quantities, prices. In contrast, the database can also contain millions of records that describe the products, orders, returned items of a medicine shop.

3.4.5.1 What is a DBMS?

Database Management System (also known as DBMS) is a software that manages database of a system. It allows for storing, retrieving, manipulating and removal of users’ data or database schema (field format, field names, length, etc.) using the appropriate security measures as provided by the user. The DBMS can be accessed either via a software (using the API provided by the database developer) or through a database administrator.

Some examples of DBMS include

- MySQL
- SQL Server
- Oracle
- dBASE

Characteristics of Database Management System

- Provides security and removes redundancy
- Self-describing nature of a database system
• Insulation between programs and data abstraction
• Support for multiple views of the information in the database
• Sharing of information and multiuser dealing process
• DBMS permits entities and relations among them to make tables.
• It follows the ACID idea (Atomicity, Consistency, Isolation, and Durability).
• DBMS supports multi-user environment that allows users to access and manipulate data in parallel.

3.4.6 Database Schema

Database schema is the overall design of the database. Database schema is the logical design of the database. The database schema for this project is shown below:

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Null</th>
<th>Default</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Primary)</td>
<td>int(11)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>varchar(40)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>age</td>
<td>int(11)</td>
<td>Yes</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>occupation</td>
<td>varchar(50)</td>
<td>Yes</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>location</td>
<td>varchar(100)</td>
<td>Yes</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>doctor</td>
<td>int(11)</td>
<td>Yes</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Patient table
An entity relationship model (ERM) is a diagram that shows the conceptual representation of our data. It is a database modeling method used by software engineers as part of their engineering diagrams to illustrate database schema, relationships, requirements, types and every necessary details in the database to be worked on. Tools like Enterprise architect, MySQL workbench are used for creating this model and diagrams. Diagrams created by this process are called Entity-Relationship Diagrams, ER Diagrams, or ERDs.

### Table 2: History table

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Null</th>
<th>Default</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Primary)</td>
<td>int(11)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>patient</td>
<td>int(11)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heart_pulse_rate</td>
<td>varchar(255)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>illness</td>
<td>varchar(255)</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>date_time</td>
<td>varchar(30)</td>
<td>Yes</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

#### 3.4.7 Entity Relationship Diagram
Due to certain constraints in this study we have improvised a few technologies and simulated much more of the system to be designed. To begin with we have developed a sensor data management system written in PHP that connects to a remote database and capable of authenticating IoT transmitting devices and humans (doctors) likewise. We have proposed to deploy this management system on a publicly hosted server to demonstrate the remote capability of the server. Both the database and application server will share the same Internet protocol (IP) address as it is a shared hosting service. An improvised request based authentication API has been developed to
simulate how the IoT devices authenticate against the web server and transmit data.

This API algorithm as shown in figure 3.3 as seen above, is the re-engineered encryption algorithm used to convert the plain text to cipher text which is then stored in the database as cipher text. While we understand that decrypting when the cipher text hits the database still opens the system to hack and loss of data integrity, having a cipher text within the database based on our layered encryption will serve as a more fortified system. When this data hits the database from the IoT devices, an automated mail is generated and sent to the doctor assigned to the specific patient to view data transmitted for review. Doctor then logs into this system and data is returned to the doctor in plain text. Not until the doctor is fully authenticated will the system reveal the plain text version of the data.

As the study implies, IoT authentication, we took it a little further as to apply the encryption to not just authentication from both ends for user data integrity, but also patient data integrity.
CHAPTER 4 SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system. In this project, a simulation of the IOT device using form controls, database were implemented.

4.1. Hardware Requirements

The IoT body area network system was developed using reactjs- a javascript framework on a Microsoft Windows 10 PC, with 4GB of RAM and 465GB Hard Disk Drive, with an Intel® core (TM) 2 Duo CPU running at 2.00 GHz.

The minimum hardware requirements for efficiently running this software are:

1. Microsoft Windows Vista/Win 7 and later
2. At least 320 GB (Hard Disk Drive).
3. At least a processor with speed above 1.7 GHz.
4. At least 2 GB RAM.

4.2. Implementation/Installation Procedure

The internet of things in body area network pages were developed locally and was served through the localhost of WAMP server before being deployed to an online space for production use.

WAMP (Windows,Apache,MySQL,PHP) sever was chosen specifically because of its robustness, easy compatibility with various database types and storage engines and also it is open source.

WAMP for windows provides an easy to install Apache-MySQL-PHP framework. It
saves time and effort and provides support for web development frameworks like Drupal, CodeIgniter, cakePHP, or Laravel on any Windows PC. Below is a detailed installation procedure.

1. In your web browser, go to [http://www.wampserver.com/en/#download-wraper](http://www.wampserver.com/en/#download-wraper) (Ensure you are connected to the internet)
2. Click on the download link for WAMP.
3. When prompted for the download, click “Save” and wait for your download to finish.
4. Once your download is complete, install the program, click on “Run”.
5. Accept default settings.
6. When the installation is complete, exit the command window by typing x on the command line.
7. Start the WAMP server.

If the installation went successfully, the wamp icon will turn green and ready for use. Open a web browser and enter “localhost” on your address bar. You will be redirected to the homepage of the server. The source file for the IoT body area network is placed in the WAMP document root folder (www). The various files would then be served through requests from the web browsers to the web server (localhost).

### 4.3. System Documentation

The IoT in body area network Software after development has carefully undergone various stages of checks and quality assurance verification. Below, we have a list of all working pages of IoT in body area network software. Each page design follows a
logical order.

Fig 4.1 Home page

Log in here

Email

enter email address

Password

password

Login
Fig 4.2 Login interface

Using the application requires having an account on the app. The above picture shows how the login form is rendered to the screen. This application keeps record and should not be open to anyhow person to access, every user trying to access the system must pass through authentication process (using email and password). A wrong login credential would be thrown aside, while valid details would take the user to the dashboard, hence, starts the session.

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Age</th>
<th>Occupation</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Joseph Wood</td>
<td>32</td>
<td>Artist</td>
<td>View History</td>
</tr>
<tr>
<td></td>
<td>Antonie Farrah</td>
<td>47</td>
<td>Lecturer</td>
<td>View History</td>
</tr>
<tr>
<td></td>
<td>Gabriella Moise</td>
<td>19</td>
<td>Commercial Driver</td>
<td>View History</td>
</tr>
</tbody>
</table>
Here, all registered patient under the logged in doctor is shown with a link to view their medical record.

Fig 4.4 Create patient account

Patient Registration Page: This page allows new patient to be added. The system is a password protected system, so, adding patient is paramount to the application life cycle.
### Fig 4.5. Patient history

The above diagram shows the medical history of a particular patient. Before this information was displayed, the system already decrypted the non-readable data in the database using the doctor’s unique secret key with OpenSSL.

### 4.4. System Testing

System testing is testing conducted on a complete, integrated system to evaluate the system compliance with its specified requirements [35]

Testing is the next phrase after the implementation stage although some units of testing were carried during implementation. Every stage of the implementation was inspected and tested throughout the development process from the user’s requirement.
definition until the final stage of implementation. Testing is a very essential and compulsory part of the Software Development process as any defects and errors are identified here then corrected.

In order to ensure that the system developed is functional, secure and error-free, different tests were carried out as one test may not be enough to judge the functionality of the automated livestock information management. The various pages of the software were checked for consistency and to ensure no broken links existed.

### 4.5. System Maintenance

At the maintenance phase of the authentication system, the system was carefully assessed to ensure it doesn’t become obsolete. The initial design is subjected to change. Normally, this phase of software development involves continuous evaluation of the system in terms of performance. Due to the nature of the software design and the software programming style used for creating the software, it would be relatively easy to add more functionality or modify any modules as required. Also, the evaluation phase of the Software Development Life Cycle is closely knit with the system maintenance phrase.

Basic maintenance on the administrator’s part would be to regularly update the WAMP server and PHP versions, which would be gotten online since it is an Open Source Software. The user, on the other hand, would have to ensure the regular update of his/her browser in other to keep up with the pace at which browser technologies are being improved upon. This would ensure smooth interoperability between the client machines.
CHAPTER 5 SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1. Summary

This project was carried out in two layers. The first layer was the design of the Web interface. The second was the development of the Web which entails the writing of codes that specified the functionalities of every item found on the website. The implementation was carefully done such that the aim and objectives of the system were met. The site was developed using HTML, PHP and JavaScript with MySQL as the backend.

5.2. Conclusion

Adoption of open source standard cryptographic products provide the opportunity to build proprietary security mechanisms that enhance security measures based on certain data analysis and parameters that help provide insights to what is required for extra security layer within the algorithm. As seen in the previous chapter, we have been able to demonstrate how creating another security layer based on certain vernacular can also improve data integrity. Users can be rest assured that their data is safe while patients can go to sleep knowing their medical data will not be tampered with. However more attention should be paid to enhancing this layer and more innovations can be taken up from here.

5.3. Recommendations and Further Study

The paradigm shift and current trend in breakthrough of Information and Technology
field requires that each individuals or organizations be up to date with the developments as they unfold.

Open cryptographic standards are very important to adopt by every security conscious personnel that seeks to provide top notch information security. As seen in this study, OpenSSL has served a purpose in its capacity as an open source cryptographic product open to tweaking and reverse engineering as the implementation requirements directs. However, there are a few recommendations for further study in this regard if IoT authentication in a body area network has to be properly guarded.

A relevant further study would be in virtual private network services. As seen in this implementation, we have hosted the database server over a public network, this means that our authentication process fulfils data integrity, but it is only a matter of time before creativity and evolution catches up with this concept, using a virtual private network on the other hand will prove more secure over the internet from attacks
REFERENCES

[9] Graphical Password Authentication by LIPSA PANIGRAHI
USA Pages 17 – 22.


[21] Application programming interface

[22] Internet of Things (IoT) Security: Current Status, Challenges and Prospective Measures Rwan Mahmoud, Tasneem Yousuf, Fadi Aloul, Imran Zualkernan Department of Computer Science & Engineering American University of Sharjah, UAE

[23] A Survey of Internet of Things (IoT) Authentication Schemes † Mohammed El-hajj 1,2,*, Ahmad Fadlallah 1, Maroun Chamoun 2 and Ahmed Serhrouchni 3,*


[25] Security in Internet of Things: Issues, Challenges and Solutions Hanan Aldowah1(&), Shafiq Ul Rehman2, and Irfan Umar1 1 Universiti Sains Malaysia (USM), Penang, Malaysia hanan_alwoah@yahoo.com, irfan@usm.my 2 Singapore University of Technology and Design (SUTD), Tampines, Singapore shafiq_rehman@sutd.edu.sg 2018


[29] K. Van Dam, S. Pitchers, and M. Barnard, “Body area networks: Towards a wearable future,” in Proceedings of WWRF kick off meeting, Munich, Germany, 6-7 March 2001


import axios from 'axios';
const api = "http://localhost/iot";

/*==========LOGIN=============================*/
export function loginUser(data){
    const request =
        axios.post(api+'/api/reducer.php', {
            action: data.action,
            email: data.email,
            password: data.password
        })
        .then(response =>
            response.data)
    return{
        type:'USER_LOGIN',
        payload: request
    }
}

export function auth(){
    let request = {
        "isAuth": false
    }
    if(localStorage.getItem('email') !== null){
        request = {
            'id':localStorage.getItem('id'),
            'name':localStorage.getItem('name'),
            'email':localStorage.getItem('email'),
            'discipline':localStorage.getItem('discipline')
        }
    }
}

export function {id}:localStorage.getItem('id'),
    'name':localStorage.getItem('name'),
    'email':localStorage.getItem('email'),
    'discipline':localStorage.getItem('discipline')
}
export function getAllPatient(id, action) {
    const request = axios.get(api + '/api/reducer.php?id=' + id + '&action=' + action);
    return {
        type: 'GET_ALL_PATIENT',
        payload: request
    }
}

export function createPatient(data) {
    const request = axios.post(api + '/api/reducer.php', {
        action: data.action,
        name: data.name,
        age: data.age,
        occupation: data.occupation,
        doctor: localStorage.getItem('id'),
        location: data.location
    }).then(response => response.data);
    return {
        type: 'ADD_PATIENT',
        payload: request
    }
}

export function getPatientHistory(id, action) {
    const email = localStorage.getItem('email')
    const request = axios.get(api + '/api/reducer.php?id=' + id + '&email=' + email + '&action=' + action);
    return {
        type: 'GET_PATIENT_HISTORY',
        payload: request
    }
}
then(response =>
  response.data); return {
    type: 'GET_PATIENT_HISTORY',
    payload: request
  }
}
export function clear_patient() {
  return {
    type: 'CLEAR_PATIENT',
    payload: {}
  }
}

Patient_reducer.js

export default function(state={}, action){
  switch(action.type){
    case "ADD_PATIENT":
      return {...state, register: action.payload}
    case "GET_ALL_PATIENT":
      return {...state, list: action.payload} case
    "GET_PATIENT_HIST ORY":
      return {
        ...state,
        view:
        action.payload.history,
        details:
        action.payload.details
      ,
      index:
      action.payload.doctor
    }
    case "CLEAR_PATIENT":
  }
return {
  ...state,
  view:
    action.pay
    load,
  details:
    action.pay
    load,
  doctor:
    action.pay
    load
}

default:
  return state;
}

User_reducer.js

export default
  function(state={},
    action){
    switch(action.type){
      case "USER_LOGIN":
        return {...state, login:
          action.payload} case
      "USER_AUTH":
        return {...state, login: action.payload}
      default:
        return state;
    }
  }

Patient_list.js

import React from 'react';
import { Link } from 'react-router-dom';

const PatientList
  = (item) => {

return (

<tr key={item.id}>
  <th scope="row"></th>
  <td>{item.name}</td>
  <td>{item.age}</td>
  <td>{item.occupation}</td>
  <td><Link to="/patient/${item.id}"> View History </Link></td>
</tr>
)

export default

PatientList;

Login.js

import React, { Component } from 'react'; import { connect } from 'react-redux'; import { loginUser } from '../../actions'; import { Alert, Button, Form, FormGroup, Label, Input, Container, Row, Col } from 'reactstrap'; class Login extends Component {
  onDismiss() => {
    this.setState({
      visible: false
    });
  }
  handleInput = (e) => {
    this.setState({
      [e.target.id]: e.target.value
    });
  }
  submitForm = (e) => {
    e.preventDefault();
    this.props.dispatch(loginUser(this.state));
  }
}
UNSAFE_componentWillReceiveProps =
(nextProps) => {
  if(nextProps.user.login.isAuth){
    let us = nextProps.user.login.user;
    localStorage.setItem('id',us.id)
    localStorage.setItem('name',us.name)
    localStorage.setItem('email',us.email)
    localStorage.setItem('discipline',us.discipline)
  }
}
render() {
  let user = this.props.user;
  return (
    <Container>
      <Row>
        <Col sm="12" md={{ size: 6, offset: 3 }}>
          <Form
            onSubmit={this.SubmitForm}>
            <h2>Log in here</h2>
            <FormGroup>
              <Label for="email">Email</Label>
              <Input type="email" name="email" id="email" value={this.state.email} onChange={this.handleInput} placeholder="enter email address" />
            </FormGroup>
            <FormGroup>
              <Label for="examplePassword">Password</Label>
              <Input type="password" name="password" id="password" value={this.state.password} onChange={this.handleInput} placeholder="password" />
            </FormGroup>
            <Button>Login</Button>
            <Alert color="info" isOpen={this.state.visible} toggle={this.onDismiss}>
              {user.login.msg}
            </Alert>
          </Form>
        </Col>
      </Row>
    </Container>
  )
}
function mapStateToProps(state) {
  return {
    user: state.user
  }
}

export default connect(mapStateToProps)(Login);

Logout.js

import React from 'react'

const Logout = (props) => {
  localStorage.removeItem('id')
  localStorage.removeItem('name')
  localStorage.removeItem('email')
  localStorage.removeItem('discipline')

  setTimeout(() => {
    props.history.push('/')
  }, 1000)

  return (
    <div>
      <h1>
        Hope to see you again
      </h1>
    </div>
  )
}
import React, { Component } from 'react'; import { connect } from 'react-redux'; import { getAllPatient } from '../actions'; import { Table } from 'reactstrap'; import PatientList from './widgetsUI/patient_list';

class PatientContainer extends Component {

    UNSAFE_componentWillMount() {
        this.props.dispatch(getAllPatient(localStorage.getItem('id'), "getAllPatient"))
    }

    renderItems = (patients) => {
        patients.list ?
        patients.list.map(item => {
            return (<PatientList {...item} key={item.id} />)
        })
    :null

    }

    }

export default PatientContainer;
<div>
  <Table responsive>
<thead>
<tr>
  <th>#</th>
  <th>Name</th>
  <th>Age</th>
  <th>Occupation</th>
  <th>Action</th>
</tr>
</thead>
<tbody>
  {this.renderItems(this.props.patients)}
</tbody>
</Table>
</div>

function mapStateToProps(
  state)
  return {
    patients: state.patients
  }
}
export default connect(mapStateToProps)(PatientContainer);

Patient_history.js

import React, { Component } from 'react';
import { connect } from 'react-redux';
import { getPatientHistory, clear_patient } from '../actions';
import { Table, Row, Col, Card, CardTitle, CardText } from 'reactstrap';
import shortid from 'shortid';

class PatientHistory extends Component {

  UNSAFE_componentWillMount()

  Mount()

    this.props.dispatch(getPatientHistory(this.props.match.params.id,
"getPatientHistory")
}

UNSAFE_componentWillUnmount(){
    this.props.dispatch(clear_patient());
}

renderItems =
( patients ) => {
    patients.view ?
    patients.view.map( item =>{

        return ( 
            <tr key={item.id}> 
                <th scope="row">{item.id}</th>
                <td>{item.illness}</td>
                <td>{item.heart_pulse_rate}</td>
                <td>{item.date_time}</td>
                {/* <td>normal</td> */}
            </tr>
        )
    }
    )
    :
    ( 
        <tr key={shortid.generate()}>
            <th colSpan="5">
                No record found!!
            </th>
        </tr>
    )
}

renderDoctor =
( patients ) => {
    patients.doctor ?

    ( 
        <CardText>
            <span style={{ display: 'block' }}> 
                Fullname: { patients.doctor && patients.doctor['name'] } 
            </span>
        </CardText>
    )
}
Discipline: { patients.doctor && patients.doctor['discipline'] }

"No record found"

render()
{ 
  let patients = this.props.patients
  return (
    <div>
      <Row style={{marginTop: '7px'}}>
        <Col sm="6">
          <Card body color="primary">
            <CardTitle style={{ color: '#333', borderColor: '#333', fontWeight: 'bolder' }}>Patient Details</CardTitle>
            <CardText>
            <span style={{ display: 'block'}}>
              Fullname: { patients.details && patients.details['name'] }<span></span>
            </span>
            <span style={{ display: 'block'}}>
              Occupation: { patients.details && patients.details['occupation'] }<span></span>
            </span>
          </CardText>
        </Col>
        <Col sm="6">
          <Card body inverse color="success">
            <CardTitle style={{ color: '#333', borderColor: '#333', fontWeight: 'bolder' }}>Doctor's Profile</CardTitle>
            <CardText>{this.renderDoctor(this.props.patients)}</CardText>
        </Col>
      </Row>
  </div>

  <Col sm="12" style={{marginTop: '7px'}}>
    <Table responsive>
      <thead>
<tr>
  <th>#</th>
  <th>illness</th>
  <th>Pulse (bpm)</th>
  <th>Date</th>
  {/* <th>Normal?</th> */}
</tr>

<thead>
  <tbody>
    {this.renderItems(patients)}
  </tbody>
</Table>

</Col>

</Row>
</div>

function
mapStateToProps(
  state){ return {
    patients: state.patients
  }
}
export default
connect(mapStateToProps)(PatientHistory);

Auth.js
import React, { Component }
from 'react'; import { connect }
from 'react-redux'; import { auth } from './actions';

export default
function(ComposedClass, reload)
{
  class AuthCheck extends
    Component {
    UNSAFE_component
      WillMount()
    { }
this.props.dispatch(auth())

UNSAFE_componentWillReceiveProps = (nextProps) => { this.setState({loading: false})
  // console.log(nextProps)

  if(!nextProps.user.login.isAuth) {
    if(reload){
      this.props.history.push('/login')
    }
  } else {
    if(reload !== false){
      this.props.history.push('/user')
    }
  }

}

render() {
  if(this.state.loading){
    return <div> loading... </div>
  }

  return (
    <ComposedClass {...this.props} user={this.props.user} />
  )
}

function mapStateToProps(state) { return{
  user: state.user
}
}

return connect(mapStateToProps)(AuthCheck)
APPENDIX B

Store.php

<?php

require_once "db.php";

class Store
extends Database
{
    public function construct()
    {
        parent::__construct();
    }

    public static function existOne($tbl, $col, $value)
    {
        $conn = Database::getInstance();
        $select = $conn->db->prepare("SELECT * FROM $tbl WHERE $col LIKE ? Limit 1");
        $select->execute(array($value)); return $select->rowCount();
    }

    public static function existTwo($tbl, $col, $col2, $value, $value2)
    {
        $conn = Database::getInstance();
        $select = $conn->db->prepare("SELECT * FROM $tbl WHERE $col LIKE ? AND $col2 LIKE ? Limit 1");
        $select->execute(array($value, $value2)); return $select->rowCount();
    }

    public static function getAllPatient($id)
    {
        $conn = Database::getInstance();
        $select = $conn->db->prepare("SELECT * FROM patient WHERE doctor = ?");
        $select->execute(array($id)); return $select->fetchAll();
    }

    public static function getPatientHistory($id)
    {
        $conn = Database::getInstance();
        $select = $conn->db->prepare("SELECT * FROM history WHERE patient = ?");
        $select->execute(array($id)); return $select->fetchAll(PDO::FETCH_OBJ);
    }

    public static function getPatientDetails($id)
    {
        $conn = Database::getInstance();
        $select = $conn->db->prepare("SELECT * FROM patient WHERE id = ?");
        $select->execute(array($id)); return $select->fetchAll(PDO::FETCH_OBJ);
    }
}
public static function CreatedOn() {
    return date('Y-m-d H:i:s');
}

private static function createPatient($data) {
    $conn = Database::getInstance();
    $name = $data['name'];
    $age = (int)$data['age'];
    $location = $data['location'];
    $occupation = $data['occupation'];
    $doctor = (int)$data['doctor'];

    $payload = array();
    $stmt = $conn->db->prepare("INSERT INTO patient (name, age, occupation, location, doctor) VALUES (:name, :age, :occupation, :location, :doctor )");
    $stmt->bindParam(':name', $name, PDO::PARAM_STR);
    $stmt->bindParam(':age', $age, PDO::PARAM_INT);
    $stmt->bindParam(':occupation', $occupation, PDO::PARAM_STR);
    $stmt->bindParam(':location', $location, PDO::PARAM_STR);
    $stmt->bindParam(':doctor', $doctor, PDO::PARAM_INT); if ($stmt->execute()):
        $payload['success'] = true;
        $payload['msg'] = 'Patient bio-data created successfully';
    else:
        $payload['success'] = false;
        $payload['msg'] = 'Patient was not created';
    endif;
    return $payload;
}

public static function savePatientHistory($patient, $pulse, $illness, $date_time) {
    $conn = Database::getInstance();
    // $date_time = Store::CreatedOn();
    $stmt = $conn->db->prepare("INSERT INTO history (patient, illness, date_time, heart_pulse_rate) VALUES (:patient, :illness, :date_time, :heart_pulse_rate)");
    $stmt->bindParam(':patient', $patient, PDO::PARAM_STR);
    $stmt->bindParam(':illness', $illness, PDO::PARAM_STR);
    $stmt->bindParam(':date_time', $date_time, PDO::PARAM_STR);
    $stmt->bindParam(':heart_pulse_rate', $pulse, PDO::PARAM_INT); if ($stmt->execute()):
        $payload['success'] = true;
        $payload['msg'] = 'Patient bio-data created successfully';
    else:
        $payload['success'] = false;
        $payload['msg'] = 'Patient was not created';
    endif;
    return $payload;
}
$stmt->bindParam(':patient', $patient, PDO::PARAM_INT);
$stmt->bindParam(':illness', $illness, PDO::PARAM_STR);
$stmt->bindParam(':heart_pulse_rate', $pulse, PDO::PARAM_STR);
$stmt->bindParam(':date_time', $date_time, PDO::PARAM_STR); if ($stmt->execute()):
    return 1;
else:
    return 0;
endif;

public static function encrypt_iot($data,$separator){
    $cipher_method = 'aes-128-ctr';
    $enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
    $enc_iv = openssl_random_pseudo_bytes(openssl_cipher_iv_length($cipher_method));
    $crypted_token = openssl_encrypt($data, $cipher_method, $enc_key, 0, $enc_iv) . $separator . bin2hex($enc_iv);
    unset($data, $cipher_method, $enc_key, $enc_iv); return $crypted_token;
}

public static function decrypt_iot($encrypted_data,$separator){
    list($encrypted_data, $enc_iv) = explode($separator, $encrypted_data);
    $cipher_method = 'aes-128-ctr';
$enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
$data = openssl_decrypt($encrypted_data, $cipher_method, $enc_key, 0, hex2bin($enc_iv)); unset($encrypted_data, $cipher_method, $enc_key, $enc_iv);
return $data;

Index.php

<html>
<head>
    <title> Admin </title>
</head>
<body>
<style>
    .container {
        display: block;
        position: absolute;
        margin-top: 20%;
        left: 25%;
        width: 50%;
    }
    form {
        display: flex;
        flex-direction: column
    }
    input {

    }
</style>
<div class="container">
    <form method="post">
        <input type="text" name="pulse" placeholder="Enter pulse rate" />
        <input type="text" name="illness" placeholder="Enter illness" />
        <input type="submit" value="Send" />
    </form>
</div>

77
<?php
require_once "classes/db.php";
require_once "classes/store.php";
$conn = Database::getInstance();
date_default_timezone_set("Etc/GMT-8");
if(isset($_POST['pulse'])){  
    $date_time = Store::CreatedOn();
    $pulse = Store::encrypt_iot($_POST['pulse'], "JQ==mololuwab6834bbdf048c5869dbb835a0f0356fb");
    $illness = Store::encrypt_iot($_POST['illness'], "JQ==mololuwab6834bbdf048c5869dbb835a0f0356fb");
    $patient = 3;
    $doctor = "boladebode@gmail.com";
    $send = Store::savePatientHistory($patient, $pulse, $illness, $date_time);
    if($send === 1):
        echo "data saved successfully";
    else:
        echo "data error!! Please try again later";
    endif;
}
?>

<?php
// Please specify your Mail Server - Example:
maint.example.com.
ini_set("SMTP","smtp.gmail.com");

// Please specify an SMTP Number 25 and 8889 are valid SMTP Ports. ini_set("smtp_port","25");

// Please specify the return address to use ini_set('sendmail_from', 'to.odeyemi@gmail.com');
// if(isset($_POST['email'])) {
   // $i = 2;
if($i >=2) {

    // EDIT THE 2 LINES BELOW AS REQUIRED
    // $email_to = "waltsteph82@gmail.com";
    $email_to = "boladebode@gmail.com";
    $email_subject = "ENQUIRY";

    function died($error) {
        // your error code can go here
        echo "We are very sorry, but there were error(s) found with the form you submitted.";
        echo "These errors appear below.<br />";
        echo $error."<br />";
        echo "Please go back and fix these errors.<br />";
        die();
    }

    $name ="Akinniyi Boluwatife"; // required
    $telephone = "07037351836"; // required
    $email_from = "boladebode@yahoo.com"; // required
    $find = "IOT info should be here"; // required
    // $comment = $_POST['comment']; // required
    ##################################################
    #
    #
    #
    $error_message = "";
    $email_exp = '/^[A-Za-z0-9_.%]+@[A-Za-z0-9-]+\.[A-Za-z]{2,4}$/';
    if(!preg_match($email_exp,$email_from)) {
        $error_message .= 'The Email Address you entered does not appear to be valid.<br />
    }

    $string_exp = "/[A-Za-z .-]+$/";
    if(!preg_match($string_exp,$name)) {
        $error_message .= 'The First Name you entered does not appear to be valid.<br />
    }
    if(strlen($error_message) > 0) {
        died($error_message);
    }
    $email_message = "Form details below.\n\n";

}
function clean_string($string) {
    $bad = array("content-type","bcc:","to:","cc:","href");
    return str_replace($bad, "", $string);
}

$email_message .= "Full Name: ".clean_string($name)."n";
$email_message .= "Email: ".clean_string($email_from)."n";
$email_message .= "Telephone: ".clean_string($telephone)."n";
$email_message .= "How did you find us: ".clean_string($find)."n";

// $email_message .= "Comment: ".clean_string($comment)."n"

// create email headers
$headers = 'From: '.email_from."r\n".
'Reply-To:
'email_from."r\n".
'X-Mailer: PHP/'.phpversion();
$success = mail($email_to, $email_subject, $email_message, $headers);
if(isset($success) && $success) {
    ?>
-- include your own success html here --> success
<?php
} else{
}

?>

echo "There was a problem sending your mail";

Open.php

<!--
$token = "The quick brown fox jumps over the lazy dog.";
$separator = "::";
-->
$cipher_method = 'aes-128-ctr';
$cipher_method = 'aes-128-ctr';
$enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
$enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
$enc_iv = openssl_random_pseudo_bytes(openssl_cipher_iv_length($cipher_method));
$enc_iv = openssl_random_pseudo_bytes(openssl_cipher_iv_length($cipher_method));
echo $crypted_token = openssl_encrypt($token, $cipher_method, $enc_key, 0, $enc_iv) .
echo $crypted_token = openssl_encrypt($crypted_token, $cipher_method, $enc_key, 0, $enc_iv) .
$separator . bin2hex($enc_iv);
unset($token, $cipher_method, $enc_key, $enc_iv); echo "<p></p>";

<?PHP
list($crypted_token, $enc_iv) = explode("::", $crypted_token);
list($crypted_token, $enc_iv) = explode("::", $crypted_token);
$cipher_method = 'aes-128-ctr';
cipher_method = 'aes-128-ctr';
$enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
$enc_key = openssl_digest(php_uname(), 'SHA256', TRUE);
echo $token = openssl_decrypt($crypted_token, $cipher_method, $enc_key, 0, hex2bin($enc_iv));
echo $token = openssl_decrypt($crypted_token, $cipher_method, $enc_key, 0, hex2bin($enc_iv));
unset($crypted_token, $cipher_method, $enc_key, $enc_iv);
?></p>

Reducer.php

<?php
    session_start();
    require_once "classes/db.php";
    require_once "classes/store.php";
    $conn = Database::getInstance();
    date_default_timezone_set("Etc/GMT-8");
    $config = 'http://localhost:3000';
    $config = 'http://localhost:3000';
    header('Access-Control-Allow-Origin: '. $config);
    header('Access-Control-Allow-Origin: '. $config);
    header('Access-Control-Allow-Headers: X-API-KEY, Origin, X-Requested-With, Content-Type, Accept, Access-Control-Request-Method');
    header('Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS');
    header("Content-type: application/json");
    header("Content-type: application/json");
    $data = json_decode(file_get_contents("php://input"), TRUE);
    $json = file_get_contents('schema.json');
    if(isset($data['action'])){
        // started here
        if($data['action'] == "login"){

81
$payload = array();
$obj = json_decode($json);
$doctor = array();
foreach($obj as $key => $val){ array_push($doctor, $key); }
if(in_array($data['email'], $doctor)){
    foreach($obj as $key => $val){
        if($key === ($data['email'])){
            if($val !== $password === $data['password']):
                $payload['isAuth'] = true;
                $payload['msg'] = 'Login successful';
            $payload['user'] = $val;
        else:
            $payload['isAuth'] = false;
            $payload['msg'] = 'Wrong login credentials';
        endif;
        }
    }
} else{
    $payload['isAuth'] = false;
    $payload['msg'] = 'Wrong login credentials';
}
    echo json_encode($payload);
}
if($data['action'] === "createPatient"){ $send = Store::createPatient($data); echo json_encode($send);
}
if(isset($_GET['action'])){ if($_GET['action'] == "getAllPatient"){
} }
$payload = array();
    $send = Store::getAllPatient($_GET['id']);
    
    echo json_encode($send);
}

if($_GET['action'] == "getPatientHistory"){
    $payload = array();
    $doctor = [];
    $data = [];
    $history = Store::getPatientHistory($_GET['id']);
    
    $token = ";
    foreach($history as $item):
        
        $obj = json_decode($json);
        $did = array();
        foreach($obj as $key => $val){ array_push($did, $key); }
        if(in_array($_GET['email'], $did)){
            foreach($obj as $key => $val){ if($key === ($_GET['email'])){ //array_push($doctor, $val); $token = $val-$key; }
                }
        }
        
        $row = array(  'name'=> $val->name,  'email'=> $val->email,  'discipline'=> $val->discipline,
                    );
        $doctor[] = $row;
    }

    $payload["history"] = $data;
    $payload["doctor"] = $doctor;
$payload['details'] = Store::getPatientDetails($_GET['id']);

    echo json_encode($payload);

};

//echo Store::encrypt_iot("hypertension", "JQ==mololuwab6834bbdf048c5869dbb835a0f0356fb");
//echo "<p></p>";
//echo Store::decrypt_iot("JQ==mololuwab6834bbdf048c5869dbb835a0f0356fb", "mololuwa");

?>

Schema.json

{

  "boladebode@gmail.com": {
    "id": 1,
    "name": "Akinniyi Bolade",
    "email": "boladebode@gmail.com",
    "password": "password",
    "discipline": "Heart Surgeon",
    "key": "JQ==mololuwab6834bbdf048c5869dbb835a0f0356fb"
  },

  "Kavanaugh@gmail.com": {

    "id": 2,
    "name": "Kavanaugh",
    "email": "Kavanaugh@gmail.com",
    "password": "password",
    "discipline": "Brain Surgeon",
    "key": ""
  }

}
CURRICULUM VITAE

MOLOLUWA JOSIAH

Education:
January 2018 – Present
Masters of Computer Science Current GPA: 3.70
University of New Brunswick, Fredericton, New Brunswick

September 2013 – June 2016
Bachelor of Computer Science (ICT Option), First Class Honors
Ajayi Crowther University, Oyo, Oyo State, Nigeria

Skills:
Programming Skills: Structured Query Language (SQL), Hyper-Text Markup Language (HTML), Python, No SQL
Effective use of Microsoft office (Word, Excel, PowerPoint)
Experience with Regular expressions Experience with GIT
Effective use of Jenkins
Knowledgeable in Agile Software development
Methodology Experience in Network Administration and IT Support

Publications: None

Conference Presentations: None