University of saad dahleb - Blida 1
Department of IT

Faculty of Sciences
Computer science field

MASTER THESIS

Role-based access control for an intranet medical practice management system with optional P2P network internet access

Authors:
Brahim Aymen
Daouadji Aymen

Promotor:
Mrs.Meskaldji khouloud

in front of members of the jury

Mme Arkam Meriem(Presedent)
Mr Khediri (examinator)

A thesis submitted in fulfillment of the requirements for the degree of Master

in

SIR
Department of IT

February 12, 2023

“*An idiot admires complexity, while a genius admires simplicity.*”

Terry Davis

Abstract

Since healthcare is a sensitive field in terms of health information privacy and security, a number of security and engineering solutions have been proposed to ensure the security of medical information in a health information system.
This project is a proposal for a standardized practice management system for healthcare facilities and the development of practice management software based on that system to facilitate the exchange, protection, and management of health information. The system provides a solution for the protection and security of patient information while enabling data exchange and medical staff collaboration. By implementing a restricted communications network with a role-based access control model, it reduces the risk of unauthorized data access and establishes a standard way for healthcare facilities to exchange information. The system is designed to run on multiple devices on a local network by adopting a client-server architecture, which will result in the development of two software applications, one for the server and the other one for the clients.

Keywords: Access control, Role based access control, Intranet, Healthcare information system, Medical practice management system, Data protection

Resumé

La santé est un domaine sensible en termes de confidentialité et de sécurité des informations de la santé. Un certain nombre de solutions ont été proposées pour assurer la protection des dossiers médicaux dans le système d'information de santé. Ce projet est une proposition de système de gestion de cabinet standardisé pour les établissements de santé sous forme d'un programme de gestion de la pratique basé sur ce système pour faciliter l'échange, la protection et la gestion des informations de santé en meme temps.
Le système fournit une solution pour protéger les informations des patients tout en permettant l'échange de données et la collaboration du personnel médical en mettant en œuvre un réseau de communication contraint par le modèle de contrôle d'accès basé sur les rôles, ce qui réduit le risque d'accès non autorisé aux données et définit un moyen standard de partager des informations pour les centres de santé.
Le système est conçu pour fonctionner sur plusieurs machines sur un réseau local en adoptant une architecture client-serveur, ce qui se traduira par le développement de deux applications logicielles, une pour le serveur et une pour les clients.

Mots clés: Contrôle d'accès, Contrôle d'accès basé sur les rôles, Intranet, Système d'information de santé, Système de gestion de cabinet médical, Protection des données.

ملخص

نظرًا لأن الرعاية الصحية مجال حساس من حيث خصوصية وأمن المعلومات الصحية ، فقد تم اقتراح عدد من الحلول الأمنية والهندسية لضمان أمن المعلومات الطبية في نظام المعلومات الصحية. هذا المشروع عبارة عن اقتراح لنظام إدارة ممارسات معياري لمرافق الرعاية الصحية وتطوير برنامج إدارة الممارسة على أساس هذا النظام لتسهيل تبادل المعلومات الصحية وحمايتها وإدارتها. يوفر النظام حلاً لحماية معلومات المرضى مع تمكين تبادل البيانات وتعاون الطاقم الطبي. من خلال تنفيذ شبكة اتصالات مقيدة بنموذج التحكم في الوصول المستند إلى الدور ، فإنه يقلل من مخاطر الوصول غير المصرح به إلى البيانات ويضع طريقة قياسية لمنشآت الرعاية الصحية لتبادل المعلومات. تم تصميم النظام ليعمل على أجهزة متعددة على شبكة محلية من خلال اعتماد بنية خادمالعميل ، مما سينتج عنه تطوير تطبيقين برمجيين ، أحدهما للخادم والآخر للعملاء.

الكلمات المفتاحية: صلاحية التحكم ، التحكم في الوصول المستند إلى الدور، الشبكة الداخلية، نظام معلومات الرعاية الصحية، نظام إدارة الممارسات الطبية، حماية البيانات.

Acknowledgements

In the Name of Allah, the Most Merciful, the Most Compassionate, Alhamdulillah all praises belongs to Almighty Allah, the Lord of the worlds and prayers and peace be upon Muhammad His servant and messenger.
First and foremost, We must acknowledge our limitless thanks to Allah, the Ever-magnificent, the Ever-Thankful, for His help and bless by giving us the courage and power to complete this thesis.
We are grateful to Dr Boumediene Merouane for his insightful ideas, comments and suggestions that lightened our way to the completion of this project.
We would also like to express our gratitude and appreciation for Mrs.Meskaldji khouloud. whose guidance, support and encouragement has been invaluable throughout this study.
Last, but not least, our warm and heartfelt thanks go to our family for their tremendous support and hope they had given to us. Without that hope, this thesis would not have been possible. Thank you all for the strength you gave us.

Contents

List of Figures

List of Tables

Abbreviations

ACL Access Control List. 17

AES Advanced Encryption Standard. 32, 40, 41, 43

API Application Programming Interface. 50, 51, 62, 64, 72–75

APIPA Automatic Private IP Addressing. 85

C-TMAC Context and Team-based Access Control. vii, 25–27

CDS Clinical Decision Support. 11

DES Data encryption standard. 32

DH Diffie-Hellman protocol. 43

DHCP Dynamic Host Configuration Protocol. 85

DLP Data loss prevention. vii, 32

E2EE End-to-End Encryption. vii, ix, 41, 42, 52

EMR Electronic Medical Record. 11, 47

ERD Entity Relationship Diagram. viii, 57

FBE File-based encryption. 43, 44

FDE Full disk encryption. 43

GUI Graphical User Interface. 50, 51

HHS Health and Human Services. 36

HIPAA Health Insurance Portability and Accountability Act. vii, 35, 36, 43

HIS Health Information System. vi, ix, xi, 2, 10, 14, 29, 30, 47, 50

HTML Hypertext Markup Language. 63, 65, 71

HTTP Hypertext Transfer Protocol. 62

I-RBAC Intranet role-based access control. vii, 25–27

ICMP Internet Control Message Protocol. 7

ICT	Information Communication Technology. 14
JWT	JSON Web Tokens. 73, 74
LAN	Local Area Network. 3, 4, 8, 11, 14, 52
MAN	Metropolis Area Network. 3, 4
NIST	The National Institute of Standards and Technology. 25, 32
NTFS	New Technology File System. 17
ORM	Object-Relational Mapping. 73
OSI	Open Systems Interconnection. 6
P2P	Peer-To-Peer. 52
PHI	Personal Health Information. 31, 35, 36
PII	Personally Identifiable Information. 15, 31
PIN	Personal Information Number. 37
PMS	Practice Management Software. vii, 2, 11, 45, 49
RBAC	Role-based Access Control. vi, vii, ix, xi, 17–23, 25–28, 30, 46, 51, 58, 74, 78
RBAC/NIST	The National Institute of Standards and Technology role-based access control model. vii, 25, 27, 28
RPM	Remote Patient Monitoring. 11
RSA	Rivest Shamir Adleman algorithm. 34, 40, 41, 43
SCSS	Syntactically Awesome Style Sheet. 63
SSL	Secure Socket Layer. xi, 39, 40
TCP	Transmission Control Protocol. 7
TCP/IP	Transport Control Protocol/Internet Protocol. ix, 6–8, 11, 27
TLS	Transport Layer Security. vii, ix, xi, 32, 39–41
UDP	User Datagram Protocol. 7
UI	User Interface. 66, 67, 70, 71, 80
WAN	Wide Area Network. 3, 4
WebRTC	Web Real-Time Communication. 62

I dedicate this thesis to my beloved mother, who for months past, has encouraged me attentively with her fullest and truest attention to accomplish my work with truthful self-confidence, and to my dear brother who has always been there for me, and who has always been my source of inspiration, and to my sister that without her, I would not have been able to accomplish this work. At the end I would thank my whole family for their support and encouragement through all the years of my life.

Brahim Aymen

First, i would like to thank our Almighty Allah for giving us the strength, patience, and wisdom all throughout this journey. For without Him, i could not accomplish even a thing. To my parents, thank you so much for the understanding, wonderful care, and support you have always given to me. In return, i offer to you all my efforts and hardworks not just in this study but in all matters. I'd like to end by saying how much I appreciate my entire family's help and patience while I worked on the project.

Daouadji Aymen

Introduction

In the healthcare system, patient privacy is a responsibility that must be protected. Health centers have to keep track of patients' medical records and provide them only to the specific patient when it is required. With the rise of the Internet, healthcare systems have been trying to integrate the task of storing and managing records using the internet to be able to collaborate and exchange data. Data privacy and security are a concern when the data travels through a network, and that's what makes it hard for the healthcare field to integrate into a centralized network that provides data through the internet. Due to the fear of data leaking, which can cause patients to lose their privacy, most health centers refuse to rely on the internet to exchange medical data. Some even still use traditional sheet cheat documents. Using a local network to share and exchange data was a good option for health centers in terms of security, as it limited access to data from outsiders.

Problematic

Protecting and securing medical records on a local network is a responsibility for the health center. That leads us to the problem, which is how to manage and exchange data on a local network and enable collaboration between medical staff while keeping the patients' data safe.

Objective

The purpose of this project is to create a new healthcare standard for medical practice management software that covers all the needs of a healthcare facility, from enabling collaboration and data exchange between medical staff, to managing and protecting data from internal and external danger. A software that aspires to be a standard in healthcare systems, covering a large community or an entire region with all types of healthcare facilities, from small clinics to large hospitals. To achieve that, it requires very good knowledge of networking and security technologies, as-well as a strong study of health care system policies and requirements.

Thesis plan

The thesis has been divided into multiple chapters. Each chapter will focus on a different area of the study. Each section of a chapter offers information regarding an intriguing technology utilized in the development of the software.

Chapter 1 This chapter will focus on the impact of computer networks on healthcare organizations' knowledge management systems. Networking terms and many concepts will be briefly explained in this section. Then, it will discuss healthcare systems' use of knowledge management systems. The integration of networks into knowledge management systems will be demonstrated at the conclusion.

Chapter 2 In this chapter we will talk about access control, which is necessary to provide a sense of security and control over the system. We will briefly discuss various access control models and their advantages and disadvantages. Finally, we'll go over the role-based access control model, its various implementations, and how we implement it in our system.

Chapter 3 This chapter is about the privacy of data which is a big concern in Health Information Systems due to medical privacy policies. It will cover medical data privacy policies, and it will explain the methods to protect these data over the network and on storage.

Chapter 4 This chapter utilizes information from all preceding chapters to describe the conception of our PMS-based Health Information System (HIS). It then compares the software to other implementations and reveals the software's characteristics. The chapter will then cover the system's structure, architecture, and operation. Lastly, it will illustrate with diagrams the system's workflow.

Chapter 5 This chapter is all about implementation. It contains all the tools that go into making the software.

Chapter 1

Networks and knowledge management

The basic idea of communications consist of a sender, a receiver, and a message that need to be transmitted through a medium, both the sender and receiver must agree on a set of communication rules or protocols to achieve a meaningfully communication. Humans have been building tools and developing technologies to help improve the way we communicate.
To extend the interpersonal communication, it was also necessary to find a way to allow communication between computers. Since the 1970s Computer communication become an essential infrastructure for new era. Nowadays, networking [1] is used everywhere. From education to government offices to health system, every business and personal aspect is using what now called computer network. Computer network helped in improving organizations knowledge and learning by offering a new way to exchange and share knowledge.

1.1 Brief on computer networking

A computer network is a set of computers that are distributed geographically but connected in a way that allows meaningful data transmission and exchange between them. Sharing information, resources and processing loads is the main objective of a computer networks. Communication between computers is done using a common communication protocol. Computers use a network interface which is a communication point for interconnecting [2], they are linked either by a physical wire or wirelessly to either a redistributed point or a communication point, the interconnection between nodes “group of nodes” and links [3] allow a message to be transmitted to every computer in the network that what is called telecommunication network.

1.1.1 Computer network types

Computer networks can be classified depending on there geographical scale and size. One or various network topologies are used to build scalable networks, each network is a cluster of network devices. In general there are three major network types are: Local Area Network (LAN), Wide Area Network (WAN), Metropolis Area Network (MAN) [49]

[1]Networking: A short term for Computer network

[2]Interconnection: the physical linking of a carrier’s network with equipment or facilities not belonging to that network

[3]telecommunication links: A communication channel that connects two or more devices

Local Area Network (LAN) A few nodes that are interconnected by a communication link and that uses a common communication protocol to communicate with each other. It is limited to a small geographical area such as a building or a floor. Figure 1.1. The advantage of a LAN is:

- High speed of communication due to the short distance between nodes which allow the communication link to maintain a high speed of data transmission.
- Better service availability and reliability due to the small size of dependents.
- It provide a sense of security as it limits access to only few devices.

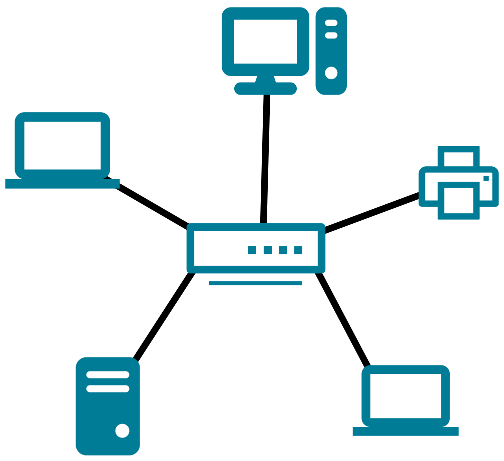


FIGURE 1.1: A LAN network using star topology

Wide Area Network (WAN) Interconnect computers in similar way to LAN but over a wide geographical area like a region or a country.WAN advantage is that it cover a larger distance which means it reaches a wider community, it allows to have distributed servers and a wider range of information. Its drawbacks is that it is slower and unreliable.

Metropolis Area Network (MAN) Its geographically larger then LAN and smaller than WAN, generally it can cover a city or part of a city. It not very commonly used.

1.1.2 Networking key concepts

The key for a successful communication is to guarantee the delivery of data and the correct read of information. In computers that requires a hardware and software collaboration.

Data communication

Data communication is the mechanism and technology used to send information across a physical communication medium[4]. Data communications focuses on the using of physical phenomena to transfer information [17].

[4]Medium: The physical link or channel used for transmission in the network. Wire, fiber and air are the three media

Network Hardware

Network hardware or equipment, is an electronic device required for communication and interaction between devices on a computer network. It mediates data transmission over computer networks. Some core devices are:

Switch: A device that connect multiple devices together on a network.

Bridge: A device that connect multiple network segments[5].

Router: A device that forwards data packets between networks.

Network application

Network application is a piece of software that enables processes to communicate with each other in a network, it mean that an application running on a computer can communicate with other application running a a different computer on the network using a communication protocol.

Packet switching

Packet switching has fundamentally changed the network and laid the foundation for the modern Internet.
Packet switching is a method of grouping data into packets. Packet switching allows multiple transmitters to send data over a shared network, rather than forming a leased line. Packet switching is based on the same basic data communication mechanisms as telephone[6] systems, but uses the underlying mechanisms in new ways. In packet switching, data is divided into small blocks called packets, and each packet contains a header and a payload. The data in the header contains information about the intended recipient, and it is used by networking hardware to direct the packet to its destination. When a packet arrives at one of the devices, the device chooses the path to send the packet and ensures that the packet finally reaches the correct destination. When a packet gets to its final destination, its payload is taken out and used by a network application.

Network protocols

To ensure that the resulting communication system is complete and efficient, a set of protocols must be constructed and agreed upon by all devices on the network. Protocols are designed into a complete collaborative set called a "suite" or "family" to cover all aspects of communication. Each protocol in the suite handles one aspect or function of communication, including hardware failures and other exceptional conditions. The entire suite is designed to allow the protocols to work together efficiently.

[5]Network segment: A portion of a computer network. The nature and extent of a segment depends on the nature of the network and the device or devices used to interconnect end stations

[6]Telephone: Is a communication device that allows two or more users to talk when they are too far apart to hear directly.

1.1.3 Internetworking with TCP/IP

The idea of linking multiple networks together is known as "internetworking". Because new technologies can be added at any time without having to replace old ones, internetworking is far more powerful than a single networking technology [17]. The problem with internetworking was that different networks had different packet switching technologies, and no single packet switching technology could meet all of the needs. The answer was to explore how various packet switching technologies could be linked together. For this type of interconnection, a set of standards was proposed in 1973. That led to these standards being known as the Transport Control Protocol/Internet Protocol (TCP/IP) [15]. Figure 1.2 shows layer modeling of TCP/IP.

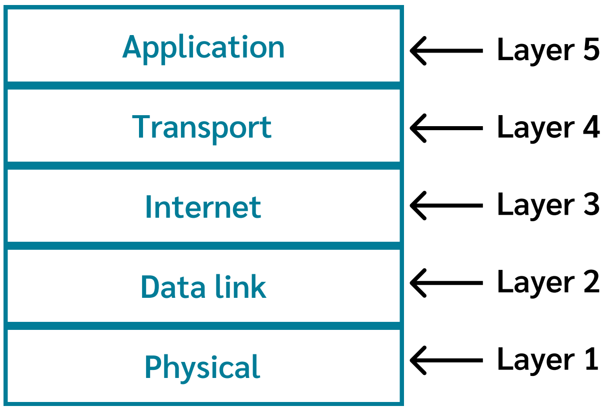


FIGURE 1.2: The layering model used with the Internet protocols TCP/IP[61].

Transport Control Protocol/Internet Protocol (TCP/IP)

TCP/IP is a practical communication protocol that permits hosts to connect over a network. OSI and TCP/IP are similar in many ways. That similarity is due to the mutual influence. TCP/IP model is organized into five conceptual layers which are:

1. **Physical layer:** This layer is responsible for literally moving data link datagrams bit by bit over the links and between the network elements. The protocols here depend on and use the characteristics of the link medium and the signals on the medium[49].

2. **Data link layer:** This layer provides services that allow to move packet from one packet switch to another over a communication channel. It also ensure the delivery of Internet layer packets. The frame, a data link-layer protocol unit, may be transferred from source to destination using different link-layer protocols on different links.

3. **Internet (Network) layer:** The Internet layer handles communication from one machine to another. It accepts a request to send a packet from the transport layer along with an identification of the machine to which the packet should be sent. It encapsulates the packet in an IP datagram, fills in the datagram header, uses the routing algorithm to determine whether to deliver the datagram directly or send it

to a router, and passes the datagram to the appropriate network interface for transmission. An additional function of the Internet layer is to handle incoming datagrams and use the routing algorithm to determine whether the datagram should be processed locally or forwarded. At the internet layer, software removes the datagram header and selects from a variety of transport protocols the one that will handle the packet for the local machine. Last but not least, the Internet layer is responsible for sending and receiving ICMP control and error messages [18].

4. **Transport layer:** Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) enable communication between application programs on different computers. This level of specification encompasses things like data rate specifications, mechanisms to prevent network congestion, and techniques for ensuring that data is received in the correct sequence.

5. **Application layer:** It specifies how two communicating applications interact with one another. Each application selects the type of transport required, which may be a series of individual messages or a continuous stream of bytes. The application program transfers data to the transport layer in the required format.

1.2 Computer networks on organization scope

In most cases, networks are managed by organizations. Networks are categorized as either intranets or extranets by their owners. The Internet is a unique form of network because it does not have an owner[14].

1.2.1 Internet

The internet is a worldwide system of interconnected computer networks that facilitates communication between devices and other networks through the utilization of the Internet protocol suite TCP/IP. These interconnected networks are linked by a wide range of electronic, wireless, and optical networking technologies. These networks range in size from local to global and include public, private, academic, business, and government networks[18].

1.2.2 Intranet

An intranet is a secure and reliable way to access the Internet. It makes information available in a variety of formats at a low cost, with no regard to a user's physical location. The term "intranet" refers to an internet that is only accessible within the boundaries of a single company or organization[14]. It uses the Internet protocol suite TCP/IP but unlike the Internet, an intranet is based on private organizational-controlled servers and it is protected from the public Internet by a security firewall[7] installed on a server or a network hardware to prevent intrusive or malicious software from entering the intranet[14]. As it is illustrated in Figure 1.3.
Intranets are typically built using open standards and on a variety of hardware and

[7]Firewall: Is a network security device that monitors and filters incoming and outgoing network traffic based on the security policies of an organization.

software platforms. Standardization in technology can have numerous advantages. Compared to other methods of publishing and disseminating information within organizations, this one is less expensive and simpler to put into practice [14]. A Local Area Network running the internet protocol TCP/IP can be used as a starting point for a basic network infrastructure.

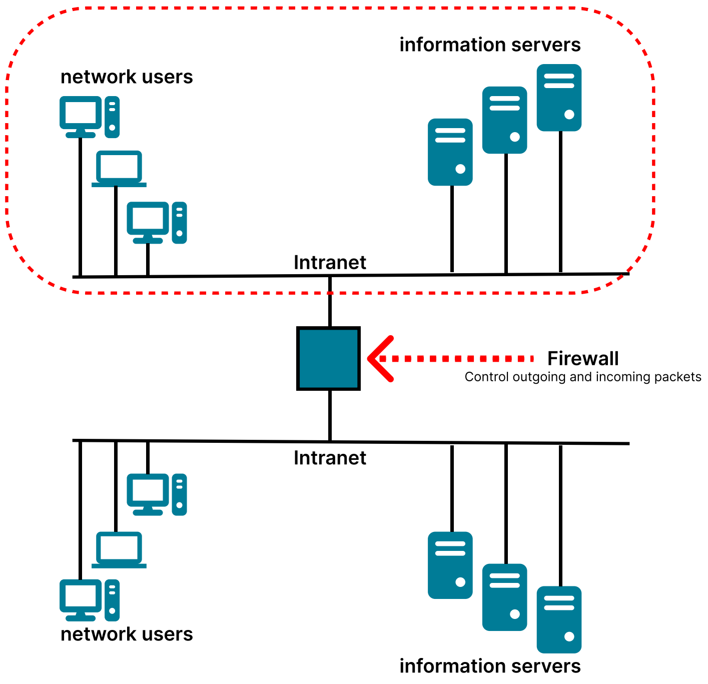


FIGURE 1.3: Intranet Implementation example

1.2.3 Extranet

Extranet or extended intranet is a term used when a firewall allows incoming packets from outside the local network [14].

1.2.4 Benefits of Intranet in comparison to Internet

In recent years, both the internet and intranet have been widely utilized. There are numerous reasons why businesses and institutions prefer intranets over the internet to connect their employees or staff. Intranet is a private network with much smaller boundaries than the internet. Despite its limitations, the intranet provides solutions for a number of unique use cases such as:

Reliability: The intranet is more reliable than the internet because it is restricted to a smaller geographical area and have fewer users.

Consistency: Because the intranet have fewer users, there are fewer known sources of information that make it technically consistent.

Availability: intranet is a local network, so it does not require a provider other than the local server, unlike the internet, which requires an internet service provider to function.

Security: the intranet may be more secure than the internet due to restrictions on access from outside the network, which make it difficult or impossible to conduct a security attack.

1.3 Knowledge management in healthcare organizations

Scientific understanding of diseases, treatments, and pathways of care is growing at an exponential rate in healthcare. Thus, healthcare knowledge is in flux, as new knowledge is rapidly created and applied that can have a significant impact on patient care and patients' quality of life. However, the ability to disseminate, translate, and effectively apply existing healthcare knowledge in clinical practice is incompatible with this growth in knowledge [1]. When making clinical decisions, healthcare knowledge plays a critical role. Knowledge is used to arrive at the correct diagnostic decisions and devise the most effective therapeutic regimens. Healthcare professionals use knowledge to validate prior hypotheses and satisfy additional constraints as they move towards the final decision in a cyclical manner. For example, a patient's evolving health status can be interpreted and a treatment plan devised in this cyclic decision-making process, which dynamically contextualizes healthcare knowledge. As a result, timely access to accurate and clinically relevant information is essential for good clinical decision-making.

1.3.1 The Difference Between Knowledge and Information Management

Information and knowledge management are often misunderstood, despite their literary significance. Information management is a cycle of organizational activity involving the acquisition of information from one or more sources, its custody and distribution to those who require it, and its ultimate disposal via archiving or deletion. The ultimate goal of information management is to ensure that the right data is available at the right time and in the correct location. A computer-based information system is the primary means of accomplishing this goal[81]. On the other hand, knowledge management is a collection of methods for creating, distributing, as well as using and managing an organization's knowledge and information. Since the mid-1990s, the term "knowledge management" has received more attention in scientific papers than "information management" [74]. Knowledge management is either used as a synonym for information management or for the "management of work practices," which are to improve the sharing of knowledge in an organization [97].

1.3.2 Organization culture and knowledge management

Organizational culture is defined as a set of practices, values, and assumptions held by the organization's members and capable of influencing the organization's behavior [36]. The intranet's use as a tool for knowledge management must be integrated into the culture of the organization by incorporating creative learning and communication practices and patterns. Information sharing on intranets is only possible if the technology is integrated into employees' daily routines and the organizational culture [12]. organizational knowledge resides in a constellation of communities of practice [94]. The intranet can be a challenge for these communities because it provides a forum for knowledge sharing and mutual comprehension.

1.3.3 Healthcare knowledge management

The healthcare system is one of the most complicated systems in society [2]. It involves a number of collaborators who work in various fields and who must cooperate in order to provide care to a human being. Healthcare professionals like doctors, specialists, nurses, radiologic technology technicians, psychologists, etc. are involved in the delivery of healthcare. Third parties are also involved, including administrators of hospitals and clinics; human resource and financial managers; the ministry of health; pharmaceutical and insurance firms; educational institutions; and research communities, among others. In addition, while working on the same patient, partners in the delivery of healthcare are dispersed across numerous geographic locations. Healthcare Knowledge Management is the systematic creation, modeling, sharing, operationalization, and translation of healthcare knowledge for the purpose of enhancing the quality of patient care [1]. The goal of healthcare knowledge management is to provide healthcare knowledge in order to create high-quality patient care decisions that are informed, cost-effective, and timely. while knowledge management systems use information technology to control the production, storage, sharing, and use/reuse of knowledge, the use of knowledge management in the healthcare industry faces unique difficulties due to the complexity of the system, the consequences of medical errors, the medical policy, the rapid expansion of medical knowledge [28]. The adoption of Knowledge management strategies can help address the complexity and challenges facing the healthcare sector. Knowledge management in healthcare has the potential to improve patient care by ensuring continuity of care [28].

1.3.4 Health Information System (HIS)

A Health Information System (HIS) is a useful entity within the structure of a broad health system to elevate the health of individuals and the population [52]. Health information systems should be able to generate essential data for use in decision making at every level of the health system with a given amount of resources. A patient's electronic medical record, a hospital's practice management, or a system supporting healthcare policy decisions are all examples of this type of system.
Health Information Systems allows knowledge management to be integrated within healthcare system structure [38].

Examples of Health Information System

Health Information System (HIS) vary depend on the implementation and features. Some examples of Health Information Systems are:

Electronic Medical Record (EMR): Is a digital record of a patient's medical history. Test results, diagnoses, and treatments are all part of it. Other healthcare providers can also access patients' medical information through the use of this system [8].

Practice Management Software (PMS): A software that manages the daily operations of a healthcare facility. It handles scheduling, waiting lines, medical sessions, and billing. The majority of PMS systems integrate an electronic medical record system and an administrative system. It uses a server-client architecture to facilitate collaboration among members [40].

Clinical Decision Support (CDS): Many different subsystems can be accessed and analyzed by CDS systems, including clinical as well as administrative ones. Using this data, clinicians can plan out the best possible course of treatment for their patients. It is possible that the data from a CDS system will help doctors make diagnoses more quickly and accurately, or predict the interactions between various medications. When using a CDS system, providers can choose to use trends and data points from a specific patient population, rather than relying on the CDS system to make decisions for everyone in the population [10].

Remote Patient Monitoring (RPM): Remote patient monitoring allows providers to remotely monitor patients. Wearable sensors can be used to monitor patients' vital signs, blood pressure, and other biometrics and regularly transmit data to the service provider [56].

1.4 Intranets for knowledge management

Intranets are frequently designed to make it easier for members within a single facility to exchange information. Organizational learning can be facilitated by an intranet [19].

1.4.1 Intranet infrastructure

Intranet uses internet protocol TCP/IP which make fairly uncomplicated and inexpensive way to be initiated using existing infrastructure. A Local Area Network (LAN) using internet protocol TCP/IP and following a standard topology makes a basic Intranet infrastructure. The structure may differ depending on how many devices are there, the used networking tool and technologies and the organization functions and structure, but overall intranet serve the same purpose which is interconnecting.

1.4.2 Key components for a basic intranet

There is no upper limit to the scope of an intranet's implementation as long as the network is monitored and secure. To initiate a basic intranet that uses Client-server architecture, the following key components are required to ensure security and availability (Figure 1.4):

Network Hardware: A collection of devices that enable a LAN connection, such as a router, cables, and network interface cards.

Servers (Source of knowledge): A collection of computers that function as a source of information using computer software.

client software: a software utilized by other organization members for communication purposes. It relies on a single information source, namely the server software. It can exchange and manage data with the server software via the application layer and transport layer.

security firewall: A software installed on a server or router to prevent intrusive or malicious packets from entering the intranet.

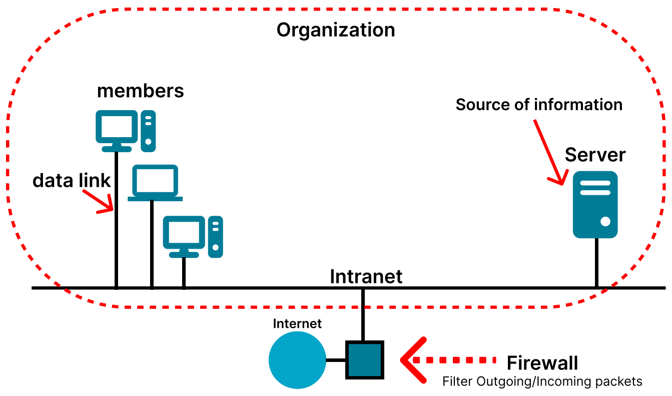


FIGURE 1.4: A basic Intranet structure using client-server architecture for knowledge management.

1.4.3 healthcare knowledge management intranet structuring

The structure of knowledge and the structure of the organizations where the frameworks are applied are taken into account when implementing a knowledge management system. Most models base the classification of knowledge types on various factors, such as whether the knowledge is explicit or tacit and whether it is structured.
The term "medical practice management" is used to refer to a subcategory of healthcare knowledge Management software that manages the daily operations of healthcare facilities. The software's implementation determines its features, but typically it offers the bare minimum functionality to permit information sharing. Medical practice management software can be implemented on an intranet with a server-client architecture, which can be described as the following:

- A server-software application that operates on a specific computer to serve as a centralized information source.
- A client-software that operates on members' computers.
- Computers are interconnected in a LAN network.
- A layer three device such as a router is used to interconnect computers.
- Each computer on the network is given a layer three IP address.
- An IP address is used to locate different computers on the network.
- Several protocols at the transport layer and the application layer facilitate communication between client and server software.

- The server acts as an intermediary to allow communication between members.
- Depending on the features provided by the software implementation, a variety of networking protocols may involved.

1.4.4 Optional internet communication through a P2P network

Enabling inter-corporate communication may be necessary in certain enterprise use cases to exchange knowledge and ideas between organizations.[51]

A P2P(Peer-to-Peer network) is a network of computers that are connected together with no central server. P2P networks are a decentralized network that offers communication between peers with equal privileges. Peers make a portion of their resources available to other peers in a mutually beneficial environment.[51]

Intranets can communicate with each other using a common medium of communication, This kind of communication is known as Extranets. Using the Internet as a medium, server software can expose its resources to other organizations by making them available as a web service. which mean all clients that are locally connected to those servers can exchange information with other servers using the Local server as a gateway. A Server software can be accessed using a public IP address[8] by other server software.

To preserve the security level of intranets, peers are protected by a firewall, which only allows trusted IP addresses. Figure 1.5 shows the communication between two intranets using a P2P network.

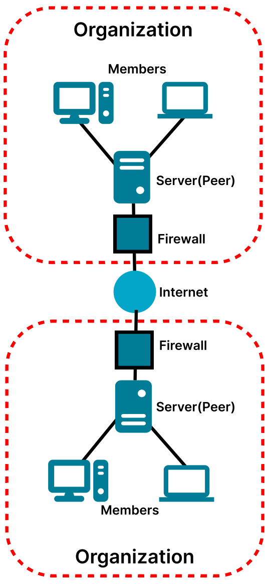


FIGURE 1.5: P2P network communication.

[8]Public IP address: An IPv4 address that is reachable from the internet, it is essential for any cloud-based server resources to be reachable from the internet. Any server resources that need to be directly reachable from the internet must have a public IP address.

1.5 Conclusion

The rise of network technologies presented a new way of interconnecting computers. An organization benefits from networking technologies such as Intranets, which are promising Information Communication Technology (ICT) tools for knowledge management in an organization. Information sharing and collaboration, which help improve an organization's practices and functions, are made possible due to computer networks. In order to be effective, knowledge management must incorporate the majority of an organization's practices and be widely accepted by both members and employees. Every aspect of running a healthcare facility, from scheduling an appointment to attending a medical appointment and keeping track of patient records, must be integrated into a healthcare knowledge management system. Health Information System (HIS) are the result of knowledge management implementation in healthcare, which can be implemented on computer networks such as LAN.

Chapter 2

Access control

Today, we live in world of technology where digital has dematerialized our life, the things that people need are no longer dependent on physical stuff but are satisfied by digital technology, we become a collection of individual pieces of data called Personally Identifiable Information (PII), so this makes people curious about their personal privacy in the digital world. Businesses, governments, schools and even hospitals have transformed digitally. This what we called data privacy and it's having two essential aspects: Access control and Data protection. In this chapter we are interested on defining the access control concept and the established methods to implement it.

2.1 Access control definition

Access control is a process which defines who or what can view and use resources in computing domain. It is a security measure to eliminate the risks on organizations and enforce the data privacy [78, 53]. It aims to protect information security by accomplishing the following fundamentals:

- **Confidentiality**: Keep the information secure and private by letting only the authorized persons to access the resources that can refers to government's secrets, financial information's or any high-level security data.
- **Integrity**: Protect the data from the deleting or modifying by unauthorized persons. For example, most users want to guarantee that password changes may only be made by them or a designated security administrator, and that bank account numbers used by financial software cannot be altered by a third party.
- **Availability**: Refers to the idea of keeping information accessible for usage when required. Attacks on availability include those that target for example Web servers, databases and so on.

2.2 Access control terminology

The terminology depends on several concepts descripting access control models and systems. Every type of access control consists of these concepts which are the following [47]:

User: This term refers to the people who interact with computer system. In many designs, a single user can have multiple login IDs, and these IDs can all be active at the same time. However, authentication mechanisms allow multiple IDs to be associated with a single human user.

Session: A process of user interaction with a system Is called a session.

Subject: The operation systems consist of multiple service process running in background to perform tasks on behalf of the user, this mechanism called a Subject. For example, a windows system can start downloading and installing updates periodically while the user continues progress his work.

Object: Refers to a resource on computer system like files, documents, peripherals and databases, it represents a collection of data basically.

Operation: Is a task or multiple tasks dispatched and initiated by a single subject to achieve a specified goal for example: a windows login, when the user is registering his password, a subject initiate and invoke a combination of processes: one is listening to user input on the keyboard and another to do the password validation at the same time. All of this is happened on behalf of the user.

Permissions: (or Privileges) Are authorizations to perform a task on the system, it represents an association between an object and operation. A specific operation performed on two distinct objects represents two distinct permissions, and two distinct permissions are represented by two distinct operations performed on a single object.

Least privilege: Is security procedure to avoid the problem of individuals having an unnecessary action on the system, the idea is granting the required permissions to perform such operation on object, every user has his job functions, so the permissions associated to him must be satisfied to perform desired functions without giving additional access. The principle of least privilege is primarily an administrative challenge that necessitates the identification of job functions, the specification of the set of permissions required to perform each function, and the restriction of the user to a domain with only those privileges [55].

2.3 Types of access control

Organizations decides to use an access control depending on the needs and the requirements of the security level of the data that are trying to protect and what's suit the environment best.
The major models of access control are the following:

- **Mandatory access control (MAC)**: It is used in environments where the highest level of confidentiality is required, such as top-secret government agencies or in military bases. It is considered the most secured method, but it also the most inflexible. The operating system manages the operations on an object based on security clearance levels, every object gets a label contain the sensitive data (secret), this process of labeling objects uses predefined rules where the administrator decides who gets access to a specific security level and the users cannot change these settings [9].

- **Discretionary access control (DAC)**: The access permission is usually on user's discretion, used in most operating systems for example the NTFS[1] implementation in windows, a user can create a file and set read/write permissions to that file. The owner decides access instead of administrator. It gives the highest level of flexibility but it's come with risk of unauthorized information disclosure and lack of centralized control. it is considered the least secure method for those reasons [45].
- **Attribute-based access control (ABAC)**: This method manage access by processing a set of rules, conditions and the attributes of users and systems, users can have complex relationships to applications and data based on many different criteria like geographical location. For example if a user is travelling and tries to access an object from outside the organization network ABAC might consider his geolocation, his machine, etc... a different user from the network so it takes intelligent decisions [42].
- **Role-based Access Control (RBAC)**: This method balance between flexibility and control makes it the mostly widely used access control mechanism. The concept of RBAC is to create a role with a set of permissions and assign it to a user or group of users, the defined role decides the access rights to the resources which means that users get access directly implicitly by those roles not explicitly [20].
- **Access Control List (ACL)**: Is a method that can be applied on different situations like routing, file system management and SQL implementations, it's an alternative solution for access control methodology. The purpose of an access list is to identify traffic by applying a packet filter with chosen attributes like: interface name, direction of traffic and IP protocol [5]. it has two types of implementations: Standard and Extended

Attribute/ Access Control Type	DAC	MAC	RBAC	ABAC
Ease of Usage	High	Varies	High	High
Performance	Low	Varies with security levels	High	High
Re-usability	Yes	No	Yes	Yes
Single Point Failure	Authorization failure	less	less	-
Authentication Failure	less	Varies	Role based	less

TABLE 2.1: Access control methods comparison.

After acknowledge the differences between the methods from the table, it is noticeable that RBAC is the most balanced in terms of flexibility and control security. ACL method wasn't included in the comparison table because ACL can be implemented on different environments and merged with other methods of access control to enforce the data privacy which make it an alternative solution, that leads us to compare those two methods separately.

2.3.1 RBAC vs ACL

Role base access control can be replacement to ACL model with features provided in data security and privacy. For a wide company security system while ACL is better in

[1]New Technology File System (NTFS): is a proprietary journaling file system developed by Microsoft

terms of implementing security at individual user level and for low-level data. For instance, an ACL can grant access to write on a certain file but it can't manage how a user affects the file. ACL further complicates management by allowing the direct association of users with permissions means that a large number of users, each one with many permissions implies a very large number of relation (user/permission) expected to be managed manually, this propagation of users increase the risk of falling on giving incorrect and inappropriate user access rights. Due to the possible issues associated with this lack of administrative security management, RBAC was introduced. RBAC defines this relation (user/permission) by creating a role with a set of actions or permissions which makes handling a wide range of users much easier. RBAC don't depend on the environment which makes it dynamic, it can be used on different platforms and databases, in network environment for example all the information about the RBAC system is spreaded across the network. Each user has a set of roles defined and known by the network servers, this distribution system make changes on user profile easier and maintainable. On the other side of the comparison, ACL follows a static and a more detailed approach, it's demands user and objects attributes that represents the groups the user joined and the access authorizations such as permissions to perform tasks on specified objects. In order to determine access ACL analyses those attributes of the user and the wanted object to access. Object security attributes are typically (but not always) preserved with the object and are located on a single server (for example, in the file header). As a result, once an object has been discovered, its security properties can be promptly acquired when it is accessed. Changes to object security properties only need to be at a single end point. However, in a network context, all servers must have access to the most recent values of user security attributes. If user security attributes are stored on a single server, then whenever these attributes are changed, that single server must be visited over the network. If a copy of user attributes is kept on each server, whenever a change happened on these attributes must be synchronized over the network which leads to latency and hard management. RBAC supports role hierarchies which will be explained later on upcoming sections. the only disadvantage from this feature is the processing time unlike ACL which is straightforward mechanism. [6]

2.4 Role Based Access Control architecture

RBAC is modern approach to solve the complexity and the cost of data privacy in large networks environments, The process of security administration is facilitated by using roles, hierarchies, and constraints to organize permissions and authorization. RBAC reduces costs within an organization because it recognizes that employees change much more frequently than job duties. If an employee moves within an organization, for example, only his role assignment is changed under RBAC. As a result, revoking his existing privileges and assigning a completely new set of permissions is unnecessary. RBAC has also been discovered to be an effective access control mechanism for workflow management systems.

2.4.1 RBAC structure

RBAC is build on three essential elements: users, roles, and permissions.This Figure 2.1 shows the relationship between these elements.

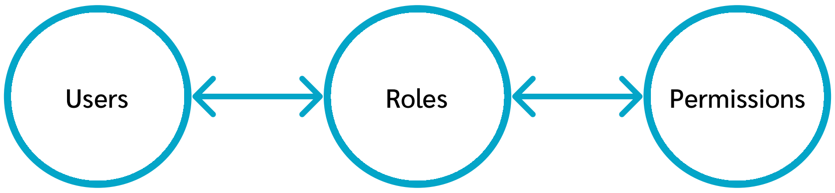


FIGURE 2.1: RBAC Elements relationship[79]

User

The definition is the same as mentioned in access control terminology , users represent an entity who interact with computer system demanding access to an object. It's not necessarily human individuals. Services and computing entities such as a virtual machine or an end-device can also be considered users for RBAC purposes.

Role

The main element is the role representing the formula of these relations. It's contained a set of permissions. Users are assigned to roles. And role determines the allowed permissions to a user. RBAC solutions may include pre-built roles that users can assign to, as well as the ability to create custom ones.

Permission

Permissions consist of the relation between operations and objects. It is the heart of role-based access control, it define a collection of rules that outline the actions and resources that a user would access. For example, if a user is assigned the writer role in such document, permissions refer to the list of actions that the user can perform in that role. As a writer, the assigned user has the ability to access the document (an object) and modify the contents (an operation) but not be allowed to delete embedded links (objects) or the entire document (an operation).
The Figure 2.2 shows two binary relations that set an example of the relationships previously discussed: one links an operation to an object, meaning "permission," and the other links a role to a permission.

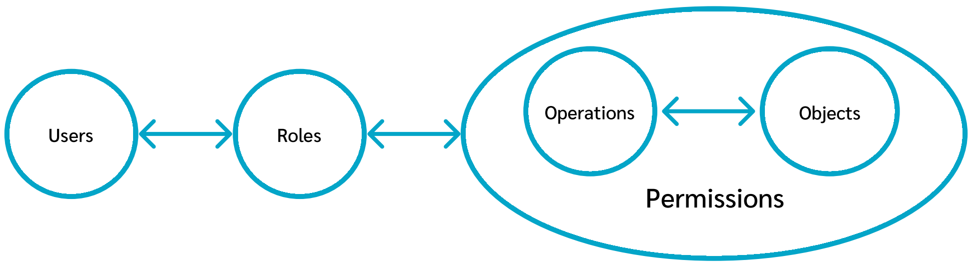


FIGURE 2.2: RBAC elements relationship[21]

The modeling of the structure is summarized as the following [20]:

- USERS, ROLES, OPS, and OBJ (users, roles, operations, and objects, respectively).

- $UA \subseteq USERS \times ROLES$, a many-to-many mapping between users and roles (user-to-role assignment relation).
- $assigned\_users : (r : ROLES) \rightarrow 2USERS$, the mapping of role r onto a set of users. Formally: $assigned\_users(r) = u \in USERS|(u,r) \in UA$
- PRMS = $2(OPS \times OBS)$, the set of permissions
- $PA \subseteq PRMS \times ROLES$, a many-to-many mapping between permissions and roles (role-permission assignment relation).
- $assigned\_permissions(r : ROLES) \rightarrow 2PRMS$, the mapping of role r onto a set of permissions. Formally: $assigned\_permissions(r) = p \in PRMS|(p,r) \in PA$.
- SUBJECTS, the set of subjects (subject is defined in Access control terminology (make ref to it).
- $subject\_user(s : SUBJECTS) \rightarrow USERS$, the mapping of subject s onto the subject's associated user.
- $subject\_roles(s : SUBJECTS) \rightarrow 2ROLES$, the mapping of subject s onto a set of roles. Formally: $subject\_roles(si) \subseteq r \in ROLES|(subject\_user(si), r) \in UA$

Figure 2.3 illustrates an example of the RBAC structure model,User u1 can perform operation op2 on object o2 because p2 $\in assigned\_permissions(r2) \wedge u1\ assigned\_users(r2) \wedge u1\ subject\_user(s2) \wedge r2\ subject\_roles(s2)$.

This structure provides a higher level of flexibility and accuracy in assigning permissions to roles and roles to users. Any increase in control over resource access strengthens the principles of least privilege.

2.4.2 RBAC core features

RBAC implementation contains features to manage resources access and auditing which are crucial to information security. Access can and need to be given based on the requirement. security is easier to maintain when there are hundreds or thousands of employees since unnecessary access to critical information is restricted based on each user's defined job within the company. The next sections are illustration of RBAC core features.

Administrative capability

The management of data authorization is widely regarded as a time-consuming and costly process. The capability on administration is one of the greatest features RBAC supports, users are assigned with roles based on their job functions, responsibilities and authority. Removing Users' assignments or giving new roles can be easily done as the administration decides. RBAC does not grant users permission to perform operations on an individual basis and static approach. Rather, permissions are assigned to their roles. Role associations with new permissions can be created, while deleting or updating can be performed as the organization functions changes. In this way, system administrations can make updates without changing the permissions to every user.

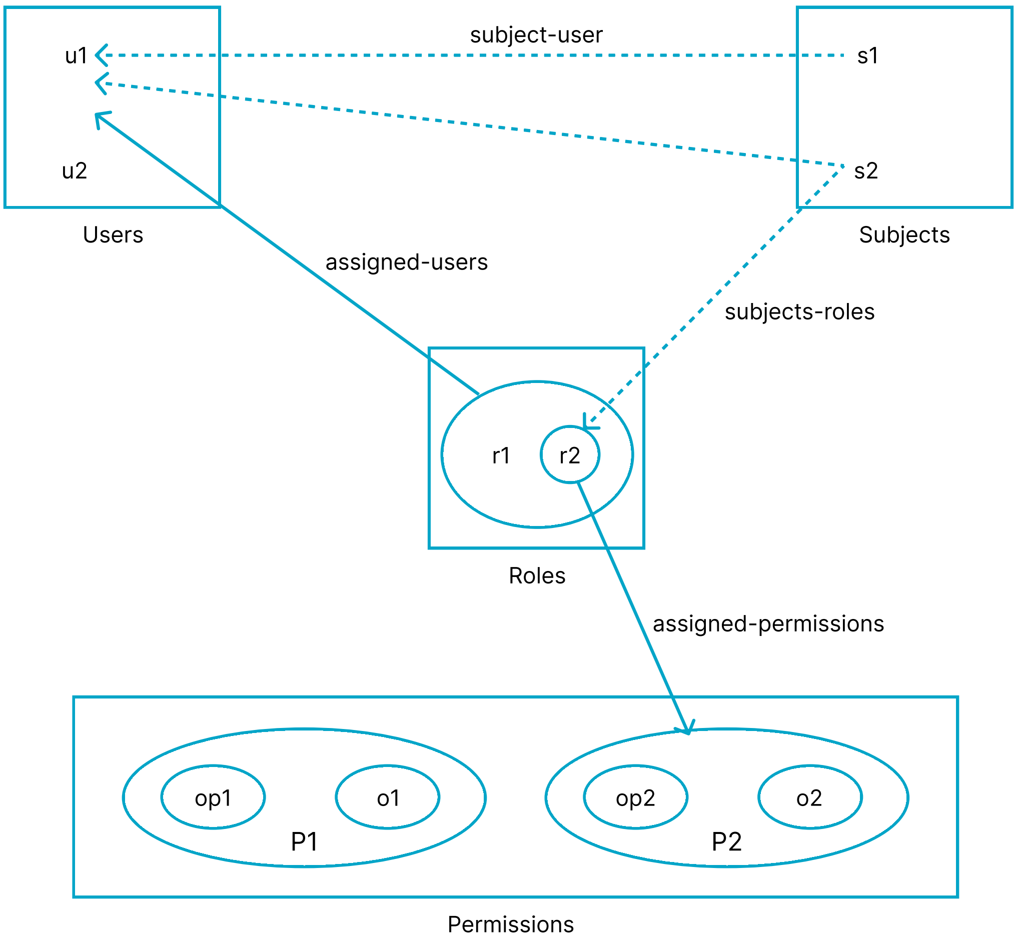


FIGURE 2.3: RBAC structure model[22]

After establishing the needed roles on RBAC system, the term of cloning plays a major part on the management side. Cloning is the practice of granting permissions to a user based on the redundancy of privileges granted to a second user who performs a similar function to the first user. Although cloning may be a quick and efficient method for establishing permissions, due to the texture nature of permission assignment.
Another advantage of RBAC is that system administrators can specify resource access requirements at the same level of complexity as typical enterprise business processes. The RBAC model requires system administrators to create roles for numerous job positions within the organization. For example, in a hospital: the role of a doctor is to create, modify or delete patients' diagnosis, prescription medications or tests, the role of the assistance is only reviewing patients' profiles and manage their appointments.

Role activation

Like the other types of access control, RBAC architecture define the functions of subjects and objects, and it's divided by two separate but related components [54]:

Static components: Means that the concept of subjects and its functions are not needed, the relation between RBAC elements takes direct approach on managing roles and user requests. The user with assigned roles performs his tasks without the help of subjects which are designed to do background services on the first place.

dynamic components: The concept of subjects which refers to active entities whose access to roles, activities, and objects is used when implementing a dynamic security policy on a computing system. Every request is fulfilled by a subject acting on the user's behalf, the use of an identifier is important to uniquely reference each subject . If the identifier is eligible for a function, it evaluate and make the subject active in the task. A user could have numerous subjects assigned to them at once and these subjects could have a unique mix of active roles. This feature follows the concept of least privilege by allowing users to be allocated to numerous roles, but he can choose to activate any portion of them to suit his needs. Figure 2.3 shows exactly the function of dynamic components using subject assigned to users and roles.

Constraints and exceptional cases

Constraints are an important aspect of the RBAC model. specially in the case of mutually disjointed roles [29]. In the healthcare facility for examples, there are different kinds of constraints:

Constraints on role assignments to users: A user cannot have two disjoint roles at the same time. Disjoint roles like: general doctor, physiotherapist, Ortho prosthetist and an assistance. Neither of them can take the role of another.

Constraints on assignments of permissions to roles: Each role have a set of permissions required to perform user functions based on capabilities and responsibilities. For example: the Doctor role have all the permissions, the physiotherapist role only has permission to access the list of patients who require rehabilitation and modify the number of sessions, the Ortho prosthetist role have the permission to see the radiology file and personal information only, the assistance role have the ability to see patients' profiles, add to queue and see statistics but cannot modify patients record or add new patients.

Constraints on users: All users must be approved by the administrator.

Constraints on roles: Roles cannot be assigned to users until they are approved.

Constraints on permissions: Each permission corresponds to a specific role authorization level.

Exception cases: In case of absence or during days off, the doctor for example can delegate all his permissions to the assistance.

Global users and roles

In order to create and maintain direct links between RBAC users and local user IDs as well as between RBAC roles and local groups, it must first map an enterprise-level RBAC view onto a system-level view. Then, local data represent resources in terms of the user IDs and groups created by the RBAC system can be guarded via the local administration interface. according to this strategy, the RBAC system connects user IDs of accounts across several systems to a single user at the corporate level. With this way, it is possible to control all of a user's ID from a single user point. On a specific system, user IDs are frequently categorized into user groups. In light of this, instead than having to approve each individual user, a security administrator can permit a group to use a resource. On an enterprise level, in order to bind RBAC entities to system privileges, groups and user IDs are essential components [23]. This feature makes administration and management on different levels of environments easier and more organized. Figure 2.4 shows an example of Mapping global users and roles to local user accounts, groups, and permissions.

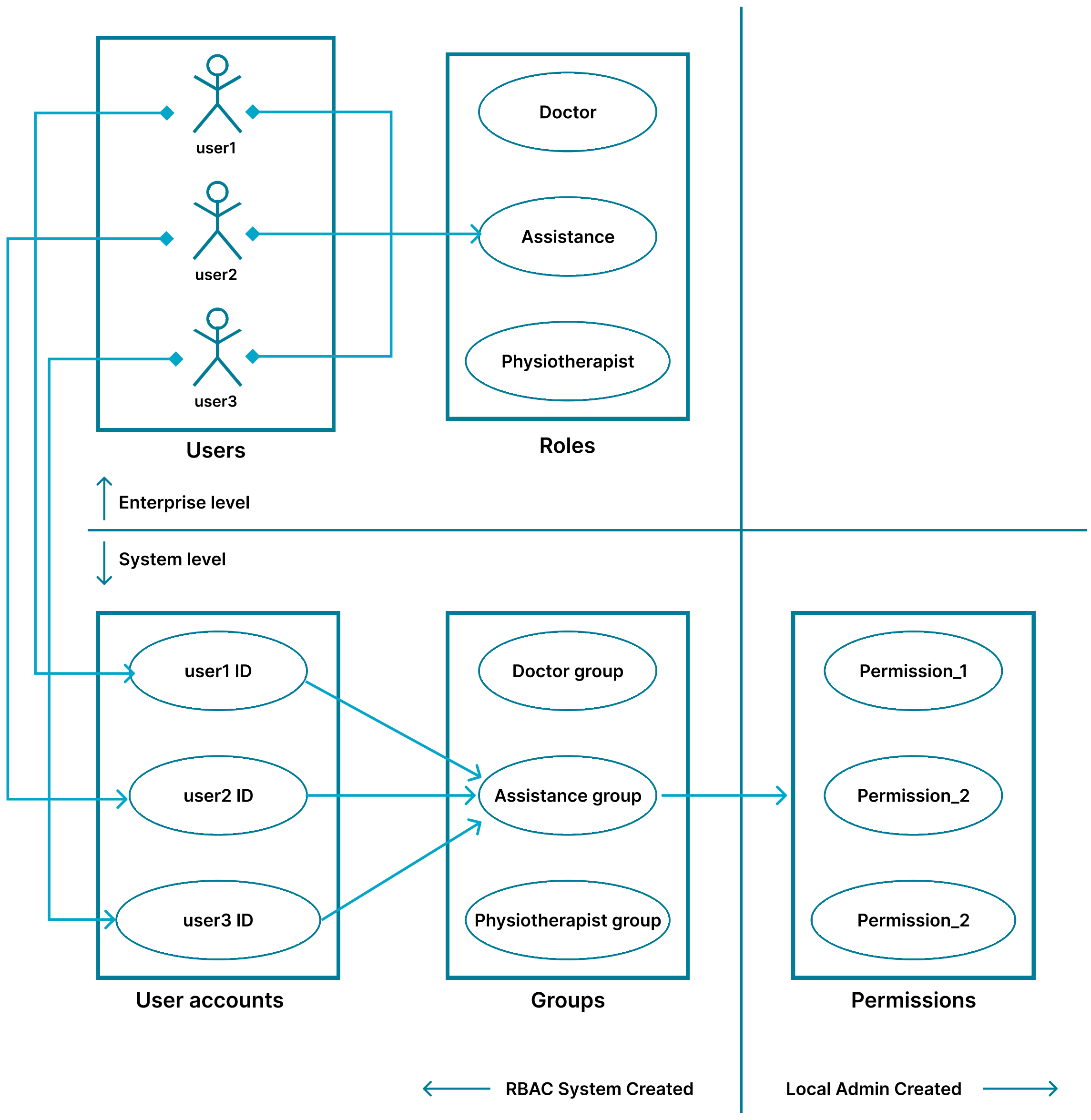


FIGURE 2.4: Mapping global users and roles [23]

Role Hierarchies

One of major features introduced by Role-based Access Control is the inheritance, role hierarchies utilize the structure described earlier in terms of managing user privileges. It allows a role to inherit permissions from all of its sub-roles using a third type of authorization is created by the role inheritance relation. If a role A inherits role B, it means that all of B's permissions are accessible via A. specifically, role B's permissions

are a subset of those granted to role A [64]. Administrators are better able to develop access policies in terms of organizational-specific functions and business structures through the formation of relations.
Figure 2.5 describes a basic example of inheritance, the roles A and B inherit the roles C and D. Since the inheritance of permissions and role memberships is reflexive and transitive, any user that is assigned to the role A is authorized for the permissions that are assigned to the role A and authorized for the permissions that are assigned to the role's B and C. Not all roles have to be hierarchically related. The roles A and B are not hierarchically related, but they can inherit some or all of the same roles, as is the case of A and B.

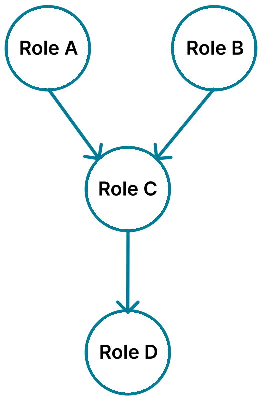


FIGURE 2.5: basic example of role hierarchies

For instance, the functions, responsibilities, and tasks that make up a role's permissions can be segmented into lower-level roles. Once these roles are created, it can be used to generate higher-level roles. No matter how much time and attention are put into developing a hierarchy of job responsibilities, the payoff is both immediate and long-term. Increased administrative productivity and the capacity to better define and analyze access control policies are two of the main advantages of this feature.
Role hierarchy follows some properties to achieve the wanted goal and function. Simple inheritance is the reflexive-transitive closure of the immediate inheritance. A first role logically inherits a second role if all of the second role's permissions are also present in the first role and all of the first role's users are also present in the second role. The advantage of being able to visualize and reason about the distribution of permissions while avoiding role permission and membership redundancy is provided by the role hierarchy, which helps define new roles in terms of current roles.
Another property is permission and user membership inheritance relationship which describe inheritance schema refers to a collection of permissions on one side and collection of users on the other side.The figure 2.6 represents a role graph illustrate the schema.

The most powerful roles can be found at the top of the hierarchy or role graph, those roles have more privileges and fewer authorized users and the roles towards the bottom of the graph are more generic roles, those roles have less privileges and a greater number of authorized users. When a user is given a role, the permissions of the given role and the inherited roles are passed down to that user, in the example shown on the last figure, The allowed permissions for U5 and U6 that are assigned to R3 are P7 and P8, as well as P4, P5, P6, P1, P2, and P3 by inheritance.

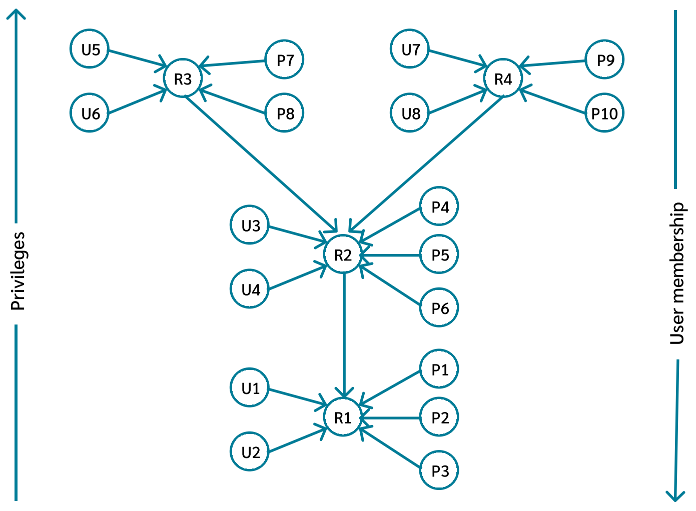


FIGURE 2.6: Combined user and privilege inheritance [24]

The ability to express and enforce enterprise-specific security policies and simplify the traditionally complex security procedure are the primary factors behind Role-based Access Control. RBAC is a framework of policy-rich mechanisms, as was previously demonstrated, and the configuration of this framework depends on organizational policies. RBAC can now be adapted to any organizational structure and way of doing business as a result. The policies put in place under RBAC could also change over time as business and organizational structures and security requirements do, this evolution leads to appearance of new implementations of RBAC, each implementation has its own purpose.

2.5 Comparative study Between RBAC implementations

Vendors have been incorporating Role-based Access Control (RBAC) features into their databases, system management, and operating system products as a structure in recent years, however there is no consensus on what constitutes an effective set of RBAC capabilities. RBAC models such as [RBAC/NIST, I-RBAC, C-TMAC...] have been proposed.

2.5.1 RBAC/NIST

The National Institute of Standards and Technology (NIST)[2] has initiated an effort to make Role-based Access Control (RBAC) available and promoted for internet web servers. The National Institute of Standards and Technology role-based access control model, or RBAC/NIST, is a Web-based version of the standard Role-Based Access Control protocol. Since there are no restrictions on which browsers can be used with RBAC/NIST, therefore any server improved with RBAC/NIST is supported on any browser. It offers the benefits of Role-based Access Control for Internet environments,

[2]NIST :Is a Commerce Department lab for physical sciences. Its purpose is to promote American innovation and competitiveness.

and because it can be integrated into existing systems without modifying server code, it is usually applicable to Web servers [30].

2.5.2 C-TMAC

Context and Team-based Access Control (C-TMAC) adds dynamic access control to Role-based Access Control. It uses active techniques for runtime security management as tasks progress. Implementing C-TMAC in healthcare intranets is proposed. This approach proposes that all clinical tasks can be created during runtime. C-TMAC includes users, roles, permissions, sessions, teams, and contexts. In conventional healthcare, a session is the length of time needed to complete a health task for a patient. Individual users can associate and control a session. A user initiates a session during which he engages a subset of his roles. Context is information about the people (e.g., patients) affected by an activity and their personal and medical data objects. Place and time may also be considered. A team is a collection of users with distinct roles working to complete a certain task. Multiple teams can share the same context. A user can belong to as many groups as desired, each with its own users [32].

2.5.3 I-RBAC

Intranet role-based access control, or I-RBAC, is an intranet security solution based on a Role-based Access Control that provides effective security management based on several levels of role authorizations. Permissions, Network objects, and roles are the three core components of I-RBAC. Permissions are divided into local permissions, which are a collection of privileges that a network object has on the different network objects available in specific servers, and global permissions, which are a set of privileges on the servers of the entire intranet, the main difference between local and global permissions is the level of granularity of the permission set, which can be applied to network objects or servers. A network object is a type of network entity that holds data, provides specific services, or represents network hardware. And a role that represents access control at a higher level. It can refer to a single network item or a group of them and is connected with various permissions. The hierarchy of roles is designed in such a way that the higher a role's position, the more privileges it has, and vice versa [88].

2.5.4 Comparison

The main objective of comparing these approaches of Role based access control is to see which one is more secure and easy to manage and organize based on how policy and data privacy problems have been treated. the table 2.2 shows a comparison between RBAC implementations by key elements.

Results

In this study, three collaborative RBAC models have been selected for comparison. Based on a number of key factors through comparison and analysis, it is determined

[3]separation of duty: A mutually exclusive relationship between two or more roles is defined by a separation of duty constraint. A separation of duties constraint might say, for example, that no user can be a member of the Purchaser and Accountant roles at the same time.

Key elements	RBAC/NIST	C-TMAC	I-RBAC
Roles are based on the context		X	
Team based		X	
Roles assigned to Users	X	X	X
Roles are separated to Global and Local role			X
A User/subject can be a member of many roles	X	X	X
Roles and permission can be granted during the runtime	X	X	
Roles can be assigned for a time-period		X	
separation of duty constraints[3](SSD and DSD)	X		
limits on the number of users that can be authorized for a role	X		

TABLE 2.2: RBAC implementations comparison [30, 32, 88, 57]

which RBAC model is superior for which reasons and in what collaborative setting. While the evaluated RBAC implementations are comparable, they all adhere from the RBAC model. I-RBAC, C-TMAC, and RBAC/NIST all shares the same core RBAC concept, but each adds features that are suitable for its use case. Choosing the best implementation for an intranet system based solely on features sufficiency, as each implementation has its own purpose, features, and fallbacks. The objective of I-RBAC implementation is to provide secure access to shared resources for objects in an intranet. However, it does not address role consistency concerns, which can be described in terms of local-role and global-role databases. In contrast to C-TMAC, which does not produce such inconsistency, intranet environments necessitate the evolution of responsibilities in order to maintain database consistency and reflect new security standards. The C-TMAC model is suitable for a teamwork-based environment and has some additional useful features, such as sessions, but it imposes restrictions on the RBAC model and is expensive in terms of management, administration, and the reality of roles and permissions, making it unsuitable for all systems. RBAC/NIST is implemented for use in both internet and intranet web servers and can reduce the complexity and cost of authorization and data privacy management. RBAC/NIST adds additional rules and restrictions to the RBAC model, making it a secure and dependable model. However, like C-TMAC, RBAC/NIST is context-aware, so it includes rules that are suitable for an intranet application that utilizes the internet protocol TCP/IP.

2.5.5 Conclusion

Despite the fact that intranets can provide significant benefits to a corporation or organization, security concerns remain. In order for intranets to realize their full potential in workplace computing, access control techniques that are both convenient and cost effective must be in place. RBAC/NIST is especially appealing for intranet applications since it simplifies and lowers the cost of authorization administration. Furthermore, RBAC/NIST provides a function for specifying and enforcing sophisticated security policies that are sometimes impractical or even impossible to apply using traditional access control systems, and according to the comparison study, the typical RBAC/NIST model outperforms the other extended models in privacy and sharing scenarios. It is the most appropriate and standard model for businesses where privacy is the major concern while exchanging data, especially in healthcare environments.

2.6 RBAC implementation for a Healthcare Information System

It is important to take information access control seriously, especially when it comes to medical records. The structure of the healthcare system must be taken into account when implementing a specific RBAC/NIST model. Medical practices and policies must be reflected in an effective RBAC/NIST implementation.

2.6.1 Roles structure

In the healthcare structure, role conception is already established. Each member of a healthcare facility plays a distinct role, and that role not only enables them to access information but also to perform duties. With that, some modifications must be made in order for RBAC/NIST to reflect that behavior.

Role

In a healthcare information system, A role represents a physical clinic or a physician speciality. A healthcare information system includes a specific information entity for each medical clinic and hospital department. Medical records and appointment bookings can be kept separate because of the ability to segment information into multiple sections or entities. Access to an information entity in the information system is granted by a role with a specific set of permissions. Role-specific medical practices and tasks include:

- A role with sufficient permission can perform medical sessions for patients.
- A role with sufficient permission has full control of a specific waiting queue.
- A role with sufficient permission owns a specific booked appointment list.

Assist role

An assist role is a role that performs tasks in support of another. It can be created by a standard role owner. An assist role does not have an information entity and it can only manage the linked role's information entity. It can perform tasks such as adding patients to a waiting list or scheduling appointments. Figure 2.7 shows the relation between assist role and regular role.

2.6.2 Information entity

A healthcare information system includes multiple entities for each medical clinic and hospital department. Each information entity allows access to the following information:

Appointment queue: A list of appointments represents the waiting room in a healthcare department or a clinic.

Booked appointments: A list of scheduled appointments.

Medical session: A medical practice or task, which is a meeting between a doctor and a patient.

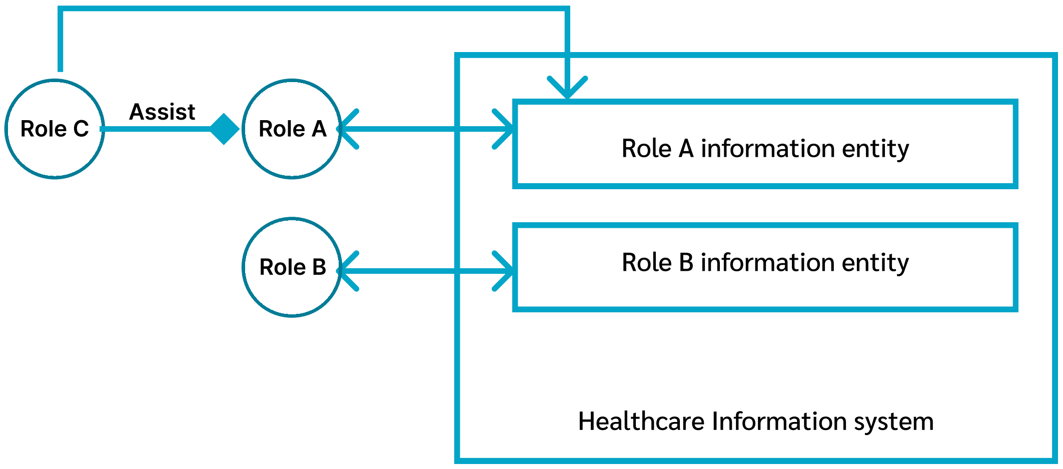

Role C
Assist
Role A
Role B
Role A information entity
Role B information entity
Healthcare Information system

FIGURE 2.7: The relation between roles and HIS

Medical records: A collection of medical information about patient.

2.6.3 User

A user in a healthcare system is a medical staff, roles define the job of users in the system.

2.6.4 Information entity Relationship

An information entity is accessible to users through a user-role relationship. Figure 2.8 illustrates the relationship between an information entity, users and roles.

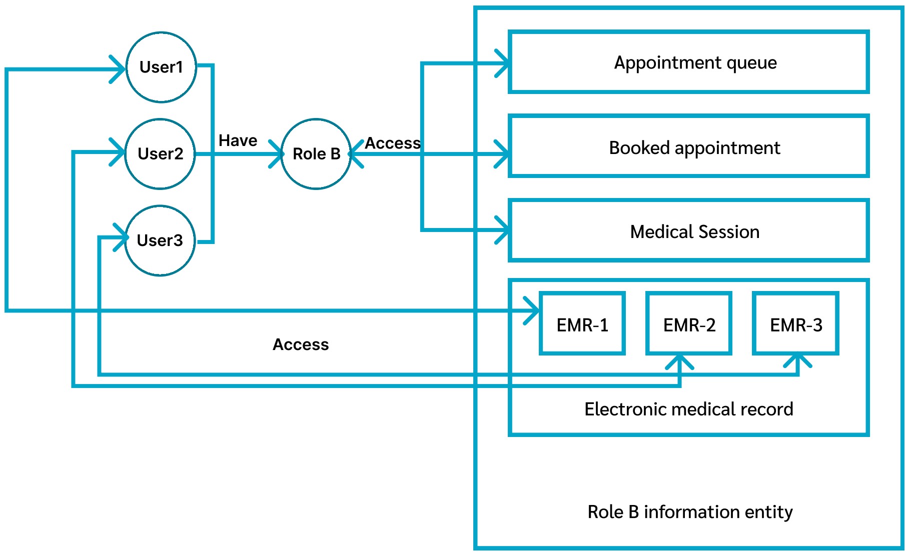


FIGURE 2.8: Information entity relationship.

2.6.5 Owner privilege

The owner is the member whom computer is running the server. Members and roles can be added and managed with this privilege. Even though it isn't an administrative privilege, it can be used to create a new administrative position.

2.6.6 Predefined roles

Pre-defined roles are roles with a set of permissions predefined for the context of a healthcare facility. These pre-defined roles are designed to save time and reduce the complexity of role creation.

2.6.7 Permissions set

A permission set is a set of rules that follow the structure of healthcare. Policies and structure of a healthcare system dictate how permissions are defined. They allow roles to be defined in a meaningful and specific way. Permissions can be administrative or practice related. The table 2.3 defines the permissions set.

Administrative permissions	Administrator
	Manage members
	Manage roles
	Manage patients
	Access to patients list
	View clinic insight
	View medical record
	Mange clinic
	View data collection
Practices permissions	Add patients
	Perform Medical session
	Perform Medical diagnosis
	Control a Queue List
	Control a scheduled appointments list

TABLE 2.3: Health Information System permissions set

2.7 Conclusion

A wide range of reasons has motivated the development of RBAC-integrated prototypes for various enterprise technologies. Better policy support and administration simplicity are common threads in all of these motivations. As a result, it comes as no surprise that some are considering using RBAC for access control.

Chapter 3

The protection of information and the policy of healthcare system

The majority of developed nations have regulations in place to protect patient data. As one of the most sensitive types of data, security and protection must be considered during the development and implementation of healthcare systems. Although it may seem obvious, the need for healthcare information privacy systems justifies the existence of these systems. The examination of data and information is essential to any study of healthcare information systems. This chapter focuses heavily on healthcare-specific data and information, such as patient data, as well as the policies and methods used to protect it and ensure a clinical treatment's compatibility with this type of sensitive data.

3.1 Data protection technologies and practices

The task of securing healthcare data is based on data privacy, which is subdivided into data protection and access control. The sections that follow concentrate primarily on data protection.
Data protection aims to achieve the following five key purposes [90]:

1. **Confidentiality:** Refers to making sure that no one other than the intended recipient has access to any of the patient data.
2. **Integrity:** Involves preventing unauthorized parties or individuals from manipulating or erasing patient data.
3. **Authentication:** Is verifying that a person is who they claim to be.
4. **Accountability:** Traceability is an essential part.
5. **Availability:** Is a term that refers to the availability of patient data to authorized individuals on a request basis.

Personal Health Information (PHI) [1] and Personally Identifiable Information (PII) [2] are typically protected by data security measures. It is essential to multiple systems, including the healthcare system. By securing sensitive data, organizations can prevent data breaches and reputational damage, as well as comply with regulatory standards more effectively.

[1]Personal Health Information (PHI): an individual's medical records

[2]Personally Identifiable Information (PII): representation of data that allows a person to be identified

There are a variety of storage and management choices to select from when it comes to protecting the data. Restricting access, monitoring activities, and responding to threats are all made easier with the right solutions. The following are a few of the most widely used methods and tools:

3.1.1 Data loss prevention (DLP)

Data loss prevention is described as a method that detects and prevents potential data breaches and data sniffing [3] by monitoring sensitive data in various situations (in-use like endpoint actions, in-motion like network traffic, and at rest which is data storage) [87].

3.1.2 Authentication

The process of identifying a user's identity is known as authentication. Combining an incoming request with one or more unique identifiers is the objective. On a local operating system or in a database server, the credentials provided are checked against a record of the authorized user's information [13].

3.1.3 Cryptography

The use of cryptography that is probably most frequently known of is to keep information secret, but it also has other implications. Verifying information and establishing identification are two that are particularly significant [46].
In general, cryptography can be divided into three categories:

Secret Key Cryptography

(Or symmetric encryption) Each of the parties that are trying to communicate has access to the same information, which is the Key, and they both keep that information private from everyone else. Most secret key encryption algorithms, or ciphers, are just mathematical transformations of the information to be encrypted along with the secret key itself, the size of the secret key is a critical factor in determining a cipher's effectiveness. The difficulty of unlocking the encryption increases with the size of the key. The figure 3.1 is an illustration of how Secret Key Cryptography works [72].
The most common algorithms used are DES [4] and AES.

Advanced Encryption Standard (AES) Advanced Encryption Standard (AES) was introduced to replace the Data encryption standard (DES), it is a newer form of encryption. Symmetric-key cryptography largely relies on AES, which was standardized and is now one of the most widely used algorithms in the field. As the gold standard in symmetric-key encryption, several companies require their staff to utilize AES in all of their connections. TLS is also particularly dependent on this standard.

[3]data sniffing: Monitoring and collecting all data packets traveling via the network
[4]DES: NIST has released the Data encryption standard (DES), symmetric-key cryptography.

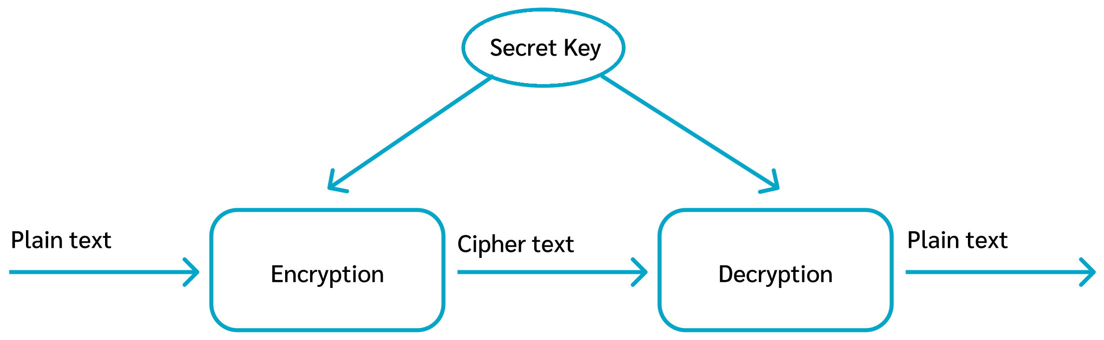


FIGURE 3.1: Secret Key Cryptography [72]

Public Key Cryptography

(Or asymmetric encryption) Encryption and decryption are accomplished through the use of distinct keys shared by the two parties. Using public-key cryptography, just one of the two keys is required to remain private, the other key which is the public key does not need to be kept a secret at all. Asymmetric encryption relies on mathematical obstacles that are much simpler to create than to solve [73]. Even though users may want to identify themselves, weak authentication techniques can leads to privacy risks. A network observer could watch users' authentication sessions to identify or track them when using a network authentication protocol. If a user puts their credentials in the wrong service, they may reveal their identity to a malicious attacker. Finally, following authentication, two endpoints can establish and maintain a session. Data sent between users or from a user to a system should be encrypted to prevent unauthorized personnel from reading it. The encryption technique uses an algorithm to change the content of the data, which an encryption key is required to reverse. Even if data is taken, encryption prevents unwanted access by making it unreadable [69].
The figure 3.2 shows how public key encryption works. For example, the Public key Register contains the public keys of each and every one of its users. If user B wishes to send a private message to user C, then user B encrypts the message using user C's public key. When user C receives the message from user B, the message can only be decrypted by user C, using his private key.

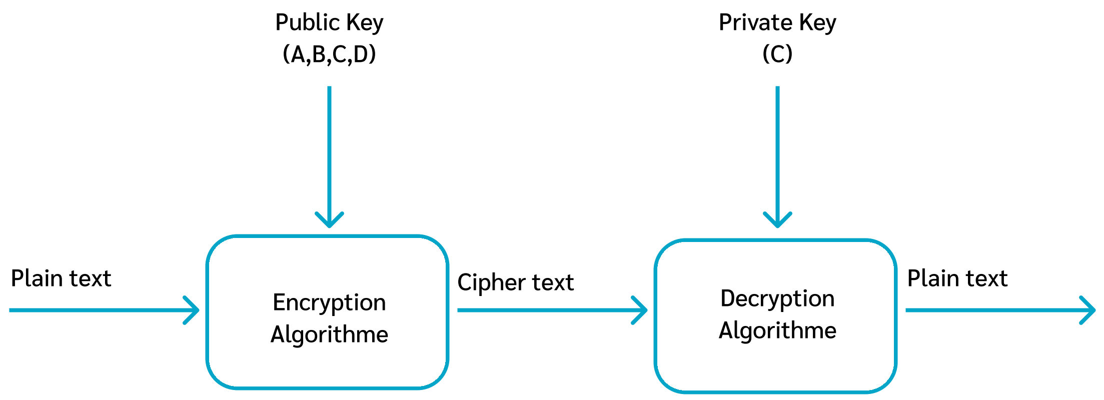


FIGURE 3.2: Public Key Cryptography [73]

The most frequent public key encryption algorithm used is Rivest Shamir Adleman algorithm (RSA).

Rivest Shamir Adleman algorithm (RSA) Rivest, Shamir, and Adleman are the three men who invented RSA, which is named after them. It's a two-way public key algorithm that can encrypt and decrypt data.
A user should generate a public key and a corresponding private key, this can be done by choosing p and q as two large primes (probably around 256 bits each). Make a sum of them and refer to it as n. The factors p and q will be kept secret.
For the public key, select a number e that is close to prime to $\phi(n)$. Since p and q are already known, then n is (p-1) (q-1). The public key is the pair (e, n). To generate the private key, it starts by finding the number d which is the multiplicative inverse of e mod $\phi(n)$. The private key is the pair (d, n). The encryption of certain message m (< n), starts by computing the cipher-text c using the mathematical formula $c = m^e \mod n$. With help of the public key. The only way to decrypt c is to use the private key by computing the second mathematical formula $m = c^d \mod n$. [62]

Hashing

The process of hashing involves creating a value from text or a list of integers using a mathematical formula called a hash function. It is a mathematical procedure that transforms a given numeric or alphanumeric key into a significantly small useful integer value. The hash table's index is created using the mapped integer value [76].
The pair possesses the form (key, value), where it can use a function that converts keys into values to obtain a value for a given key. The goal of a hash function is to determine the key for a specific object. If "i" for example is the key for an array A, for instance, the value can be obtained by looking to A[i] [85]. The figure 3.3 shows the process of Hashing plain text.

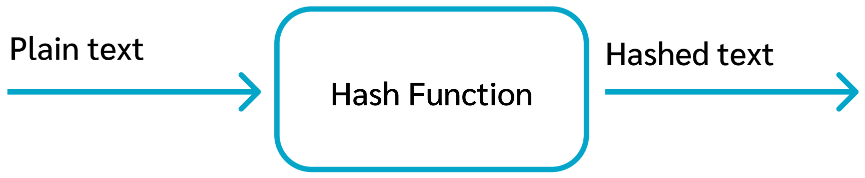


FIGURE 3.3: Hashing Cryptography [84]

There are multiple hash functions that use either numeric or alphanumeric keys like:

- Division method
- Mid Square method
- Folding method
- Multiplication method

3.2 The policy of the healthcare system

3.2.1 healthcare information definition

According to the Health Insurance Portability and Accountability Act (HIPAA), "health information" is defined as all information that relates to a patient's physical and mental health or condition, whether it is verbal or recorded, or written in any form by a healthcare provider.
Personal Health Information, or PHI, is the term used by HIPAA to describe this type of data [92].

3.2.2 The Need for Privacy in the healthcare system

Health information can have a significant impact on a person's life, because of the level of sensitivity required to offer healthcare. Patients' bodies must be physically examined and may need to be treated with more vulnerability than they would allow for anybody else in their lives [83]. Every patient should have the right to control access to personal information about themselves, is particularly necessary for circumstances when the disclosure of such information could result in discrimination, loss of rights, or physical or emotional harm to the individual.
Healthcare providers have a responsibility of fidelity, and having failed to do so would be a violation of that duty. Disclosing a patient's private information would also violate the promise of confidentiality that is a part of medical practice. Working as a doctor for example necessitates patients to share their most private information which represents PHI during a medical practice like diagnosis, and the doctors should live up to the confidence that patients invest in them. In the absence of guarantees of privacy, patients in need of medical treatment would be discouraged from seeking it out because of their fear of releasing their personal information to the public [92, 3].

3.2.3 HIPAA Privacy and Security Rules

Standards for safeguarding Personal Health Information (PHI) were set by the Health Insurance Portability and Accountability Act (HIPAA). Protecting PHI while permitting the flow of health data was the goal of the HIPAA Privacy and Security Rules, both the healthcare provider and the patient are subject to these regulations.
The Rules establish a balance between protecting the privacy of those seeking care and recovery, and allowing the use of critical information. The Rules are flexible and comprehensive to address the wide range of uses and disclosures that need to be handled because of the diversity of the healthcare systems.
These aspects of security are addressed by the HIPAA security standards [35]:

- **Administrative security**: Appointing a specific individual to be in charge of security.
- **Physical security**: : Necessary for safeguarding electronic systems, equipment, and data.
- **Technical security**: Authentication and encryption are used to limit the access to data.

The following are the main rules of HIPAA to establish those aspects of security [89]:

- **Privacy Rule:** Individuals' PHI and medical information are safeguarded under the Privacy Rule, which sets limits on what can and cannot be performed with them and how they can be used or disclosed. In addition, every patient has the right to view and request a copy of their medical records, as well as to demand corrections to their records.
- **Security Rule:** To ensure the security of electronic PHI while it is being stored, accessed, and transmitted, the security rule defines and regulates the applicable standards and procedures. There are three levels of security safeguards to choose between. Protecting electronic systems, data, and equipment in the facility and organization requires a combination of administrative, technical, and physical safeguards. Hardware, software, and transmission risk analysis and risk management protocols fall under this rule.
- **Transactions Rule:** Health and Human Services (HHS) adopted standards for the electronic exchange of healthcare data in accordance with HIPAA. This rule applies to transmissions involving a code set that refers to encrypted data. Appropriate use of these codes is essential in order to protect PHI and medical records from manipulation.
- **Identifiers Rule:** In accordance with the HIPAA Administrative Simplification regulation, the HIPAA Identifications Rule establishes unique identifiers for health organizations in HIPAA transactions. Standardization, efficiency, and uniformity will all be enhanced through the use of these specific identifiers.
- **Enforcement Rule:** this rule covers some aspects of healthcare organizations like: New federal privacy and security leak reporting requirements, including HIPAA security and privacy requirements, are being implemented, as are new privacy requirements and accounting disclosure restrictions.

3.3 Data protection over the network

This section discusses how to protect network data flow as well as how to secure communications between two parties.
Two stages must be validated in order to establish secure communications: authentication between the parties and secure transmission through an unsafe channel.

3.3.1 Authentication on a network

Users can only communicate within a network if they can verify their identity. After establishing this link securely, communications can proceed. User authentication is a crucial aspect of protecting systems and networks.

Authentication Factors Categories

To verify the identity of an online user, there are three categories of factors that can be used[7]:

- **Knowledge factors**: Knowledge factors are the most common type of identity authentication, requiring the user to demonstrate their knowledge of hidden data. Passwords, PINs[5], and challenge responses are all examples of these.
- **Ownership factors**: (or Possession factors) are like a key to a security lock. They are the physical objects that an authorized user must have in their possession in order to establish a connection to a client computer or portal. A credit card and one-time hardware or software password token would all fall under this category.
- **Inherence factors**: Bio-metrics, or a person's physical characteristics, belongs to this category.

Types of Authentications

There are numerous varieties of authentication systems, including [7]:

- **Single-Factor authentication**: This was the initial security technique to be created. The user must input their username and password in order to verify if they are logging in or not using this authentication system. Now, if the user's username or password is incorrect, they won't be able to access the system or login.
- **Two-Factor authentication**: Two-factor authentication (also known as 2FA) confirms a user's identity by combining two multiple components from two distinct factor categories.
- **Multi-Factor authentication**: For better security, it can combine more than two types of authentication factors, unlike 2FA, which only uses two categories of factors.
- **Strong authentication**: This type of authentication is much like two-factor authentication (2FA) But unlike 2FA and multi-factor authentication, strong authentication requires the use of digital certificates to provide users with an even higher level of verification.

3.3.2 Practical Aspects of cryptography

For instance, data communication can be intercepted and sent to unauthorized stations. This necessitated the use of encryption.

Using secret key cryptography

When using secret-key cryptography, the parties involved in the communication must exchange a key. However, the issue is the initial security of the key's transfer. The fact that the secret key can be detected and intercepted by eavesdroppers means that both the sender and the recipient are exposed [63]. The figure 3.4 demonstrates exactly that.

[5]Personal Information Number (PIN): is an identity-verification code. password-like

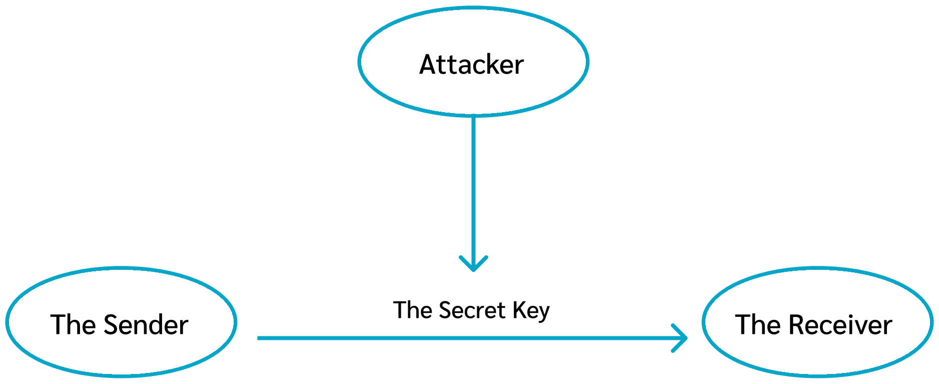


FIGURE 3.4: Secret key cryptography weakness [63]

Using Public key cryptography

As described previously regarding the use of Secret key cryptography, it is necessary for the communicating parties to trade a secret in order to communicate. As a result, eavesdroppers use a vulnerability in the communication privacy, which leads to compromised data and false authentication.
Using only public key cryptography is far more convenient. The only piece of information a user must remember is his private key. To verify the identity of thousands of entities, he will need to know (or have access to) a significant number of public keys. In this situation, the healthcare system for example should organize and safeguard the public keys for each member, which includes protecting the record of public keys against unauthorized access. It becomes simpler to maintain the keys (public key/private key) as the number of keys that must be kept secret reduces.
On the other hand, using public key technology only is not ideal for encrypting large data because the throughput of encryption and decryption is inversely proportional to the key length which is a disadvantage [63].

Using the combination of Public Key Cryptography and secret key cryptography

Despite the fact that public key cryptography can perform all of the same functions as secret key cryptography, the most well-known public key cryptography algorithms are generally time-consuming and slower, but they can be used in combination with secret key algorithms.
Network security based on public key cryptography is typically easier to configure, it is particularly beneficial when trying to authenticate the parties involved, and it may be used at the start of the communication. The remaining portions of the communication may then be encrypted using secret key cryptography by creating a temporary shared secret key [50, 27].
Assume, for example, that an assistant wants to communicate with a doctor. He uses His public key to encrypt a secret key, which is used to encrypt anything else he wants to transfer to him, encrypting the secret key using public key cryptography is not a huge performance hit because the secret key is significantly smaller than the message. To decrypt the secret key, the doctor uses his private key. Once he has the secret key, he can communicate with whoever sent him the message. Because of this mechanism, the doctor

now has no idea that the assistant is the sender of the message. To fix this issue digital Signature was introduced.

Digital Signature: Public key cryptography is necessary for digital Signature. To sign a message, the sender uses his private key, and to verify it, the receiver uses his public key. Valid digital signatures ensure that communication has not been compromised by the sender before it reaches its intended recipient [70]. The figure 3.5 explains digital Signature flow.

Overall, using the combination of public key cryptography and secret key cryptography

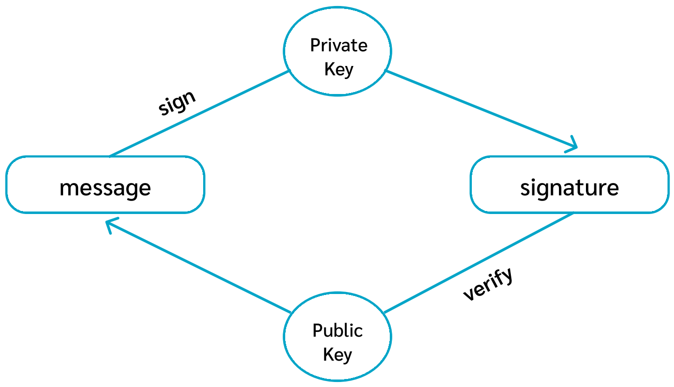


FIGURE 3.5: Digital Signature

for establishing secure communications in all phases is more legit and convenient with advantages that they provides, making it recommended for network security depending on the environment and several factors.

3.3.3 Secure Sockets Layer/Transport Layer Security

Data transported over a network is protected by Transport Layer Security (TLS) a more secure and robust alternative to Secure Socket Layer (SSL). The data sent between two nodes typically a user's web browser and a web app server is protected from inspection by attackers (or even Internet Service Providers) through this. Most website administrators and operators are obligated to use SSL/TLS in order to secure the flow of sensitive data, including passwords, account information, and other private personal info. Both protocols use the hybrid solution which is the combination of public key cryptography and secret key cryptography, it's a good trade-off between speed and security when sending sensitive data via a secure channel like this.

But there is a notice that should be mentioned which is that a secure transfer of data is ensured, preventing eavesdropping from happening. But the data on the end systems is not safe simply because these two protocols are implemented to secure communication for client-server environments [25].

SSL has been deprecated and replaced by TLS for several reasons which be explained next.

TLS vs SSL

The main differentiation between Secure Socket Layer and Transport Layer Security is that SSL uses the MD hash function [6] to generate a secret key and offers the fundamental security services of authentication and confidentiality. While the secret key is generated using a pseudo-random method [7] in TLS.
The table represents the other differences [68, 59].

SSL	TLS
uses a port to establish a direct connection	establishes an implicit connection using a protocol
SSL is slower and less trustworthy	TLS is incredibly trustworthy and updated. Less lag is achieved
SSL is less secure	TLS is a highly secure protocol
SSL is more complicated to configure	TLS is simple to configure
uses the message authentication code	uses Hashed Message Authentication Code protocol

TABLE 3.1: SSL and TLS comparison.

TLS workflow

TLS protocol consists of two parts [26]:

- **Handshake**: The handshake part is simply a key exchange. Starting by trading the public keys and in the end, the two parties involved agree on a set of secret (symmetrical) keys using RSA encryption in which a safe channel of communication is established.
- **Post-Handshake**: It is the part where the communication between the two parties is protected by encryption. using the set of keys generated at the conclusion of the handshake part and an approved encryption algorithm like AES.

The figure 3.6 illustrates TLS work-flow :

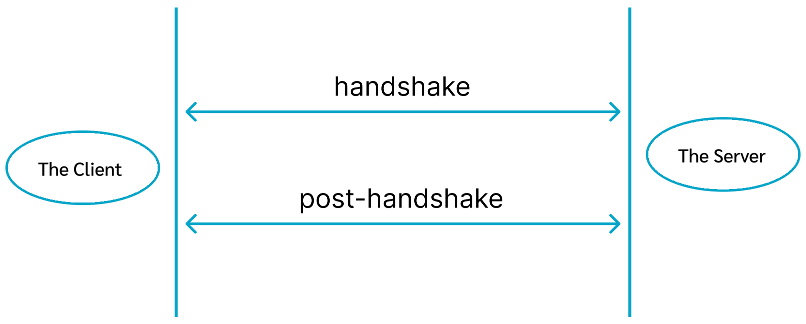


FIGURE 3.6: TLS work-flow [26]

[6]MD : It is a one-way cryptography hash function that returns a string of digits.

[7]pseudo-random method: An algorithm that generates random sequences of integers by applying mathematical formulas

3.3.4 End-to-End Encryption (E2EE)

The term "end-to-end encryption" refers to the use of cryptography in network communication between the end points. Secure systems that use end-to-end encryption ensure that users' data is protected while also guaranteeing the system's integrity and authenticity by using the combination of public key cryptography and secret key cryptography for establishing communications.
Encryption of data or communications occurs on the user device where they are made and sent and are protected until the data reaches its intended receiver. In this manner, the data is safeguarded while it transfers through the channel. An intended sender and recipient can read it only if they are in the same channel and the encrypted data can't even be read by the service provider or the server that hosts [37, 33].

E2EE workflow

The process begins with the sender obtaining the receiver's public key from the host server, then encrypting the message to be sent using AES encryption. In order for the receiver to decrypt the message, he needs the AES key, which represents the secret key. The sender then encrypts the secret key using the receiver's public key to protect it from eavesdroppers during transmission. To ensure authentication, he signs the encrypted secret key with his private key. After receiving the data, the recipient confirms the sender by verifying the signed encrypted secret key using the sender's public key. Then using his private key, he decrypted the encrypted secret key to use it for the AES decryption of the message. A notice that should be mentioned is that RSA algorithm is used for the encryption and the decryption using public and private keys of both parties. Figure 3.7 illustrates all the phases of the E2EE process.

3.3.5 TLS vs E2EE

TLS and E2EE are both encryption technologies that apply a combination of public key and secret key cryptography to prevent unwanted parties from accessing data.
TLS is utilized exclusively when communicating between a user and a server, which is the main difference. As a result, data sent to and received from a server remain safe throughout transport, and data saved on the server remain in their unencrypted form. The server must have access to the user's data in order for a web application to function properly. However, from a privacy standpoint, this is not always appropriate. If users wish to transfer data to one another, for instance, they may not want the service provider or the host server to be able to read this data. Therefore, E2EE is recommended in such a circumstance. E2EE guarantees that only the two individuals communicating can view data. No one has to rely on the service to manage their data if it is created appropriately. E2EE gives users complete control over who can access their data, allowing them to maintain secure communications.
There are costs associated with implementing a TLS certificate, which is another element to consider. Cost is determined by the number of required domains and subdomains. Additionally, it may be based on the verification of a person's

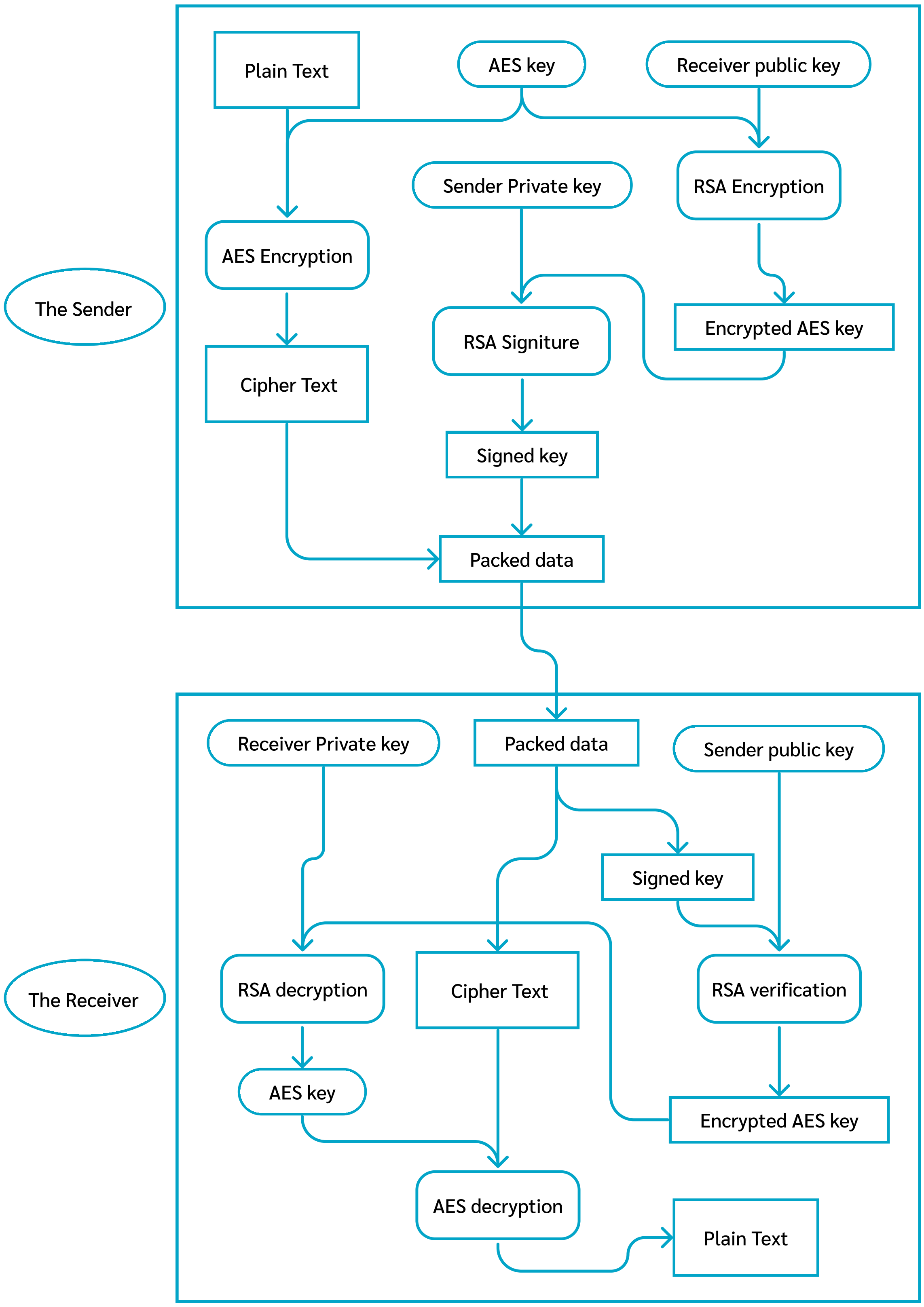


FIGURE 3.7: End-to-End Encryption work-flow [37, 33]

identification. In contrast to E2EE, which can be constructed from scratch using several protocols such as RSA and AES or the Diffie-Hellman protocol (DH)[8].

3.3.6 Firewalls

Firewalls safeguard private networks from illegal access. To communicate with or use the internet, all data must travel via the firewall, which may identify and reject any security risks [96].
The Health Insurance Portability and Accountability Act (HIPAA) requires security controls to guarantee confidentiality and compliance. Implementing HIPAA firewall controls is essential for protecting against costly healthcare data breaches.
Configuring firewall rules based on individual needs allows for different levels of network access to be granted to different people which is fully cooperating with the healthcare system [4].

3.4 Storage Privacy

Storage privacy refers to the capability of storing data without allowing anyone else aside from the person that stored the data, referred to as the data owner here to read it or modify it. The prevention of unauthorized parties accessing the stored data is a significant problem for the implementation of private storage. Physical access control may be helpful if the data owner maintains their data locally, but it is insufficient if the computer equipment is connected to a network because a hacker for example may be able to access the data remotely. Physical access control is not even an option if the data owner saves their data on the cloud. Therefore, technology countermeasures are required.

3.4.1 Operating system controls

The most widely described of ensuring some kind of storage privacy are user authentication and access control lists administered by the operating system. However, an attacker can access the data if he has physical access to the machine or is able to get past the operating system protections.

3.4.2 Local encrypted storage

One simple solution is to save the data locally in encrypted form. There are two types of encryption for storage: Full disk encryption (FDE) and File-based encryption (FBE).

Full disk encryption (FDE)

Uses symmetric key cryptography for Automatic encryption when data is written to or read from a disk, but no files are encrypted. The entire disk is encrypted with the same secret key, which is instantly decrypted when the correct user credentials are entered[48].

[8]DH : is a protocol for transmitting cryptography keys via open channels safely. Keys are collectively derived, not really traded. It bears the names of the two men who invented it, Whitfield Diffie and Martin Hellman.

File-based encryption (FBE)

Uses symmetric key cryptography to encrypt specific files or even small groups of files on a disk. A disk/drive can contain several files, each of which can be encrypted using a separate secret key, unlike full-disk encryption.

3.4.3 Steganographic storage

The art and science of writing hidden messages in a way that only the intended recipient is aware of their presence are known as steganography.
A very high level of privacy is provided to the user by this approach since the user data are steganographically integrated into cover data.
A steganographic file system delivers a file to a user who is aware of its detail like the name and password, but an intruder who is unaware of this detail cannot determine whether the file is present, even if they have full access to all hardware and software[71].

3.4.4 Secure remote storage

In remote contexts, private storage can also be obtained using encrypted and steganographic storage. Steganographic storage is actually more desirable in a cloud environment because the hardware and software are not in the user's control.
However, in this scenario, if any storage privacy is to be achieved, the keys used in the encryption/embedding and decrypt/recovery processes must remain in the power of the user and must be performed locally.

3.5 Conclusion

Privacy is a fundamental human right. Healthcare providers must safeguard the confidentiality and security of patient information in order to create a trustworthy environment for patients. As data protection methods grow, data privacy in healthcare is continuously in flux. As a result of the fast expansion of digitization, data breaches and cyber-attacks have considerably grown, making the use of patient information riskier than ever before. As a result, healthcare services require the highest level of data privacy.

Chapter 4

The conception of an intranet medical practice management software

Increasing the efficiency of healthcare requires effective management. There is a pressing need to do more with fewer resources in the healthcare industry due to rising demand and stagnant or declining funding [52]. Information is essential at every level of the healthcare system. It is essential for the management of patients and clients, for the management of health units, and for the planning and management of the health system. Aside from policymakers and managers, this means that healthcare providers such as doctors and health technicians must also use information in decision-making [52]. Given the available knowledge and technologies, a thoughtful understanding of the healthcare system is all that is required to create efficient Practice Management Software, which is a one of various health information systems that reflects medical practices while adhering to medical policies.

4.1 Practice Management Software (PMS) requirements

There are no restrictions on the number of features that can be included in a Practice Management Software (PMS) as long as it helps healthcare facilities to improve the quality of services. But for the system to be acceptable it must contain the following attributes:

Consistency: It is necessary for the system to maintain its consistency in order to supply the users with the services they require.

Convenience Convenience in using the system to carry out operations is a necessary requirement to avail the services.

Availability One of the most important requirements is to provide a system that is available and operational at all times.

Usability The system must be user-friendly and have an intuitive interface.

Security The system must provide security mechanisms to safeguard personal information and medical records. Any breach of information security will result in legal action.

Reliability The healthcare management system must be reliable in delivering the functionalities.

Reflecting The system must be able to meet the demands of all healthcare providers, and it must incorporate medical duties that can be integrated digitally.

4.1.1 Requirement finding

In order to design and develop the proposed system, it is necessary to know how the system works. There are a variety of techniques and methodologies that are used to investigate and identify system requirements. The most common approach is to conduct a thorough investigation of the problem and its application. This method of requirements analysis requires looking in journals, publications (including newspaper articles), documents, and reports of existing, similar, or proposed systems. Information about current systems may also be obtained by speaking with users of the current system who are also potential users of the proposed system.

4.1.2 Software functionalities

A set of functionalities has been chosen and constructed carefully to be a part of our system. Some of these functions are not required but favorable by doctors.

Control access

Securing software requires tight control over who has access to what data and how it is used. The RBAC model includes all of the features required to completely manage who has access to what information.

Administrative management

Any organization's administrative management is critical. It is only by providing a means of administration that other features will have any real value. Administrative tasks can be accomplished using the RBAC model thanks to the relationship and structure of roles and permissions. A specific member can administer the site if they have the necessary permissions.

Appointment Scheduling and Waiting list management

Appointment scheduling and waiting room management are fundamental requirements. They are generally applied in outpatient clinics and other healthcare services.
Members can use the software to schedule appointments for specific patients on specific dates and add them to the waiting list when they show up for their appointments.
The software also provides methods to manage waiting rooms efficiently, which are:

- Each member with sufficient permission has a waiting queue.
- The owner of a queue is responsible for hosting patients and providing medical services such as diagnosis or appointments.
- The queue owner is not required to be physically present in the waiting room because the software enables collaboration with other members on the network. Members can assist in managing the queue in the waiting room. They have the

ability to add or remove patients from the queue, register new patients, and collect medical information such as weight or blood type.

Electronic record management

It's an optional feature for practice management software to offer an Electronic Medical Record (EMR), but healthcare facilities must have one. EMR is a collection of patients' medical records; it must be carefully secured and protected as it contains sensitive data that should only be accessible to the patient and the treating doctor.

Medical Session and prescriptions generation

A medical session is the meeting between a doctor and a patient. The doctor may give the patient a notice or a prescription in form of a paper. The software help to generate the prescription/notice and send it to the printer.

Member communication and collaboration

Medical care can be provided by a variety of parties. This means that a patient can be transferred to another medical facility in order to receive treatment. To improve the standard of care, a facility members must be able to communicate and collaborate.

4.1.3 Benefits

The practice management system enhances the safety, effectiveness, and efficiency of healthcare services. It offers solutions for medical problems and help healthcare facilities with the following :

- Permit healthcare facilities to fully administer the software and facilitate medical staff collaboration.
- Providing a method to manage appointments digitally will enhance service quality.
- Establish a user-friendly application to distribute a flexible environment for healthcare workers.
- This aids in the secure storage of medical records, which may lead to a reduction in the time required for medical diagnosis and the prevention of medical errors.
- Establish a trustworthy system that is outfitted with the most up-to-date privacy safeguards to ensure the safety of patient information.

4.2 Similar Implementations

The widespread implementation of HIS has been hampered by a lack of generalizable knowledge regarding the HIS types and implementation strategies that will improve care and reduce costs for particular health organizations [82]. Some implementations are good enough for a specific use case, but cannot be compared to a standard HIS practice management software that can manage a small clinic to a large hospital. Three software implementations have been chosen because of their resemblance to ours.

4.2.1 openEMR

OpenEMR is a free and open-source application for managing electronic health records and medical practices. It provides fully integrated electronic health records, practice management, scheduling, electronic billing, internationalization. It is compatible with variety of operating systems. The supported features are as following [67]:

- Patient Scheduling
- Electronic Medical Records
- Prescriptions
- Medical Billing
- Clinical Decision Rules
- Patient Portal
- Reports

This software provides numerous advantageous features for healthcare facilities. With many features comes the difficulty of organizing the requirements into a meaningful user interface so that they are accessible and simple for clients to use. This software failed to provide that because its user interface lacked organization and accessibility, making it difficult for clients to access the software's bare minimum requirements. Its complex, component-laden user interface makes the learning curve steep and costly in terms of time, and makes simple tasks such as adding an appointment or accessing patient information more time-consuming.

4.2.2 ERPNext

ERPNext is a comprehensive business management tool that centrally record all business transactions.It is build with modular architecture and offers modules for many business models. ERPNext Healthcare which is a module helps healthcare facilities manage practices by offering the following [39] :

- Appointment scheduling.
- Appointment Analysis
- Vitals capturing.
- Medication management.
- Patients' Medical history.
- Allows multiple practitioners (medical staff).

This software is well-designed and ideal for business administration. Its modular design enables the existence of the ERPNext Healthcare module, but its modularity may be a weakness due to its lack of focus on a single business model. The ERPNext Healthcare module provides a number of useful features, but it does not fully integrate with medical data privacy policies because it does not grant explicit access to patient records to the responsible member. And while it may function on an intranet, it was designed for the internet, and improper implementation on an intranet could result in vulnerabilities and data leaks.

4.2.3 Medintux

Medintux is a medical record management software suite written for the French environment whose main objective is to adapt to its users. It is therefore built with the constant aim of assisting and relieving the user as much as possible in his work [60]. The software works in a network and multi-user environment and supports the following features:

- consultations.
- meetings.
- prescriptions.
- accounting.
- visualisation of bio-metric curves.
- statistics.

The lack of features makes it hard for this software to fit in PMS Category. It is dated software that does not meet most of the fundamental requirements necessary to function as a PMS in the modern era.

4.3 Software architectural

The software is based on the client-server architecture, which is a distributed application [1] structure that divides tasks or workloads between service providers (also known as servers) and consumers of that service (also known as clients) [66]. Clients and servers communicate over a computer network using separated hardware. A server host operates on one or more computers and shares its resources and services with clients. The client initiates a communication session with the server when it requests resources or services from it. Depending on the implementation, clients may also share resources with the server.

4.3.1 Client software

The Client software is a software that runs on the staff computers of a healthcare facility. It is executable software compatible with multiple operating systems, including Windows and Linux. Users can register their local information (e.g., name, password, avatar) locally, as the software does not require a running server to function. It enables

[1]Distributed application: Refers to an application that operates in a distributed environment. In a distributed system, all components are located on separate networked computers and exchange messages with one another using a computer network.

users to join a healthcare server, which is a live server on the network, by being a member of the server or by using an invitation key. Users can join multiple servers but not at the same time.

Graphical User Interface

A Graphical User Interface (GUI) is a type of computer human interface that enables users to interact with electronic devices via graphical icons and audible cues [43]. The client software provides a GUI that is intended to aid users in using and manipulating the software.

Connect to a server

A user's ability to connect to a server depends on whether or not they have previously joined the server.

First-time joining: An invitation key is required to be able to join a healthcare server for the first time. Invitation keys are generated by server members with the permissions to invite. A selected role can be assigned automatically using the invitation key. This operation only happen once.

Previously joined: If a user is already a member of the server, he can simply select the healthcare server from the clinics list to connect to the server.

Local data

Non-clinical data is not necessary and is overwhelming for the server to store. But may be needed for the software to perform as intended. That is why the client software stores some data locally on the user's machine. Although this data is stored locally, it may still be required by the server software.

4.3.2 Server software

In Health Information System, the server software is the core source of information. All medical records are stored on the server. In addition, it's in charge of ensuring that only authorized individuals have access to sensitive data. It also acts as a middleman for various forms of user communication. Its provide multiple services to ensure the software functionalities, which are:

API service

An Application Programming Interface (API) is a type of software interface that allows client and server software to communicate. The client/server software exposes the API by adhering to the API specification, a communication standard that enables meaningful interaction. There is a wide range of data transfer options defined in the API specification documentation and handled by the API service.

Authorization service

The RBAC model dictates how users are granted access to resources. The authorization service determines which resources are accesible to users based on their roles. Figure 4.1 illustrates how the service works.

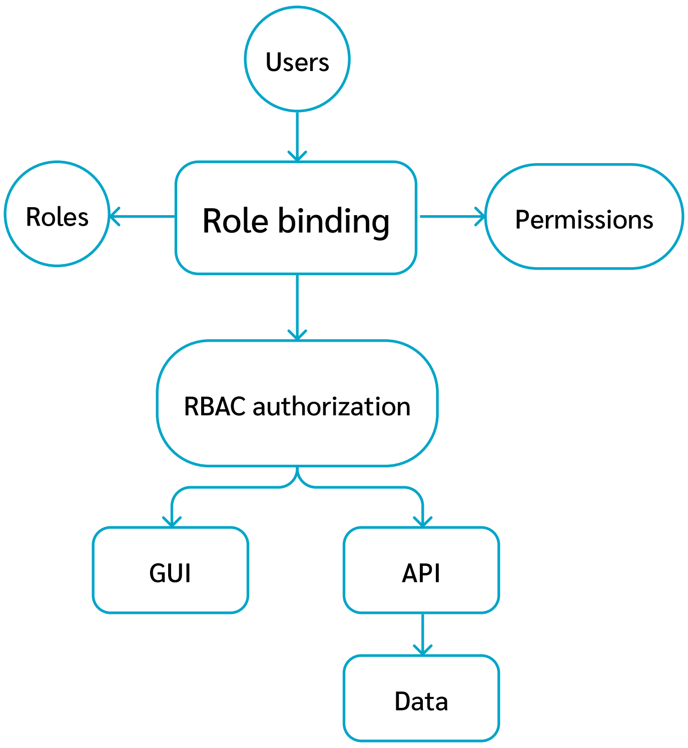


FIGURE 4.1: Authorization service using RBAC binding.

The resources are divided into two categories:

Graphical User Interface (GUI) The GUI is rendered based on users's permissions. Certain graphical elements, such as the context menu, are role-based, meaning that their appearance may vary based on the user's permissions.

Application Programming Interface (API) The API is the service that provides the correct data and information to users. RBAC is used to grant users access to particular pieces of information and prevent unauthorized access.

Authentication/identification service

An Authentication/identification service is used by the server to handle user identification and authentication. Its the first line of defense against any potential threats. It ensures the authentication and identity of users so that only authenticated users can be authorized. Several methodologies are used for that:

Invitation key Is a form of authentication key that is given to new members by an existing member who is authorized to do so in order to allow new members to join the server.

HWID Is a unique hardware identification consisting of a string of characters, is used to identify and authenticate users. When a user joins the server for the first time, the server saves a unique identifier for the user's hardware (computer), which is then used to identify users and assign them to the correct member identity whenever they attempt to rejoin from the same device.

Access key Is a passcode generated by the software to help members retrieve access in the event of changing hardware. The access key should be kept safe by the user as it's the only way to regain access to the server with the same member identity.

E2EE Is a security protocol that provides communication authentication and encryption. Its allow the software to authenticate members during communication.

4.3.3 client-server network

A client-server network is the communication channel that enables clients to access the server's services and resources over a computer network [66]. A Local Area Network (LAN) is used to interconnect clients and server on the same network. The central management of information and data is the major benefit of the client-server network.

4.3.4 server-server network(P2P)

A Server-Server network is a Peer-To-Peer (P2P) communication that uses Internet to interconnect servers [51]. servers can expose a portion of their API to be used by other servers. This kind of communication is optional and requires a Public IP address. Servers are protected by a firewall to only allow trusted IP addresses, and the data that travel trough the Internet is encrypted using E2EE.

4.4 System Design and workflow

It is necessary to conduct a comprehensive methodological analysis of the workflow before constructing the software. This involves describing the processes of operations through structured graphical and textual representations, which will ultimately assist in the construction of the software. There will be a display of the system's activity diagrams, use case diagrams,data class diagram, and ER diagram as well as a brief description of each one.

4.4.1 Activity diagram

The activity diagram is a graphical representations of the workflow. By visualising activities and actions in step-wise manner. Activity diagrams are used to represent the system activities depending on actions and user choices.

Join a Clinic server

This activity represents the actions of a client joining a clinic server. As depicted in Figure 4.2, joining a server necessitates either an invitation key issued by an authorized server member or a secret key. A secret key is only used if the client has previously attended the clinic.

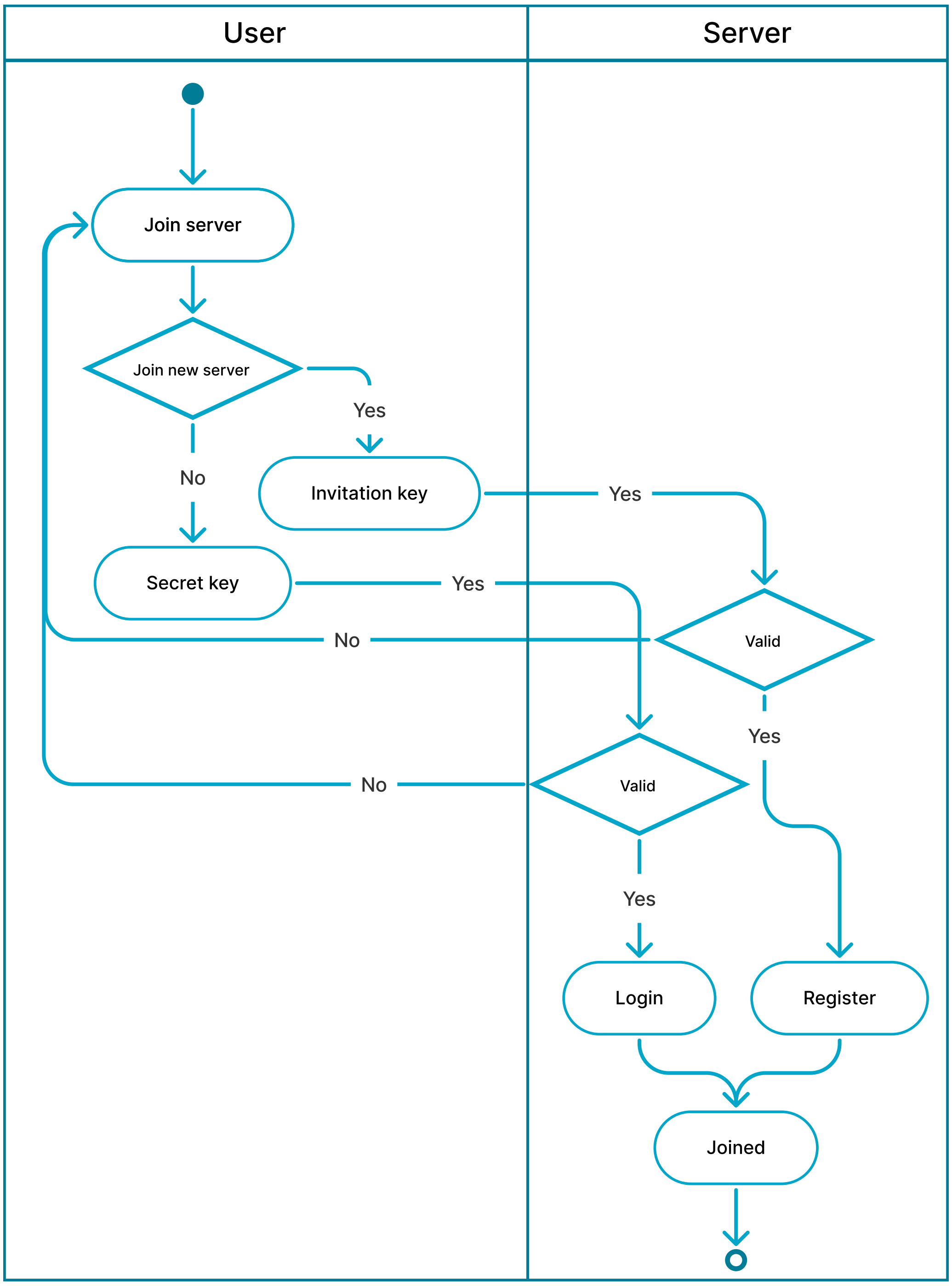


FIGURE 4.2: The activity of joining a clinic.

Assign appointments to the Queue

To enable the concept of assistance, the queue element permits multiple actors to manage the same activity of assigning to the queue. Patients can be assigned to the queue by both the queue owner, who is a member with queue management permission, and members with an assistant role to the owner. This allows doctors to have assistance in a clinic waiting room. Figure 4.3 depicts the actions that make this behavior possible.

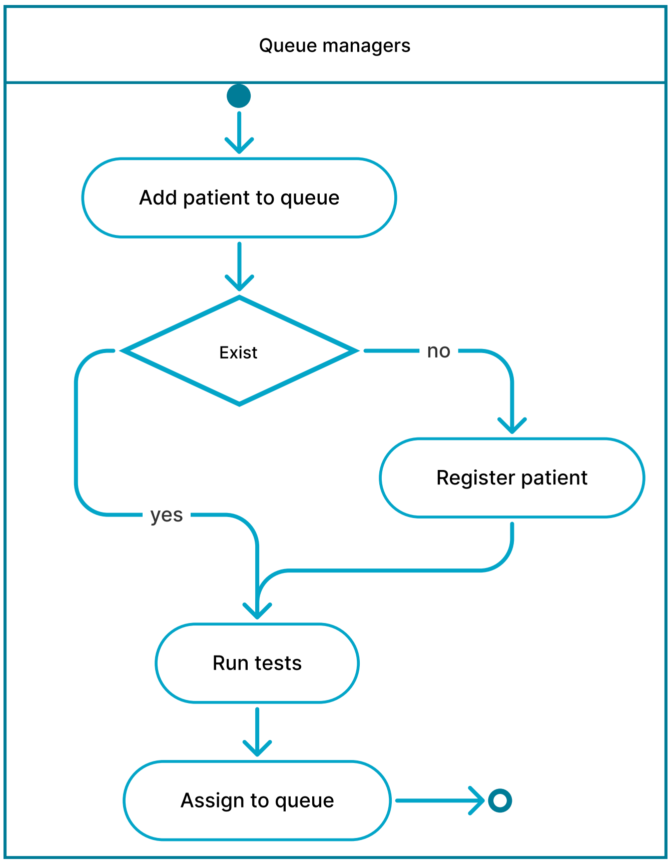


FIGURE 4.3: The activity of adding appointments to queue.

Start a medical session

When a healthcare professional and patient confer in a clinical setting, they are engaged in a medical session. Once the doctor calls for the next patient in line, all queue (waiting room) managers will be notified. The manager or owner of the queue must then personally notify the corresponding patient. As soon as the patient arrives, the session can begin. Figure 4.4 illustrates medical session activity.

4.4.2 Use case diagram

In the process of system workflow analysis, use case is a methodology that is utilized to determine, clarify, and organize system requirements. As the system is role-based the actors are role owners and their abilities can vary depending on their roles and permissions. For that, a set of roles has been chosen to represent a simple healthcare facility system. Actors are members with the following roles:

Admin Members with this role are allowed to do administrative tasks.

Doctor Members with this role can have a waiting queue with the ability to fully control the queue and initiate a medical session.

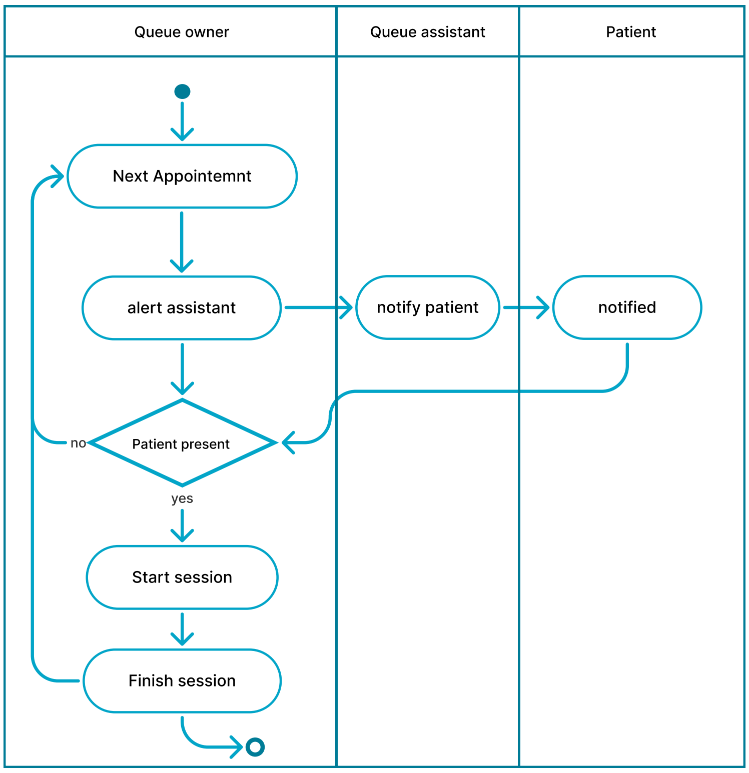


FIGURE 4.4: Activity of a medical session activity.

Assistant This role is linked to the "Doctor" role. Members with this role are allowed to control a waiting queue and book appointments.

Appointment scheduling management

Figure 4.5 depicts appointment scheduling procedures. A member who owns the appointment list can either schedule an appointment, assign a scheduled appointment to the waiting queue, or cancel a scheduled appointment. A member with an assist role to the owner's role is also capable of performing the same operations.

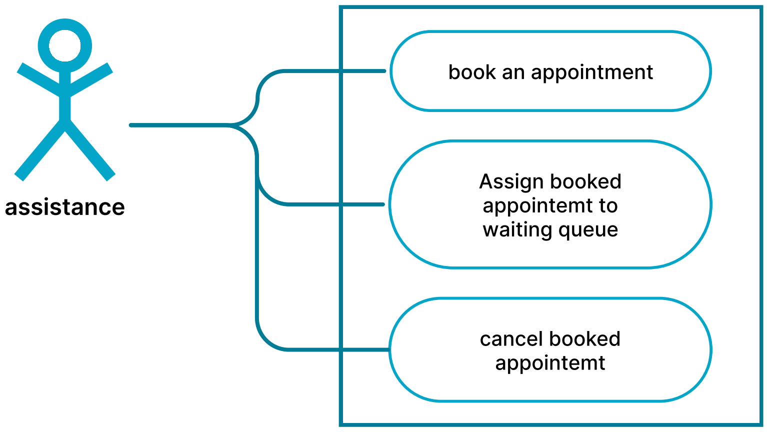


FIGURE 4.5: Appointment scheduling use case diagram

Queue management

A member who have a "doctor" role can either add an appointment or select one from the queue. A member with the "assistant" role can assist by adding appointments to the waiting list and conducting boimetric screening to collect information such as weight and blood type. The doctor needs to notify the assistant before starting the medical session. The assistant in the other hand needs to physically notify the patient to be able to start the medical session. The doctor can then write a diagnosis, or fill a prescription list when the medical session begins. A printed copy of the prescription is then provided to the patient at the end of the session. Figure 4.6 illustrates the operations of a doctor on a queue. Figure 4.7 represent the operations of a assistant on a queue management operations.

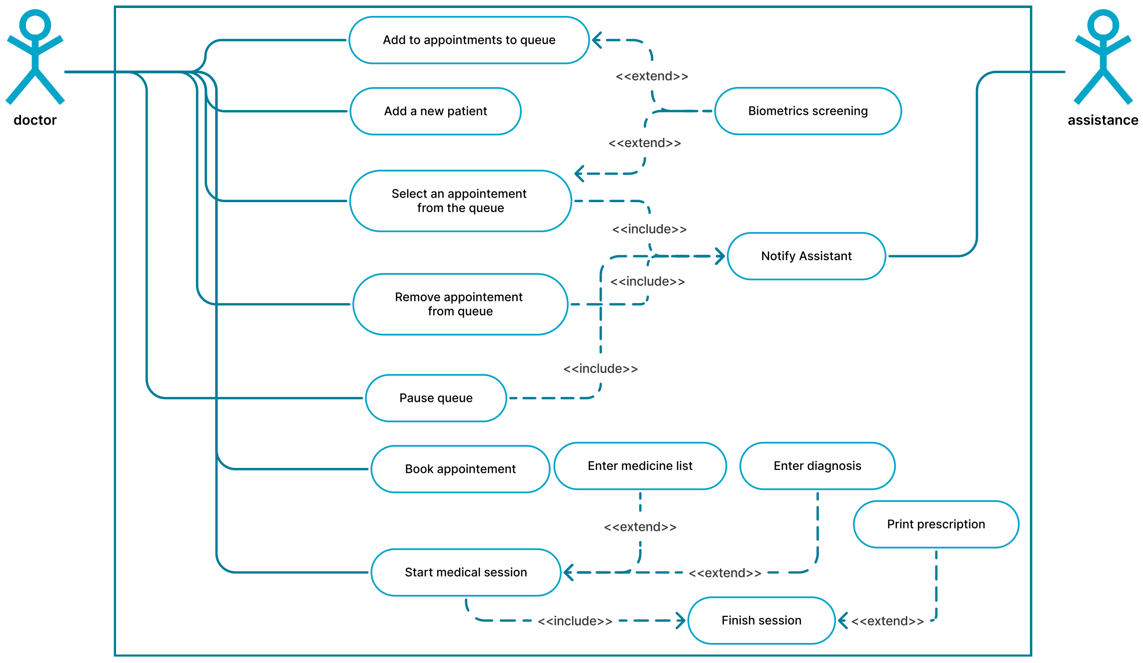


FIGURE 4.6: Doctor queue management use-case diagram.

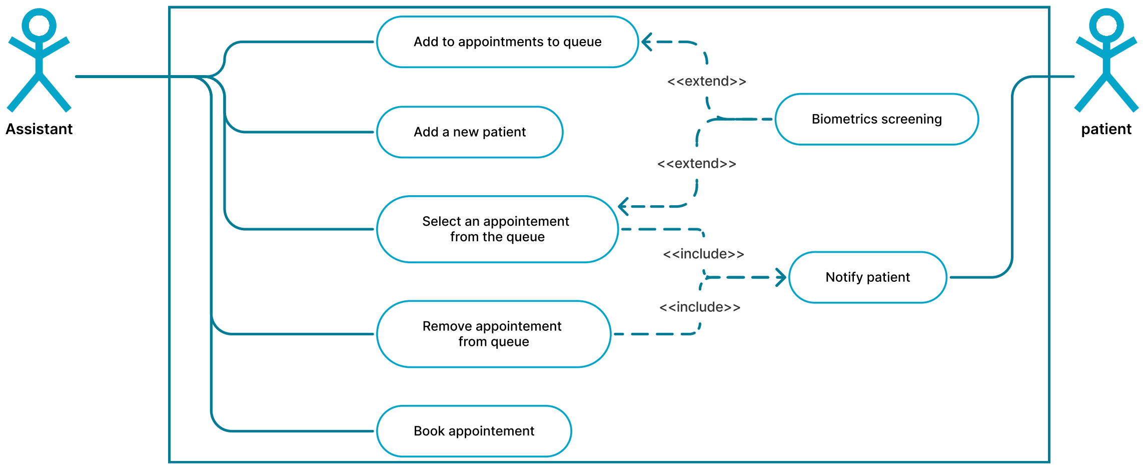


FIGURE 4.7: Assistant queue management use-case diagram.

administrative management

Figure 4.8 shows administrative management in action. It is possible for a member with sufficient permission to manage:

Roles: Create, delete or edit a role name and permissions .

Members: Manage members by adding removing or managing member roles.

Clinic: Contain actions from clinic general information to security and privacy settings.

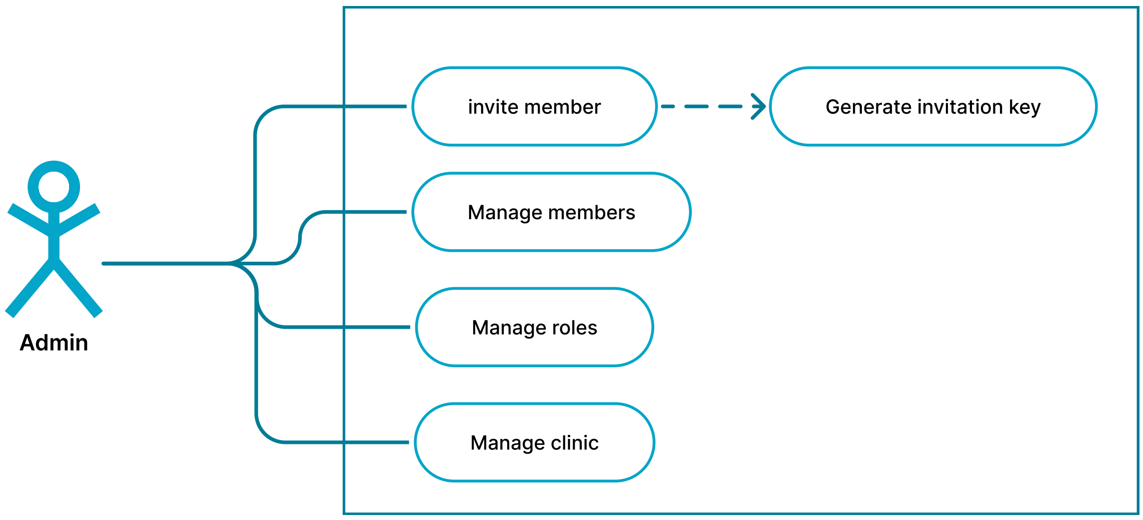


FIGURE 4.8: Administrative management use case diagram

4.4.3 Data class diagram

Data class diagram is used to describe the structure of the system by illustrating system classes with their attributes. Figure 4.9 represents the system data classes.

4.4.4 Entity Relationship Diagram

ERD is used to describe the relationship between data classes of the system database. Figure 4.10 represents the system ER diagram. There are three categories of classes, each with its own service or instance that is in charge of running it. Each category of classes is linked to the other ones via identifiers and other references, allowing them to work together in parallel. Figure 4.11 illustrates the system class categories.

Local data classes

Data is locally stored by each client instance. Local data classes are specific to users, but they are linked to other class categories on the system and can be used in conjunction with them.

Administrative Core data classes

Administrative classes fall under this category. These classes are utilized by the authentication and authorization services. They are an application of the RBAC model.

Management data classes

this category contain all the classes that are related to medical data.

4.5 Conclusion

Although the concept of a practice management system is not new, the current implementations lacks generalization and market comprehension. With a thorough understanding of healthcare information systems and requirements, as well as a well-defined structure, it is possible to develop software that is inexpensive, user-friendly, and effective at delivering the intended functionality.

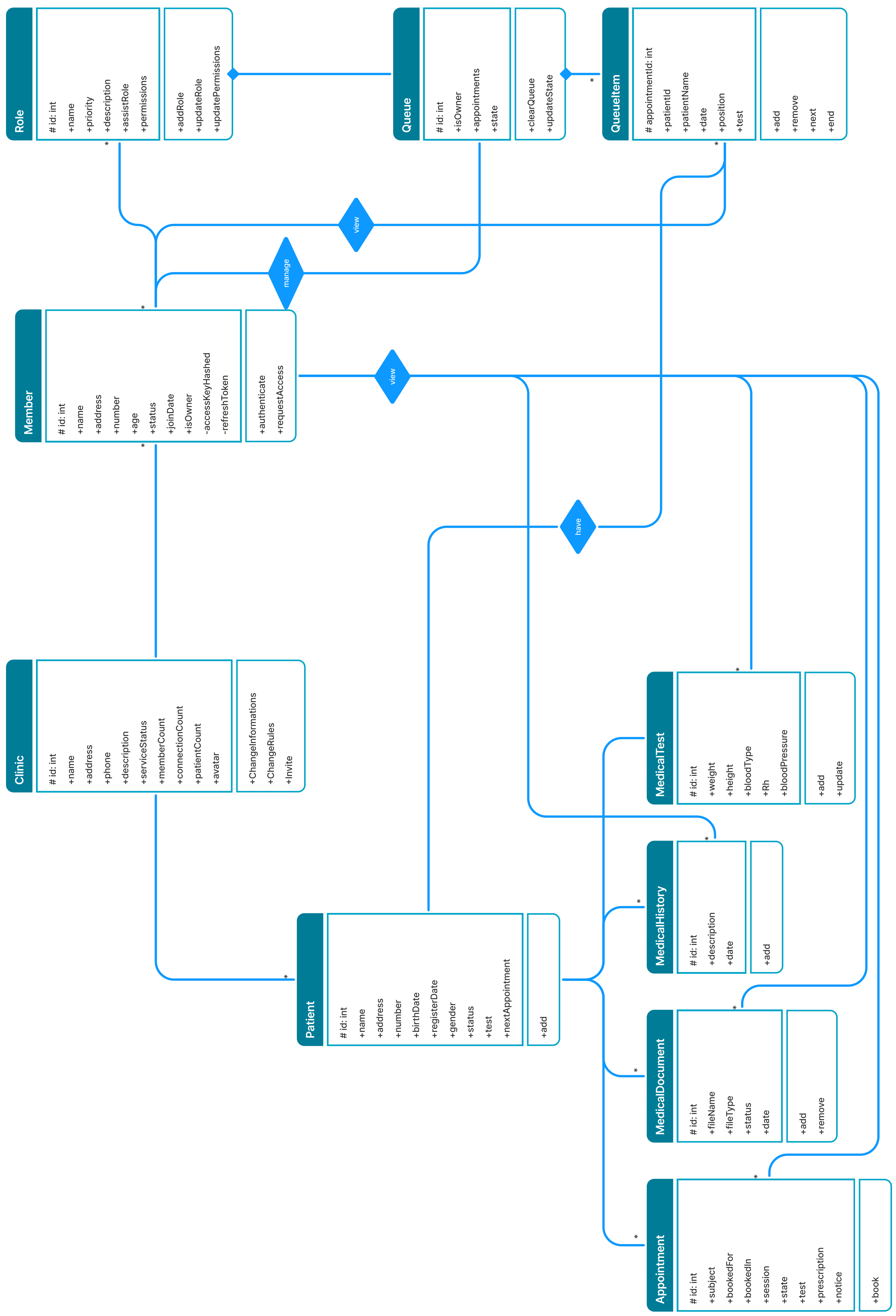


FIGURE 4.9: Data class diagram.

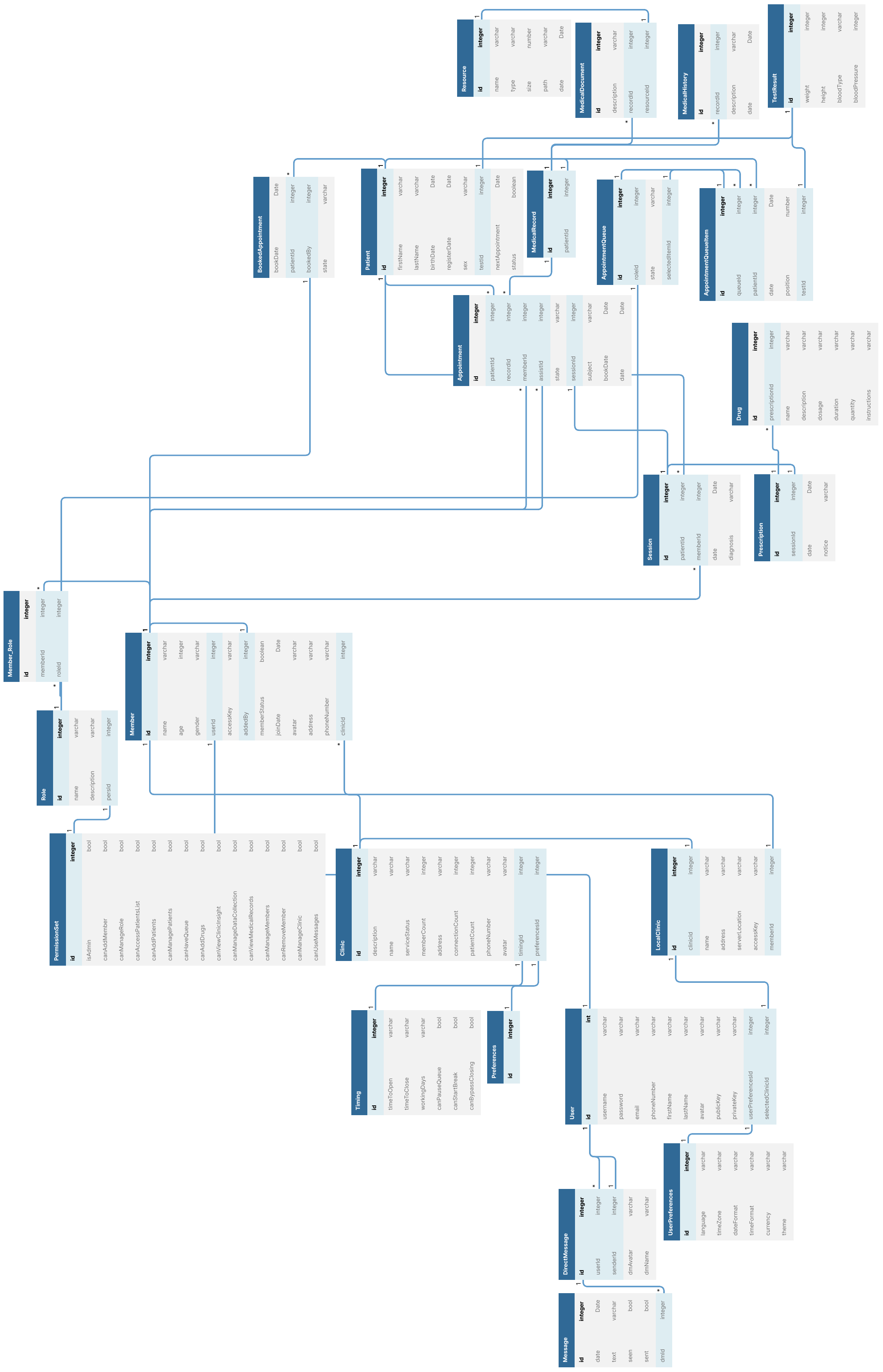


FIGURE 4.10: ER diagram
]]>

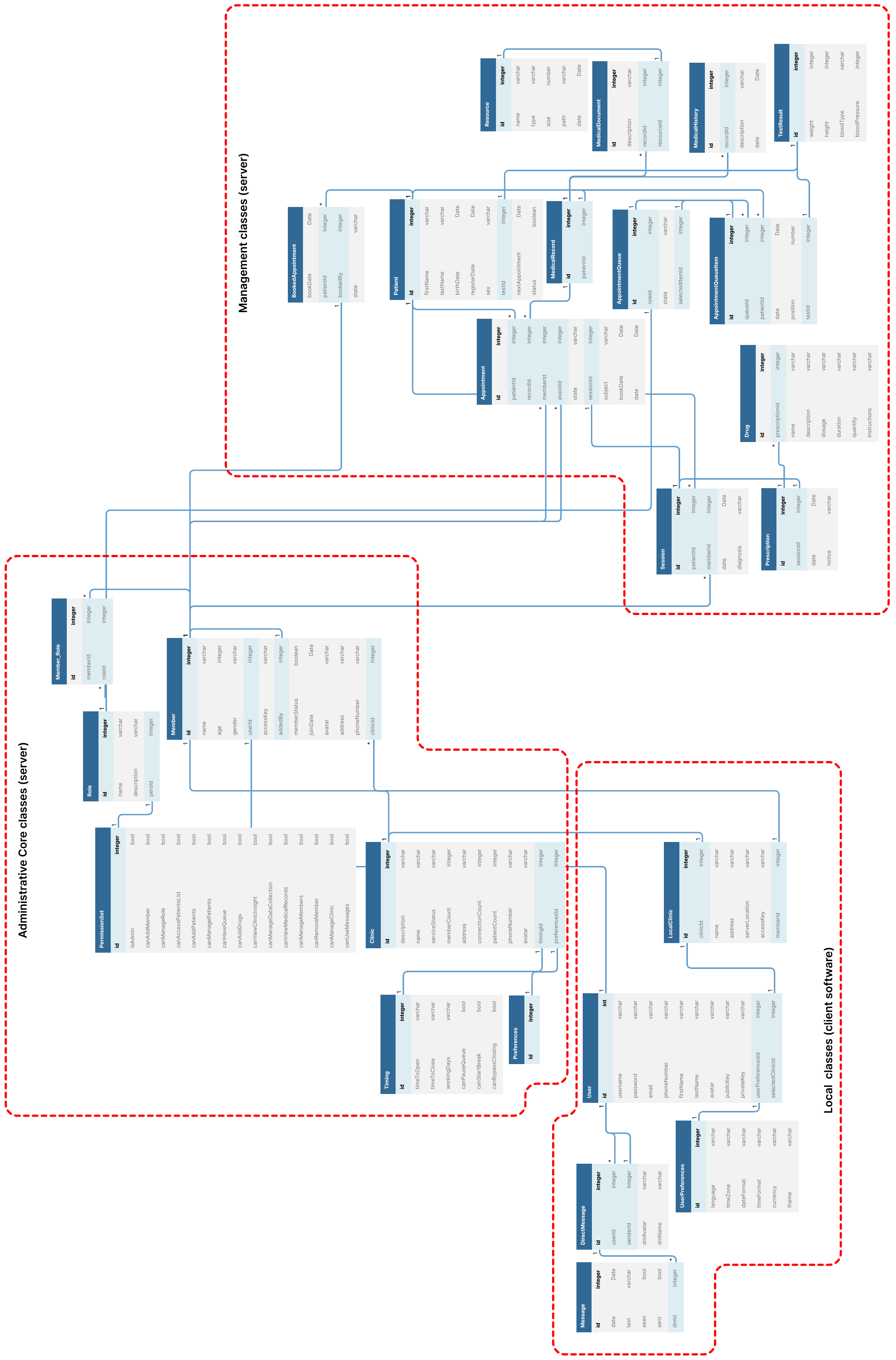


FIGURE 4.11: ER classes separation in the system

Chapter 5

Implementation of a medical practice management software

5.1 Web technology

5.1.1 HTTP

Programs communicate over the World Wide Web using the Hypertext Transfer Protocol (HTTP). Although HTTP has various uses, two-way communication between web browsers and web servers is the one for which it is most well-known. Web servers are the source of web content. These servers maintain the data of the Internet and make it available to clients on request. Servers respond to HTTP requests from clients by returning the requested data in HTTP responses.
HTTP methods are the various request commands that HTTP offers. A method exists for each request message. The technique informs the server what action to take (fetch a page, upload and delete a file ,etc.). The common HTTP methods are: (GET, PUT, DELETE, POST, HEAD)[11].

5.1.2 Web socket

WebSocket is a full-duplex protocol that can be utilized in the same client-server setup. This protocol maintains a persistent connection between the client and server until it is explicitly closed (client or server). When a client or server closes a connection, it is closed at both ends[93].

5.1.3 WebRTC

Web Real-Time Communication (WebRTC) is a technology that allows websites and web apps to record and stream audio and video, as well as transmit any kind of data directly between browsers. WebRTC is a collection of standards that enable peer-to-peer data sharing and teleconferencing without the need for additional software or plug-ins on the user's end. As a collection of APIs and protocols, WebRTC is able to accomplish this goal[58].

5.2 Client software

The client software is a software that's runs on medical staff's computers. Each user of the client software is considered as a member of the clinic if he is able to join a clinic server. This software is fully dependent on a server software to maintain its functionalities. There are various tools, technologies and language that goes in the construction of such a software.

5.2.1 Languages

HTML

HTML is an acronym that stands for Hypertext Markup Language. Programming languages are distinct from markup languages. While markup languages enable us control how items are presented on a webpage, programming languages allow us to change data. The building components of an application page are HTML elements. Elements are expressed by their opening and closing tags and can be nested inside of other elements. A page can be divided into sections, headings, and other content blocks using elements[41].

SCSS

The application styles are defined using CSS. It specifies a template that is authored in a markup language and its appearance and formatting. Typically, it works in conjunction with HTML to change how web pages and user interfaces look and feel.
The superset of CSS is called Syntactically Awesome Style Sheet. The more advanced form of CSS is SCSS. It is frequently referred to as Sassy CSS because of its powerful features. All of the CSS features are included in SCSS, but it also has additional features that CSS does not have like variables allowing shorter and dynamic code. SASS adds the @import capability, which makes it easy to import other customized SCSS files, and allows coding styles with nested syntax[80].

TypeScript

The JavaScript development team launched JavaScript as a client-side programming language when it was created. However, as more and more people started using it, programmers began to understand that it could also be utilized for different purposes and platforms like mobile development for example. But as JavaScript expanded, the code grew heavier and became more complicated. As a result, JavaScript was unable to even satisfy the criteria for an Object-Oriented Programming language and it has some issues that can occur on the runtime which make it harder for developers to handle errors and solve bugs in the application. For these reasons, Typescript was introduced.
TypeScript is a strongly typed, object-oriented, compiled language. It was designed by Microsoft. TypeScript is both a language and a set of tools. It is basically a typed superset of JavaScript compiled to JavaScript. In other words, TypeScript is JavaScript plus some additional features.
The project application code was written with typescript to prevent unwanted bugs and errors that can occur during the runtime by using data types like strings, numbers, or

objects. The functions' proprieties are checked by declaring interfaces which represent the shape of the function itself.
During the development of applications, typescript helps solving problems by compiling code on every saved file instead of checking the app runtime error stack. Static/strong typing is supported by TypeScript. This implies that type accuracy can be verified at the compiling stage. JavaScript does not support this feature[44].

5.2.2 Tools

Visual Studio Code

The application code was written with help of visual studio code, it is free, open-source, and cross-platform. With its built-in debugger, Visual Studio Code makes development flow more productive and keeps the code and the debugger in the same view, in contrast to many other code editors. This greatly facilitates and speeds up the issue tracking and code run-through processes.
VS Code supports extensions with new ones coming seemingly every single day, it can serve many purposes and goals like UI themes, programming languages support, debugging, GitHub source control, and more cool features. Making the development process easier and more interactive.
Another feature of VS code is Intelli-Sense, if any part of the code is missing, it can find it. Additionally, common variable definitions and syntaxes are created automatically. Ex: If a user forgets to declare a variable, they are using in their application yet IntelliSense will do it for them.
Terminal support is implemented on vs code, In-built terminal or console support allows users to avoid switching between two screens for the same task when they frequently need to start from the directory's root in order to perform it[91].
The project application workspace contains the API Server implementation and the client application, with help of vs code. It is possible to open numerous projects, each containing numerous files and folders. There may or may not be connections between these undertakings/folders.

Github

Developers can collaborate and manage versions of their code online with GitHub. It is used to keep track of all modifications made to a project's source code and store those changes. Providing tools for handling potentially contradictory modifications from many developers enables them to work on a project more effectively. Code can be modified, adjusted, and improved using GitHub's public repositories[34].
The Project application consists of two main repositories which are the API server code and the client app code making the organization and management easier with help of GitHub repositories by cloning to VS Code ,committing changes and syncing to the head bunch of the project repo.

Chrome Developer Tools

Google Chrome is a free web browser created by Google that is used to visit websites on the internet. It is the most widely used web browser worldwide. A cross-platform

browser, Google Chrome supports a variety of computers, smartphones, and operating systems. The Chrome browser comes with a complete set of developer tools called Chrome Developer Tools. With the use of these technologies, building better websites is quicker the ever, and editing them in real-time.
The client application design is inspired by web-based applications, that is the reason chrome is used for implementing style and testing with features that Chrome Developer Tools contains like[16]:

- **Element tab:** Displays HTML/CSS code that was used to create the page.
- **Console tab:** It informs the developers of any interactive components present on a page. They can create JavaScript interactions with web pages in Console, which then display it to show that the JS code was executed. It is also used to see any errors or issues from the app or the server's requests which is a handy feature.
- **Network tab:** Provides statistics on data transmission, including initiator and time to load data, as well as all the files that are loading in a given URL.
- **Application tab:** displays the contents of the browser's storage, including local storage and in-browser databases like Web SQL.

5.2.3 Front-end design

Figma

Figma is a tool for designing and implementing cloud-based digital apps. Users can collaborate on projects and do business virtually anywhere due to its design. It has numerous features including[31]:

- **Collaboration:** A method must exist for individuals to cooperate, present their work, and receive feedback. Figma possesses these features and more.
- **Real-time project updates:** In a UI/UX design process, the complete team must utilize more than one third-party solution to exchange project updates and design mockups. Multiple file transfers are required to keep each team member informed of the project's status. The Figma app, on the other hand, manages the exchange of prototypes and changes between team members through real-time updating.
- **Variants for components management:** Variants is a clever component management module. If a team has many copies of the same asset, it may easily merge them into one.

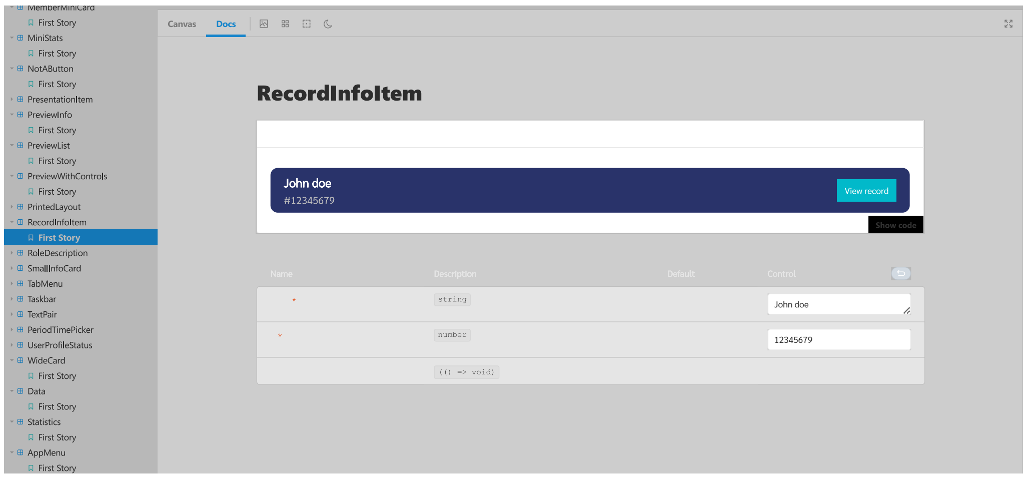
Utilizing these Figma tools to construct a project application from scratch, from the smallest component design to completely decorated pages is efficient and straightforward.

React

React.js, usually known as React, is an open-source, cost-free JavaScript library. The ideal way to create user interfaces is by assembling little pieces of code (components) into complete websites. It was initially created by Facebook, and is now maintained by the open-source community. For instance, the library can be used for creating one React component on a single page or design the complete website [77]. JSX, a JavaScript and XML mix, is used to create React.js. JSX is used to construct the elements, and JavaScript or typescript is used to render them on the application.

Storybook

Storybook enables programmers to create well-organized user interface UI components, enhancing the efficiency and usability of both the building and documenting phases. The beauty of Storybook is that it offers an isolated or distinct environment where developers can create and experiment with components, so no need to start the frontend/React development server. Front-end developers may create and present their components with documentation more quickly and easily thanks to this functionality.
The stories in a component's storybook describe the various stages of that component. A tale can be thought of as the smallest, or atomic, unit. A developer may create a story for a component and test out various parameters or props to see how it responds[86]. The figure 5.1 shows an example of using storybook



Name	Description	Default	Control
*	string		John doe
*	number		12345679
	(() => void)		

FIGURE 5.1: Storybook UI

5.2.4 Design patterns and Workflow

To ease the user's transition from the physical world to the digital one, it's understandable that realism was the first trend in visual design. The invention of more advanced features in digital tools and an increase in user confidence led to the birth of flat design, which later developed into minimalism.
The minimalist user interface design gives users an image of professionalism and clarity. Moreover, it offers benefits beyond the visual beauty. A minimalist interface loads quickly, takes less resources than a more complicated website.
Color selection is a major factor, if not the most significant factor, in design. it affects how people evaluate written content, how they interact with a layout, and how they feel about it. Color theme is divided into 3 categories which are :

- **Primary color:** As the name implies, a primary color is the one that is used most frequently throughout the app's pages and components.

- **Secondary color:** Adding a secondary color to the design gives an additional room to play with contrast and accent. The secondary color, if is used should be put wisely to draw attention to key features of the user interface.
- **Tertiary color:** These colors are made by combining different shades of the primary and secondary colors.

Layout and architecture

The software leverages a block separation design in which UI elements are represented into building blocks. The blocks are stacked in levels, with the highest tier dominating the view port.

Layers: As the first level building blocks, layers work as an entry point for the UI. They are used to render the higher-level building blocks like containers and components, but they don't provide any UI styles or elements themselves.

Containers: Containers contain a collection of components and rely on them to generate meaningful content. However, they may have their own user interface elements and styles. They are classified in the second level of building blocks.

Components: Are the third building blocks level, components are UI elements that are used to represent pieces of data, it is used to refer to reusable and stand-alone bits of code. In the same way that JavaScript functions do their job, these do the same thing but operate independently and return HTML.

Modals: Modals are in lowest building blocks level they do not belong to any layer, they pops up as a window over the current view. Since modals add an overlay to the screen, they are the primary focus of the user's attention. The figure 5.2 show an example of a modal.

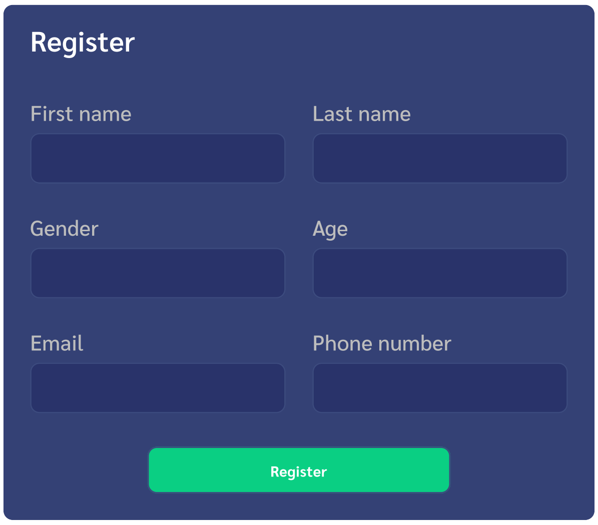


FIGURE 5.2: Modal example

Software states

The software states describes what the software should look like at a specific time. States depends on conditions that if met they define the state. The software have a three main states:

- First-use - When a user first start the software.
- Offline - When the user is not connected to any clinic server.
- Connected - When the user is connected to a clinic server.

Layers hierarchy

Layers represent the current state of the software. The state's context establishes the links between them as shown in figure 5.3. Each layer stores its own unique collection of containers and components, and these layers can communicate with one another and exchange data in order to better serve the need of the user.

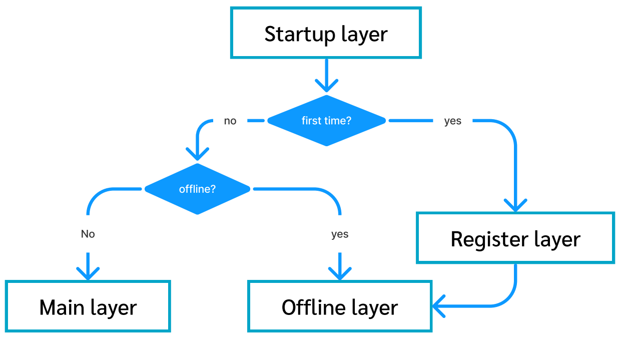


FIGURE 5.3: Layers hierarchy.

Startup layer: The startup layer is the entry of the software. Its only purpose is to forward the user to the next layer depending of the current state.

Register layer: Register layer is the responsible for the first-use state. Its the first layer that will be presented to the user when he launches the software for the first time. It displays a single registration modal that contain a form component to collect user information. The provided information are stored locally for the next usage.

Offline layer: It is a layer that represents the offline state. displayed when either the user is offline, or when he is not connected to any clinic server. Its only display a single content container which contains the necessary components to allow users to either reconnect to a previously joined clinic server or join a new one.

Main layer: Once the conditions that determines the connected state are fulfilled, the main layer will be displayed. The main layer is responsible for the major functionalities that the software provides. the layout is divided into three horizontal containers as shown on the figure 5.4:

- **Menu:** It is a container on the left side of the view port with the job of helping the user navigate through the software. Menu items are role based which mean each item is rendered if the user has the required permissions.
- **Content:** It's the central container that occupies practically the most viewport in order to render various pages depends on the navigation path. Each page reflects various services and functionalities the software offers.
- **Sidebar:** A container that displays components which contains information about various services, such as the queue.

Session layer: When a medical session started, this layer will be displayed to the user. It contains components that allow members to diagnosis and generate prescriptions to patients, divided by two horizontal containers:

- **Sidebar:** The sidebar container is located to the left side of the viewport, it shows components that contain information about the patient, from test results to medical documents and history.
- **Content:** It is responsible for displaying components that are necessary to perform a medical session.

Workflow

The Implementation is the result of all the studies that goes during the development of the software. The key for flexibility and consistency in collecting and presenting a large set of services in a simple user interface is the orchestrated patterns and organized activities that define each phase of the development. The collection of those phases and the translation between one phase and an other in a smooth manner is what makes a good workflow. The development phases are described as the following:

Whiteboard: The software design started as an idea in a white board with varies scratches and wire-frames for different idea's of implementation. With suggestions and feedback, the best appealing design is chosen.

Design phase: The design phase is the transformation of whiteboard scratches and wire-frames to an actual victors and graphs that when combined, make a user interface.

Coding phase: The result of the design phase is not enough as it only results a static vectors. Converting those vectors and graphs to a code is what gives it the dynamic of an actual software.

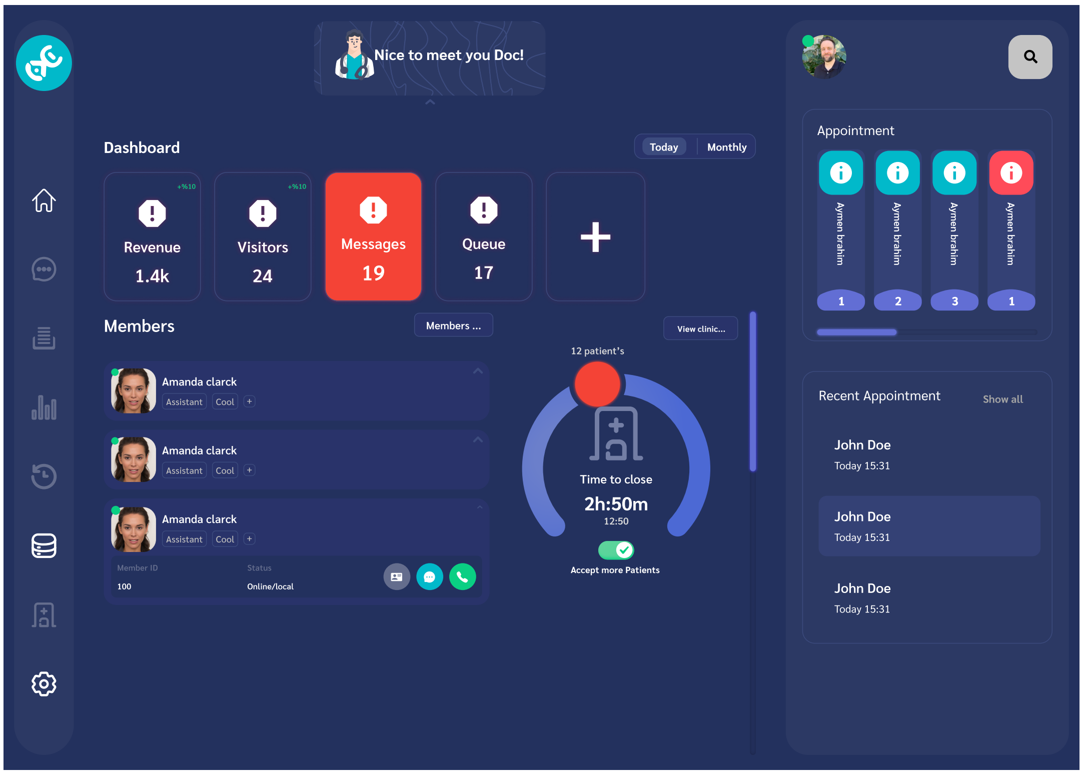


FIGURE 5.4: Main layer example

5.2.5 State management and local storage

The state

All React components come equipped with their own state object. State is encapsulated data that stores information that are shared between renderings of different components. The state is a generic name for a data structure in JavaScript. The application's user interface (UI) may look very different when a user interact with it causing a state change, since the new state will be used to represent the UI rather than the previous one.
In React applications, there are four primary forms of state that must be managed correctly:

- **Local state:** The data that is being handled by a single component is considered local state. In React, the useState hook is typically used to manage local state.
- **Global state:** The data that is being handled across all of the application is called the global state. Accessing and updating data across multiple app components requires the use of global state. One common type of global state is the information about a user's authentication status. The ability to retrieve and update a user's information within the app depends on their login status. The project application uses a libary called Redux. It is JavaScript's state container. As the app grows, it's harder to maintain data flow and organization. Redux manages application state with a global object called Store. Redux fundamentals help maintain application consistency, which simplifies debugging and testing.
- **Server state:** Data from an external server that must be integrated with the UI state. Server state is a simple concept, but it can be hard to manage alongside local

and global UI state. Loading and error state must be managed every time a fetch or update data from an external server. Fortunately, a tool like RTK Query makes managing server's state easier.

- **URL state:** It is an information about URLs, such as the path name and query parameters. Even though it should be considered as a distinct category of state, URL status is all too often overlooked. RTK Query functionalities manage this kind of states in order to perform queries and mutations on certain portion of data.

local storage

Local storage is a web storage object that stores data on the user's computer and persists between browser sessions. The Redux state can be saved in persistent storage using local storage as engine, thanks to a new library package called Redux Persist. Because of this, the state of certain portion of the app will be maintained even after a browser refresh. Additionally, Redux Persist provides methods to modify the persisted and rehydrated state in a way that suits the needs.

5.2.6 Bundling and building

A React.js project can be executed in two modes: development and build production. During the development phase, code is executed locally in development mode, where React provides numerous helpful warnings and tools for locating and resolving problems in the application code and eliminating potential bugs.
In production mode, the code will be minified, the assets will be optimized, and lighter-weight source maps will be generated. Additionally, all of the debugging and development mode capabilities, including warning messages, will be disabled. This significantly decreases the bundle size, which speeds up page loads.

Webpack

Traditionally, when constructing an application with JavaScript, the code would be broken up into individual files (these files may or may not have been actual modules). As a result, a script tags that represent the code of of these files should be added in the index.html file in order for application to work. Webpack's main function is to analyzs the application's modules, generates a dependency graph, and intelligently bundles them together so that the index.HTML file can make use of them.

Nodejs

Nodejs is a free and open-source server-side and network application development framework. Applications built with Nodejs can be deployed to and run on any platform that supports the Nodejs runtime, including Mac OS X, Windows, and Linux.
To further facilitate the creation of web applications, Nodejs includes a comprehensive collection of useful JavaScript modules.

Electron

Electronjs is a runtime framework that allows developers to create desktop applications with HTML5, CSS, and JavaScript. With Electron, it is possible to develop cross-platform apps for Windows, macOS, and Linux while maintaining just one JavaScript (or Typescript) code base thanks to the integration of Chromium and Nodejs in its binary. which means that JavaScript frameworks like React can be used to implement the front-end of the applications.

5.3 Server software

The server software is the core component of the system. It work by providing different services to the client software. Its subject to the server-client communication to enable exchange of information between clients and the server. It also provides a centralized pipe of communication which allows members to exchange information between them using the server software as a middle-man.

5.3.1 Technologies

The server software provides services that allow the system to function by utilizing various technologies.

NestJS

The server software is build using a well-known NodeJS framework called NestJS. NestJS is a progressive framework for building efficient, reliable and scalable server-side applications. NestJS architecture allows to combine multiple resource providers to form a knowledge base. It also provides an http communication channel which allows the exchange information through the network using an API.[65]

PostgreSQL

PostgreSQL is an object-relational database management system (ORDBMS) with an emphasis on extensibility and standards compliance. It is a free and open-source software product, and is available for all major operating systems. It is a relational database management system (RDBMS) based on the SQL (Structured Query Language) standard, and is fully ACID compliant. It is also fully transactional, meaning that all changes to the database are atomic, consistent, isolated, and durable.[75]

5.3.2 Server components

The server software adheres to a particular architecture in which services or functionalities are divided into distinct components. wherein each component serve distinct functionality such as providing or manipulating a distinct resource. The purpose of this architecture is to isolate system services so that each component manages a specific service or manipulates a specific block of data, aiding scalability and preventing resource collisions. Components can communicate and exchange information with one another, and some rely on others to perform their functions. Figure 5.5 represent the server architecture.

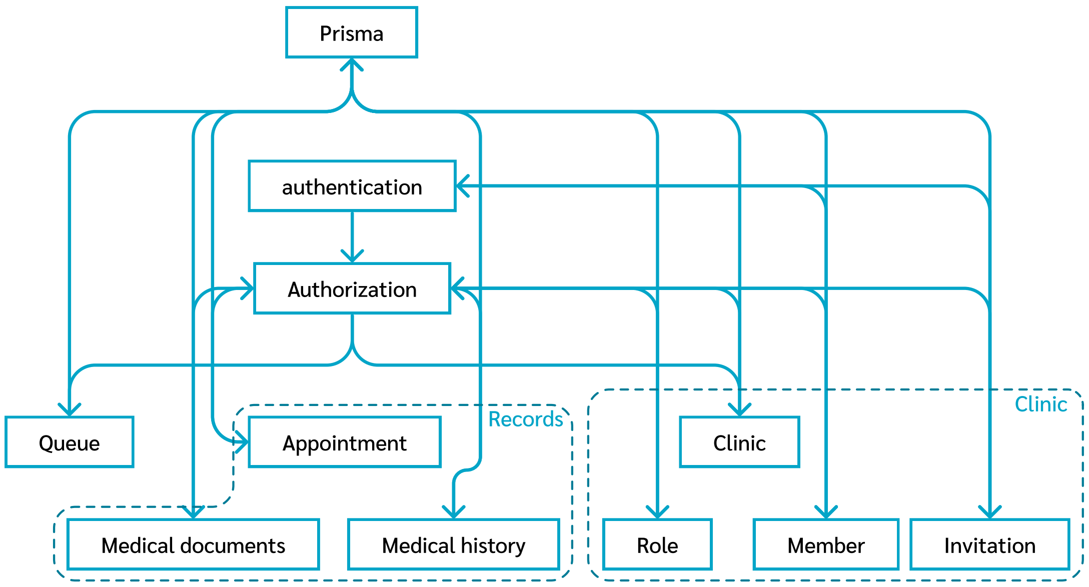


FIGURE 5.5: Server components architecture.

Database (Prisma)

The database (Prisma) component is responsible for all communication between the database server and the software services. Database component uses an open source toolkit called Prisma, Prisma is an Object-Relational Mapping (ORM)[1] database toolkit that provides a clean and type-safe API to communicate with database [95]. Prisma component only functionality is to provide other components the API to access the data.

Authentication

Authentication is a process of verifying the identity of a user or process. The authentication process is a sensitive part to identify users and prevent malicious access to the system. Each member need to be identified before he can perform any action.
The authentication component is the only component that can access the database directly to verify user credentials and to create new users. The authentication component is also responsible for generating and managing the JSON Web Tokens (JWT)[2], tokens that are used to identify users and to provide access to the system. The authentication component provide an API controller that enable communication between the server and the client software, and it allows the following actions:

Generating tokens : After each successful connect, register, or relink request, the authentication component generate an access JWT token that contains the user

[1]Object-Relational Mapping (ORM): is a technique that enables database querying and data manipulation using an object-oriented paradigm.

[2]JSON Web Tokens: are an open, industry standard method for representing claims securely between two parties.

information, and a refresh JWT for the purpose of refreshing authentication. The tokens then will be sent to the client software to be used in future requests.

Refresh : It is the process of refreshing the user access token. After a successful refresh the user will be provided with a new access token that will be used to identify the user and allow access to the system.

Register: It is a process of joining the clinic server as a new member. Registration requires an invitation key to be provided. After a successful registration, the client software will be provided with a secret key that allows to regain access to the member account.

Connect: It is the process of verifying the user credentials and generating a JWT token that will be used to identify the user and provide access to the system. The member is required to provide the secret key that was given to him after the registration process to be verified.

Relink: It is the process of regaining access to the member account after the member loses his secret key. Relink requires the member to provide an invitation key that can be provided by a permitted member with previous known information like the member name and email. After a successful relink the client software will be provided with a new secret key that allows to regain access to the member account.

Disconnect: It is the process of invalidating the user access token. After a successful disconnect the user will be required to connect again to regain access to the system.

Authorization

The authorization component implements the Role-based Access Control (RBAC) model. It works by providing an authorization API to other components which then can be used by those components to verify the user's ability for specific permissions which then will be used to determine whether the user can access a specific resource or not.

Queue

The queue component is responsible for managing and providing queue resources. The queue component depends on the role component to initiate and manage queues. Each role with sufficient permissions can have a queue, with only members of that role or an assist role that is linked to it which can manage that queue resources. The queue component provides an API controller to the client software to allow permitted members to perform the following actions on queues:

Add item adds an item to the queue which contains the patient's name and id with his position and test results if any.

Remove item removes an item from the queue.

Update item updates an item in the queue, usually updating only test information.

Get items gets items from the queue.

Get queue state gets the current state of the queue which can be one of the following: *waiting*, *in progress*, *paused* or *idle*.

A queue always has an owner which is the role that created it, and a queue can have multiple assistants which are members that have an assist role that is linked to the owner role.

Records

The record component is responsible for managing and providing patient records resources. Each permitted member can add a medical record to a patient, but members can only view records that they have committed. The record component is the combination of three other components which are medical documents, medical history, and the appointment component.

Medical documents: This component is responsible for managing and providing medical documents which are files that belong to a specific patient. It provides an API controller to the client software to allow permitted members to perform addition, view, and deletion of the medical documents.

Medical history: This component is responsible for managing and providing medical history which is information about patient health conditions history. It provides an API controller to the client software to allow permitted members to perform addition, view, and deletion of the medical history.

Appointments: This component is responsible for managing and providing appointments which are information about patient appointments in the clinic. Appointments contain information about previous medical sessions and booked appointments. It provides an API controller to the client software to allow permitted members to multiple actions.

Clinic

The clinic component is the main component of the server. This component is responsible for managing and providing clinic resources. It contains three necessary components which are:

Role: This component is responsible for managing and providing roles and resources. It uses the RBAC model to manage roles and permissions, and it provides an API controller to the client software to allow permitted members to perform actions such as creating, updating, or removing a role, and revoking/assigning a role to a member.

Member: This component is responsible for managing and providing members with resources. It is used by other components in the system.

Invitation This component is responsible for managing and providing invitations resources. It provide an API controller to the client software to allow permitted members to perform actions such as creating, updating, or consuming an invitation.

5.3.3 Database server

The database server uses a PostgreSQL database server to store all the data in the system. It's a NodeJS script, its job is initiate and start the PostgresSQL server using the official PostgresSQL binary files. This allows the database server to be portable and embedded with the software without requiring a separate installation. When the database server first started, it checks if there is already a database server running, and if that is our server. If it is not, the script will start a new database server and creates the database with the required tables, otherwise it just connects to the database.

5.3.4 Invitation keys generation

The invitation key is a string used to authorize a limited number of users to perform a specific actions for a limited amount of time. The invitation key contains information that allows the client software to determine the server's IP address. The invitation key is generated using the clinic's IP address and a string of three random characters. The generation algorithm encrypts the IP address using the random string. The invitation key is then saved as a unique identifier in a database invitation table, in addition with a key type and other information. The invitation key decryption is performed by the client software which uses the decrypted IP address to communicate with the server. The figure 5.6 shows the invitation key generation process. There are two types of invitation keys:

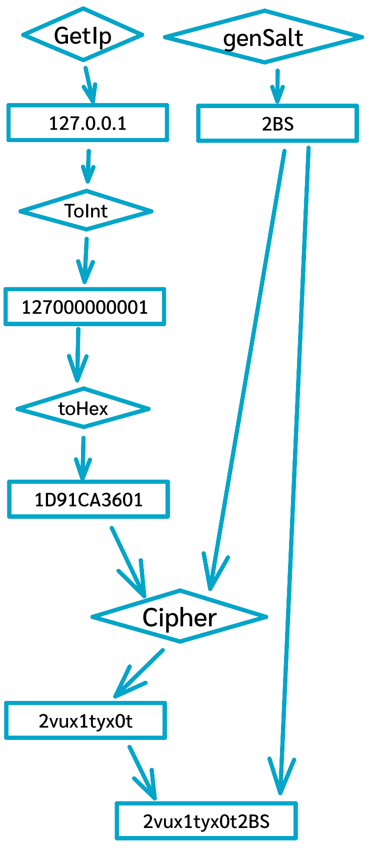


FIGURE 5.6: The process of generating an invitation key.

Join type it's used to register a new user to the clinic; it requires a set of roles that will be assigned to the user once he join the clinic.

Relink type It’s utilized to enable a member who has lost access to his account to regain access.

5.4 conclusion

Combining multiple technologies in a single system is a challenge, but with the help of the right tools and the right conception, it was possible to implement the system. The personal skills and the knowledge that was gained from this project will be very useful in the future. The system is still in development and will be improved in the future. Other software functionalities and protocols are presented in the appendix.

General Conclusion

Medical Practice management software provides a way to manage patient records, appointments, and billing. It was also responsible for storing and providing patient-sensitive data. According to health regulations and policies, patient privacy must be protected by the healthcare system. The goal of this thesis is to construct a novel secured system to protect patient privacy and information in Medical practice management software.
The system requires intercommunication between multiple devices to achieve its purpose which leads us to Use the intranet. By using the Intranet as a network security measurement we were able to limit access to the system through the network.
The process of limiting access to patient data requires an appropriate access model that suits the needs of the healthcare environment. The system uses Role-based Access Control to manage the authorization and protect the resources due to the balance and flexibility that it offer.
Healthcare system regulations and standards were taken into account during the system construction. The system focuses heavily on respecting health information policies by using various data protection practices that help in avoiding data breaches that may cause harmful exposure to patient privacy.
With The combination of networking security approaches and software security approaches, the software was implemented with the help of current technologies to serve the purpose of digitalizing medical practice management tasks while protecting patient health information.

Limitations

The software currently is in the state of a preview model and it's not ready for production. The development of the software was limited by time constraints and the lack of knowledge. Due to the size of the project, it was not possible to implement all the features that were planned.

future improvements

The system is still in development and there still a long way for the system to be considered stable. The system need to be tested in a real world environment, by testing it in a high scale healthcare facility we will be able to collect feedback about the performance of the system and about requirements or improvements that can be made to the system. There are features that was not implemented in the system due to the time limitation and the lack of knowledge about the healthcare industry. The following features are planned to be implemented in the future:

- The system should be able to support multiple forms of medical session.
- The system should support multiple languages, currently the system only support English.
- The system should be able to support multiple types of medical devices.
- The system will offer a real-time video and audio streaming feature.
- A mechanism will be implemented to enable the exchange of patient data under their consent. The data can transfer through the internet, and between members of different clinics or healthcare facilities.

Appendix A

Coding the software

A.1 Coding the Client software

The client software was codded using Typescript as a programming language with react library for UI manipulation. Various tools and techniques has been used during the development to help in the construction of the software and to make it more efficient.

A.1.1 React components implementation:

To implement a reusable react component, it requires the separation of the component into three parts: the component itself, the storybook and the styling file.

TSX file:

This file contains the html code and the logic of the component written with Typescript. The figure A.1 shows an example of the file.

SCSS file:

The figure A.2 represent the style sheets of a react component. The style is isolated in its own file to make it easier to adjust and modify the UI concerns.

Storybook file:

The storybook is a file that contains a code that allows to preview and test the component. the figure A.3 shows the configuration file of storybook.

A.1.2 Modal component:

Modal components are essentially pop-up containers that appear within the application. In user interface design, they are frequently used to present messages or request user agreement or information like a form on a course of action. The figure A.4 illustrates a modal that displays a warning dialog.

A.1.3 Redux structure:

There are three major components of Redux toolkit:

```tsx
interface MembersPanelProps {}
function MembersPanel({}: MembersPanelProps) {
  const { data, isLoading, isSuccess, error } = useGetMembersQuery();
  const { navigate } = useNavigation();
  return (
    <div className="members-panel">
      <Header
        title="Members"
        buttonNode={
          <DarkLightCornerButton
            onPress={() => {
              navigate('/clinic/Members');
            }}
            text="Members ... "
          />
        }
      />
      <div className="members-list-container">
        <div className="members-list">
          {isLoading ? (
            <LoadingSpinner />
          ) : isSuccess ? (
            data.length > 0 ? (
              data.map((member, index) => (
                <MembersPreview
                  {...member}
                  key={member.id.toString() + index}
                />
              ))
            ) : (
              <div>No Members</div>
            )
          ) : (
            <span>error occurs when getting members </span>
          )}
        </div>
      </div>
    </div>
  );
}
```

FIGURE A.1: TSX file example

Redux store:

Creating a store using the redux toolkit is super simple. the below code on the figure A.5 represent the configuration of the store.

Redux slice:

Redux Toolkit provides an API called *createSlice*. Which handles the initial state, automatically creates action types and creators. The figure A.6 shows an example of the slice that manage connection state.

RTK query file:

CreateAPI is the core function for RTK query. It defines a collection of "endpoints" for sending and receiving requests from a REST API. It generates an "API slice" structure that contains Redux logic to fetch and cache data. Building an RTK query file for Clinic data is illustrated in the code at A.7.

```scss
@use '~styles/appColors' as *;
.members-panel {
  display: flex;
  flex-direction: column;
  justify-content: flex-start;
  align-items: stretch;
  padding: 0px;
  gap: 20px;
  flex-grow: 1;
  > .header {
    display: flex;
    flex-direction: row;
    justify-content: space-between;

    > span {
      color: $white;
      font-weight: 600;
      font-size: 23px;
    }
  }
  > .members-list-container {
    overflow-y: scroll;

    > .members-list {
      padding-right: 10px;
      display: flex;
      flex-direction: column;
      justify-content: flex-start;
      align-items: stretch;
      gap: 5px;
    }
  }
}
```

FIGURE A.2: Style file

```tsx
export default {
  title: 'MembersPanel',
  component: MembersPanel,
};
const Template: Story<ComponentProps<typeof MembersPanel>> = (args) => (
  <MembersPanel {...args} />
);
export const FirstStory = Template.bind({});
FirstStory.args = {
  membersList: [
    {
      fullName: 'John Doe',
      imgSrc: test,
      memberID: '100',
      status: 'Online',
      roleArray: ['doctor', 'Assistant1', 'Assistant2', 'Assistant3'],
    },
  ],
};
```

FIGURE A.3: Storybook configuration

A.1.4 Project Configurations:

Configuration is essential in software development because it ensures that all the necessary tools are set up to work together in cooperation to accomplish a defined objective. Multiple configuration files are utilized for the project configuration.

```
interface WarningModalProps {
  title: string;
  description: string;
  children?: ReactNode;
  content?: ReactNode;
}
export default function WarningModal({
  title,
  description,
  children,
  content,
}: WarningModalProps) {
  return (
    <ModalContainer
      gap={10}
      title={title}
      controls={<div className="warning-controls">{children}</div>}
    >
      <span className="warning-description">{description}</span>
      {content}
    </ModalContainer>
  );
}
```

FIGURE A.4: Modal example

Package.json:

The figure A.8 represents the core file of any Nodejs project. It contains information that defines the features of a project. Tools such as node package manager utilizes this file to install dependencies or run scripts. It also contains information about licensing and the identity of authors.

tsConfig.json:

tsconfig.json is a typescript configuration file that contains a set of information that are required by the typescript engine to compile the project. It includes information like the location of the project's root files and compilations settings. Figure A.9 represent the configuration file.

Webpack config file:

The basic features of Webpack can be customized and expanded with the help of Webpack configs. One of Webpack's settings can be adjusted by modifying a JavaScript object called a Webpack config. The majority of projects use a *webpack.config.js* file at the root to define their setup as shown on the figure A.10.

A.2 Coding the Server software

Node.js and Typescript were utilized to develop the server software in order to reduce development time and ensure a consistent codebase. The combination of the Nestjs framework with tools such as Prisma, PostgresSQL, and websocket makes development more efficient and the codebase easier to maintain.

A.2.1 Nestjs structure

Nestjs is a framework for building efficient, scalable Node.js server-side applications. It uses progressive JavaScript, is built with and fully supports TypeScript and combines elements of OOP (Object Oriented Progamming), FP (Functional Programming), and FRP (Functional Reactive Programming). Nestjs architecture divides code to the following:

Controller file:

Controller files are responsible for handling the incoming requests and returning the response. an example of a controller is shown in figure A.11.

Service files

Service files represent the function of handling the business logic. An example of a service file is shown in figure A.12.

Dto files

Dto files contain classes for a Data Transfer Object, an object used to encapsulate and transmit data from one application subsystem to another. . An example of a guard file is shown in figure A.13.

Guard files

A guard file contain code that is responsible for handling the authentication and authorization. An example of a guard file is shown in figure A.14.

A.3 Software walk through

- **Appointments queue page:** displays the patients' scheduled appointments, the queue list, and appointment booking controls. The user interface of the page is shown in figure A.15.
- **Session page:** shows the current medical session, which includes information about the patient, elements for describing a prescription, input for writing a diagnosis, and other customizable options. The figure A.16 represents the page.
- **Clinics page:** shows the available clinics to connect and option to join a new clinic by invitation key. The figure A.17 shows an example of rendered clinics page.
- **Records page:** displays details about the patient, their records, and all of their appointments. The figure A.18 describe the record page.
- **Invitation modal:** shows options to create an invitation to join or re-link a member. The figure A.19 shows the modal on join option selected.
- **Join clinic modal:** displays a field to enter a valid invitation key in order to join a clinic. The figure A.20 is example of a Join clinic modal with invitation key entered.

- **Connect modal:** this modal popup when already joined member try to reconnect with secret key. The figure A.21 is example of a Connect modal with secret key entered.
- **Add patient to queue modal:** this modal is designed to add a patient to the queue or to run test for the selected patient. The figure A.22 represent the UI of the modal.

A.4 System installation

The installation process is a crucial and should be processed carefully by a professional. The installation of the system requires a basic knowledge of networking hardware and a basic knowledge of the software.

A.4.1 Hardware

The installation of the system requires basic knowledge of computer and network hardware. The system does not require expensive or specific network equipment, a basic home-use router device with a WIFI or cable ethernet interface support, will satisfy the needs for a simple clinic facility. However, the scale of the clinic will rise the hardware requirements and the complexity of the setup, at which point it will become necessary to consult a professional IT.
The system also requires a computer that runs the server software. The computer device needs to be connected to the network with a sustainable energy source. The security of the device is the responsibility of the owner of the clinic server that's why a professional installation is recommended. Enabling a firewall to prevent unauthorized internet access for certain application and preventing the human access to the device with addition to a strong antivirus will improve the server security.
Every computer device on the network should be able to join the clinic server if it's connected to the same network as the server software. Members of the clinic have to be responsible for the security of their devices. The health facility can reduce the risk of a security breach by providing a computer device for each member of the clinic or by training the members of the clinic on how to secure their devices.
Interconnecting can be done by either using a cable or a wireless connection. The wireless connection is more convenient and easier to setup, however, it is less secure and more prone to security breaches. The cable connection is more secure but it is complicated to setup and requires more effort to maintain.
The network hardware should be able to support the number of devices that will be connected to the network. It's easy to use a device that have a built in Dynamic Host Configuration Protocol (DHCP) server such as home routers to provide dynamic IP address to the devices on the network. However, it's possible to use Automatic Private IP Addressing (APIPA) that is provided by the operating system instead of DHCP.
The server computer device need to be assigned to a static IP address to avoid manually updating the IP address on the client software each time the server IP address changes.

A.4.2 Software

The installation of the client software is a simple process that can be done by a non-technical person. The installation process requires the following steps:

1. Install the software on the computer device.
2. Run the software.
3. Join the clinic.

The installation of the server software requires knowledge of computer and network hardware. The installation process requires the following steps:

1. setup a firewall.
2. install an antivirus.
3. Secure the computer device with a password.
4. Install the software on the computer device.
5. Run the server software.
6. Assure that the server software will be able run at all circumstances.

```
const persistUserConfig = {
  key: 'user',
  storage,
  //  whitelist: [...Object.keys(template)],
};
const persistAuthConfig = {
  key: 'auth',
  storage,
  whitelist: ['refreshToken', 'accessToken'],
};

const persistedUser = persistReducer(persistUserConfig, userSlice.reducer);
const persistedAuth = persistReducer(persistAuthConfig, authSlice.reducer);

const appReducer = combineReducers({
  [clinicApi.reducerPath]: clinicApi.reducer,
  [smallRoleInvSlice.name]: smallRoleInvSlice.reducer,
  [roleApi.reducerPath]: roleApi.reducer,
  [memberApi.reducerPath]: memberApi.reducer,
  [invitationApi.reducerPath]: invitationApi.reducer,
  [authSlice.name]: persistedAuth,
  [settingsSlice.name]: settingsSlice.reducer,
  [sessionSlice.name]: sessionSlice.reducer,
  [connectionStateSlice.name]: connectionStateSlice.reducer,
  [userSlice.name]: persistedUser,
  [authApi.reducerPath]: authApi.reducer,
  [AppointmentQueueApi.reducerPath]: AppointmentQueueApi.reducer,
  [patientApi.reducerPath]: patientApi.reducer,
  [medicalDocumentApi.reducerPath]: medicalDocumentApi.reducer,
  [medicalHistoryApi.reducerPath]: medicalHistoryApi.reducer,
  [appointmentApi.reducerPath]: appointmentApi.reducer,
  /* your app's top-level reducers */
});

const rootReducer = (
  state: ReturnType<typeof appReducer> | undefined,
  action: AnyAction
) => {
  if (action.type === 'RESET' && state) {
    const myState = Object.fromEntries(
      Object.entries(state).map(([key, value]) => [
        key,
        key == 'user' || key == 'authSlice' || key == authApi.reducerPath
          ? value
          : undefined,
      ])
    ) as ReturnType<typeof appReducer>;

    return appReducer(myState, { type: undefined });
  }
  return appReducer(state, action);
};
export const store = configureStore({
  reducer: rootReducer,
  middleware: (getDefaultMiddleware): Middleware[] =>
    getDefaultMiddleware({
      serializableCheck: false,
    }).concat(
      rtkQueryErrorLogger,
      AppointmentQueueApi.middleware,
      patientApi.middleware,
      medicalDocumentApi.middleware,
      medicalHistoryApi.middleware,
      appointmentApi.middleware,
      authApi.middleware,
      invitationApi.middleware,
      memberApi.middleware,
      roleApi.middleware,
      clinicApi.middleware
    ),
});
```

FIGURE A.5: Store configuration

```
const initialState: {
  state:
    | 'connected'
    | 'connecting'
    | 'unreachable'
    | 'reconnecting'
    | 'disconnected'
    | undefined;
} = {
  state: undefined,
};

const connectionStateSlice = createSlice({
  name: 'connectionState',
  initialState: initialState,
  reducers: {
    disconnect: (state) ⇒ {
      state.state = 'disconnected';
    },
    refresh: (state) ⇒ {
      state.state = 'reconnecting';
    },
    connect: (state) ⇒ {
      state.state = 'connecting';
    },
    connected: (state) ⇒ {
      state.state = 'connected';
    },
    unreachable: (state) ⇒ {
      state.state = 'unreachable';
    },
  },
});

export const disconnect = (dispatch: Dispatch) ⇒ {
  Promise.resolve(dispatch(setSelectedServer())).then(() ⇒ {
    dispatch(connectionStateSlice.actions.disconnect());
    dispatch({ type: 'REST' });
  });
};
export const connect = (dispatch: Dispatch, selectedIndex: number) ⇒ {
  Promise.resolve(dispatch(setSelectedServer(selectedIndex))).then(() ⇒ {
    dispatch(connectionStateSlice.actions.connect());
  });
};
```

FIGURE A.6: Slice file

```
const clinicApi = createApi({
  reducerPath: 'clinicApi',
  baseQuery: StaticQueries.clinic.query,
  tagTypes: ['clinic'],
  endpoints: (builder) ⇒ ({
    getClinic: builder.query<Clinic, void>({
      query: () ⇒ '',
      providesTags: ['clinic'],
    }),
    updateClinicOverview: builder.mutation<
      boolean,
      { name: string; description: string; phone?: string; address: string }
    >({
      query: (body) ⇒ {
        return { url: '', body: { ...body }, method: 'PATCH' };
      },
      invalidatesTags: ['clinic'],
    }),
  }),
});
```

FIGURE A.7: RTK query file

```
"scripts": {
   "devServer": "set mode=development&& webpack serve  --env mode=development",
   "productionWeb": "set mode=production&& webpack  --env mode=production",
   "storybook": "set mode=development&& start-storybook -p 6006",
   "realtimeServer": "http-server ./build/renderer",
   "startApp": "set platform=electron&& set mode=development&& electron-forge start",
   "package": "set platform=electron&& set mode=production&& electron-forge package",
   "make": "set mode=production&& electron-forge make",
   "publish": "electron-forge publish",
   "eslint": "eslint --ext .ts,.tsx .",
   "eslint-fix": "eslint --ext .ts,.tsx --fix",
   "check": "prettier --check \"src/**/*.{js,jsx,ts,tsx}\"",
   "format": "prettier --write \"src/**/*.{js,jsx,ts,tsx}\"",
   "exampleServer": "http-server ./example",
   "check-sass": " npx stylelint \"src/**/*.{css,sass,scss}\"",
   "fix-sass": " npx stylelint \"src/**/*.{css,sass,scss}\" --fix",
   "build-storybook": "build-storybook",
   "test": "web-test-runner \"src/**/*.test.tsx\""
 },
```

FIGURE A.8: Package.json file

```
{
  "compilerOptions": {
    "module": "esnext",
    "target": "es5",
    "moduleResolution": "node",
    "jsx": "react-jsx",
    "jsxImportSource": "@emotion/react",
    "baseUrl": "./",
    "paths": {
      "*": ["node_modules/*"],
      "@components/*": ["src/renderer/ui/components/*"],
      "@buttons/*": ["src/renderer/ui/components/buttons/*"],
      "@containers/*": ["src/renderer/ui/containers/*"],
      "@layers/*": ["src/renderer/ui/layers/*"],
      "@assets/*": ["src/renderer/assets/*"],
      "@libs/*": ["src/renderer/libs/*"],
      "@constants/*": ["src/renderer/utils/constants/*"],
      "@helpers/*": ["src/renderer/utils/helpers/*"],
      "@models/*": ["src/renderer/models/*"],
      "toSvg/*": ["src/renderer/assets/svg/*"],
      "toPng/*": ["src/renderer/assets/pictures/*"],
      "@api/*": ["src/renderer/API/*"],
      "@store": ["src/renderer/services/redux/index.ts"],
      "@redux/*": ["src/renderer/services/redux/*"],
      "@colors": ["src/renderer/assets/styles/color.ts"],
      "@ability/*": ["src/renderer/services/ability/*"],
      "@stores/*": ["src/renderer/services/zustand/*"],
      "@channels/*": ["src/main/channels/*"]
    },
    "noEmit": false,
    "typeRoots": ["src/main/channel/types", "node_modules/@types"],
    "strict": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true,
    "resolveJsonModule": true,
    "allowSyntheticDefaultImports": true,
    "importsNotUsedAsValues": "remove",
    "esModuleInterop": true,
    "noImplicitAny": true,
    "noUnusedLocals": false,
    "noUnusedParameters": false,
    "sourceMap": true,
    "outDir": "dist"
  },
  "skipLibCheck": true,
  "include": ["src", "@types"],
  "exclude": ["node_modules"]
}
```

FIGURE A.9: tsConfig.json file

```
module.exports = {
  mode: 'development',
  entry: './foo.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'foo.bundle.js',
  },
 ...
};
```

FIGURE A.10: Webpack configuration

```
@Controller('queue')
@UseInterceptors(QueueInterceptor)
export class QueueController {
  constructor(private readonly queueService: QueueService) {}
  //QUEUE METHODs

...
  @ApiTags('Queue')
  @Delete('')
  clearQueue(@Queue() { id, owner }: QueueInfo) {
    if (!owner)
      throw new ForbiddenException('You are not allowed to update this queue');
    return this.queueService.clearQueue(id);
  }
  @ApiTags('Queue')
  @Get('ownership')
  getOwnership(@Queue() { owner }: QueueInfo) {
    return owner;
  }

  //ITEMS methods
  @ApiTags('Item')
  @Get('item')
  getQueueItems(@Queue() { id }: QueueInfo) {
    return this.queueService.getQueueItems(id);
  }
...
}
```

FIGURE A.11: A controller file example

```
@Injectable()
export class QueueService {
  constructor(
    private prisma: PrismaService,
    @Inject(forwardRef(() => AppointmentService))
    private appointmentService: AppointmentService,
    ) {}

  ...

  async getQueueItems(id:number): Promise<AppointmentQueueItem[]> {
    const res = await this.prisma.queue.findUnique({
      where: {  id },
      select: {
        QueueItems: {
          include:{
        patient: { select: { firstName: true, lastName: true } },
        appointment: { select: { id: true } },
      },orderBy: {position: 'asc'}}},

      });


    if (!res) {
      throw new NotFoundException(`Queue with id ${id} not found`);
    }

    return res.QueueItems.map(
      ({ patient, date, patientId, position, test, appointmentId }) => {
        return {
          date: date,
          position: position,
          patientId: patientId,
          patientName: patient.firstName + ' ' + patient.lastName,
          test: test as any as TestResult,
          appointmentId: appointmentId,
        };
      },
    );
  }
 ...
}
```

FIGURE A.12: A service file example

```
export class QueryIdDto {
  @IsNumber()
  @Min(1)
  patientId: number;
}
```

FIGURE A.13: A dto file example

```
import { Injectable, CanActivate, ExecutionContext } from '@nestjs/common';
import { Observable } from 'rxjs';

@Injectable()
export class AuthGuard implements CanActivate {
  canActivate(
    context: ExecutionContext,
  ): boolean | Promise<boolean> | Observable<boolean> {
    const request = context.switchToHttp().getRequest();
    return validateRequest(request);
  }
}
```

FIGURE A.14: A guard file example

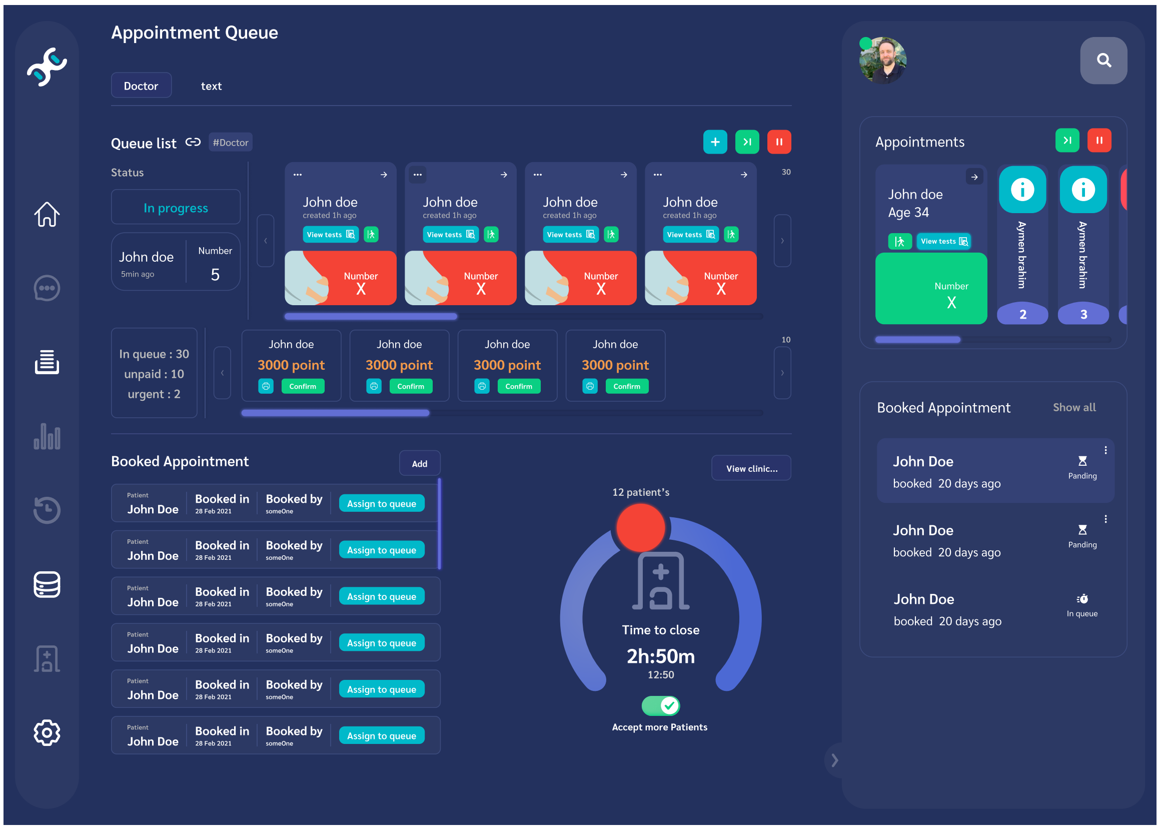


FIGURE A.15: Appointments queue page

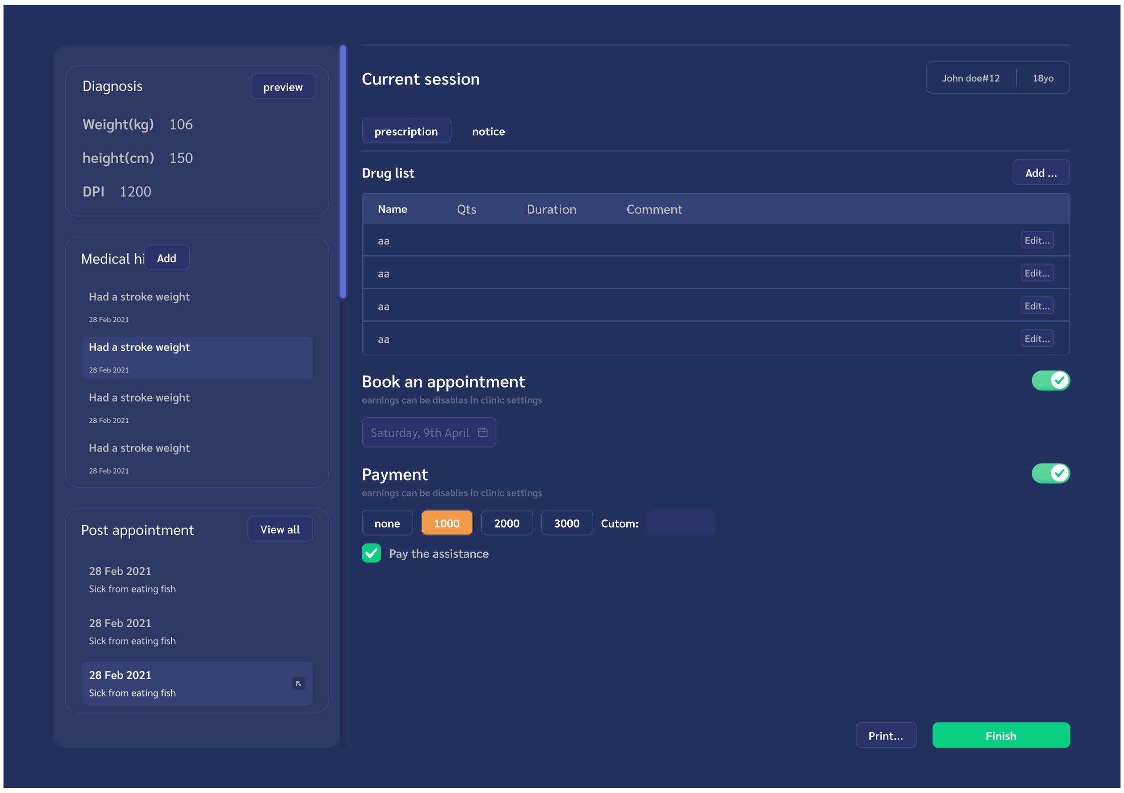


FIGURE A.16: Session page

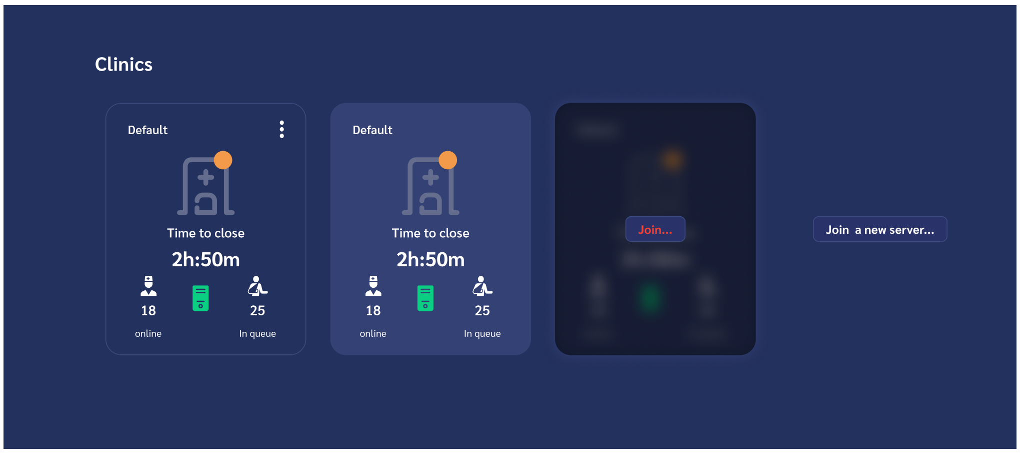


FIGURE A.17: Clinics page

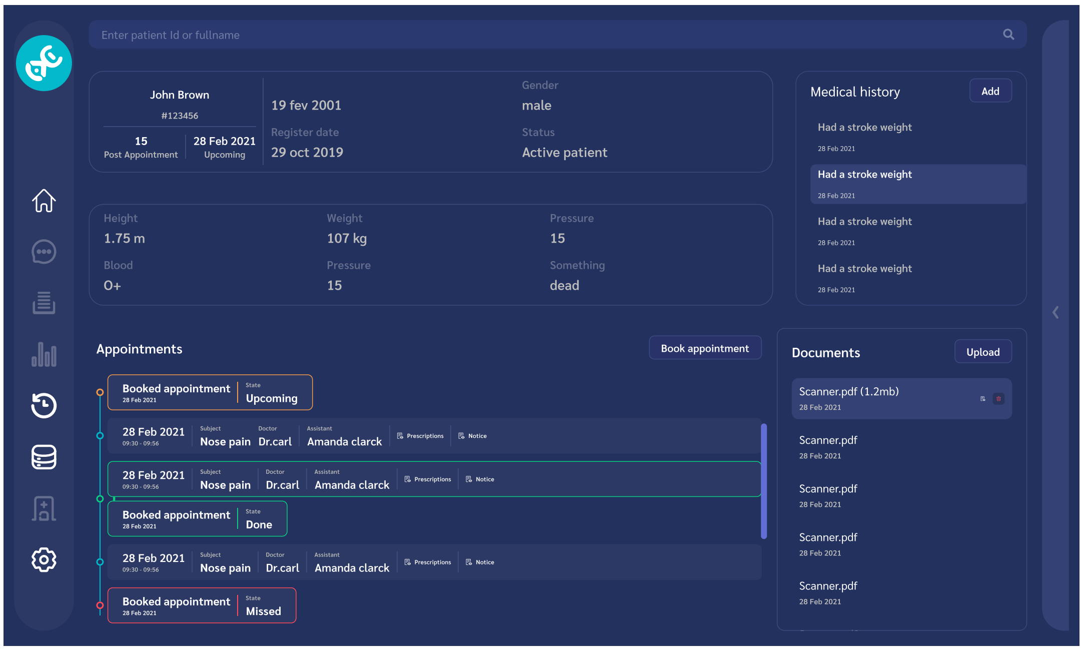


FIGURE A.18: Records page

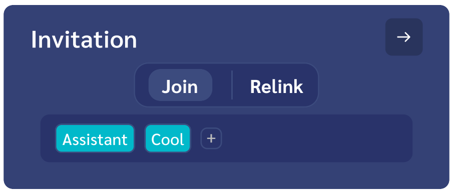


FIGURE A.19: Invitation modal

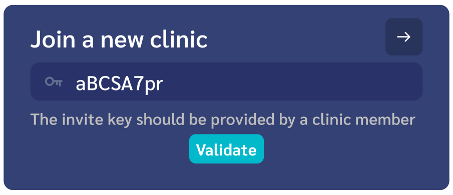


FIGURE A.20: Join clinic modal

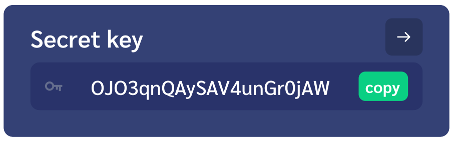


FIGURE A.21: Connect modal

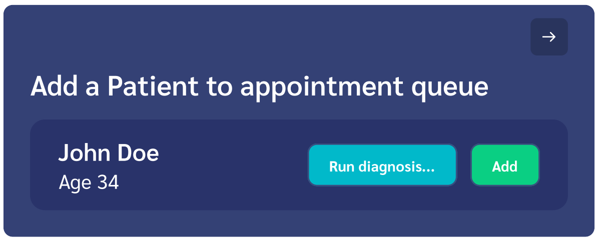


FIGURE A.22: Add patient to queue modal

Bibliography

[1] Syed Sibte Raza Abidi. “Healthcare knowledge management: The art of the possible”. In: *AIME Workshop on Knowledge Management for Health Care Procedures*. Springer. 2007, pp. 1–20.

[2] Ruth A Anderson and Reuben R McDaniel Jr. “Managing health care organizations: where professionalism meets complexity science”. In: *Health care management review* (2000), pp. 83–92.

[3] George J Annas. “HIPAA regulations: a new era of medical-record privacy?” In: *New England Journal of Medicine* 348 (2003), p. 1486.

[4] Raja Waseem Anwar, Tariq Abdullah, and Flavio Pastore. “Firewall Best Practices for Securing Smart Healthcare Environment: A Review”. In: *Applied Sciences* 11.19 (2021), p. 9183.

[5] Qing-hai Bai and Ying Zheng. “Study on the access control model”. In: *Proceedings of 2011 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*. Vol. 1. 2011, pp. 830–834. DOI: 10.1109/CSQRWC.2011.6037079.

[6] John Barkley. “Comparing simple role based access control models and access control lists”. In: *In Proceedings of the second ACM workshop on Role-based access control*. ACM Press, 1997, pp. 127–132.

[7] Sreenivasa Rao Basavala, Narendra Kumar, and Alok Agarrwal. “Authentication: An overview, its types and integration with web and mobile applications”. In: *2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing*. 2012, pp. 398–401. DOI: 10.1109/PDGC.2012.6449853.

[8] David W. Bates et al. “A Proposal for Electronic Medical Records in U.S. Primary Care”. In: *Journal of the American Medical Informatics Association* 10.1 (Jan. 2003), pp. 1–10. ISSN: 1067-5027. DOI: 10.1197/jamia.M1097. eprint: https://academic.oup.com/jamia/article-pdf/10/1/1/2438665/10-1-1.pdf. URL: https://doi.org/10.1197/jamia.M1097.

[9] Sergey V Belim and S Yu Belim. “Implementation of mandatory access control in distributed systems”. In: *Automatic Control and Computer Sciences* 52.8 (2018), pp. 1124–1126.

[10] Eta S Berner. *Clinical decision support systems*. Vol. 233. Springer, 2007.

[11] Tim Berners-Lee, Roy Fielding, and Henrik Frystyk. *Hypertext transfer protocol–HTTP/1.0*. Tech. rep. 1996.

[12] Kees Boersma and S. Kingma. “Intranet and Organizational Learning”. In: *Encyclopedia of Knowledge Management* (Jan. 2005), pp. 305–310. DOI: 10.4018/978-1-59140-573-3.ch040.

[13] WE Burr et al. “NIST Special Publication 800-63-2 Electronic Authentication Guideline”. In: *Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology* (2013).

[14] James Callaghan. *Inside Intranets and Extranets: Knowledge Management and the Struggle for Power*. Palgrave Global Publishing, 2002. ISBN: 0333987438.

[15] Vinton Cerf and Robert Kahn. “A protocol for packet network intercommunication”. In: *IEEE Transactions on communications* 22.5 (1974), pp. 637–648.

[16] *Chrome DevTools - Chrome Developers*. https://developer.chrome.com/docs/devtools/.

[17] D. Comer. *Computer Networks and Internets*. Pearson/Prentice Hall, 2009. ISBN: 9780136061274. URL: https://books.google.com/books?id=tm-evHmOs3oC.

[18] D. Comer and D.L. Stevens. *Internetworking with TCP/IP.: Client-server programming and applications*. Internetworking with TCP/IP v. 3. Prentice Hall, 2001. ISBN: 9780130320711. URL: https://books.google.dz/books?id=dNpFAQAAIAAJ.

[19] Adrienne Curry and Lara Stancich. “The intranet—an intrinsic component of strategic information management?” In: *International journal of information management* 20.4 (2000), pp. 249–268.

[20] F David, D Ferraiolo, and K Richard. “Ramaswamy chandramouli”. In: *Role-Based Access Control* (2007).

[21] F David, D Ferraiolo, and K Richard. “Ramaswamy chandramouli”. In: *Role-Based Access Control* (2007), pp. 64–65.

[22] F David, D Ferraiolo, and K Richard. “Ramaswamy chandramouli”. In: *Role-Based Access Control* (2007), pp. 65–66.

[23] F David, D Ferraiolo, and K Richard. “Ramaswamy chandramouli”. In: *Role-Based Access Control* (2007), pp. 68–71.

[24] F David, D Ferraiolo, and K Richard. “Ramaswamy chandramouli”. In: *Role-Based Access Control* (2007), pp. 77–78.

[25] Joshua Davies. *Implementing SSL/TLS using cryptography and PKI*. John Wiley and Sons, 2011.

[26] Tim Dierks and Eric Rescorla. *The transport layer security (TLS) protocol version 1.2*. Tech. rep. 2008.

[27] Pooja Dixit et al. “Traditional and hybrid encryption techniques: a survey”. In: *Networking communication and data knowledge engineering*. Springer, 2018, pp. 239–248.

[28] Christo El Morr and Julien Subercaze. “Knowledge management in healthcare”. In: *Handbook of research on developments in e-health and telemedicine: technological and social perspectives*. IGI Global, 2010, pp. 490–510.

[29] David Ferraiolo, Janet Cugini, D Richard Kuhn, et al. “Role-based access control (RBAC): Features and motivations”. In: *Proceedings of 11th annual computer security application conference*. 1995, pp. 241–48.

[30] David F Ferraiolo et al. “Proposed NIST standard for role-based access control”. In: *ACM Transactions on Information and System Security (TISSEC)* 4.3 (2001), pp. 224–274.

[31] *Figma: the collaborative interface design tool.* https://www.figma.com/.

[32] Christos K Georgiadis et al. “Implementing context and team based access control in healthcare intranets”. In: *Medical informatics and the internet in medicine* 27.3 (2002), pp. 185–201.

[33] Martijn Geurden. *A New Future for Military Security Using Fully Homomorphic Encryption*. 2018.

[34] *GitHub: Where the world builds software*. https://github.com/. [Online; accessed 2022-09-09].

[35] Candace Gray. “Understanding and complying with HIPAA”. In: *Journal of PeriAnesthesia Nursing* 18.3 (2003), pp. 182–185.

[36] Brian T Gregory et al. “Organizational culture and effectiveness: A study of values, attitudes, and organizational outcomes”. In: *Journal of business research* 62.7 (2009), pp. 673–679.

[37] Amal Hafsa et al. “A hardware-software co-designed AES-ECC cryptosystem”. In: *2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET)* (2017), pp. 50–54.

[38] Reinhold Haux. “Health information systems–past, present, future”. In: *International journal of medical informatics* 75.3-4 (2006), pp. 268–281.

[39] *Healthcare*. [Online; accessed 2022-07-30].

[40] Christian Heath, Paul Luff, and Marcus Sanchez Svensson. “Technology and medical practice”. In: *Sociology of Health & illness* 25.3 (2003), pp. 75–96.

[41] *HTML Tutorial*. https://www.w3schools.com/html/. [Online; accessed 2022-09-09].

[42] Vincent C. Hu et al. “Attribute-Based Access Control”. In: *Computer* 48.2 (2015), pp. 85–88. DOI: 10.1109/MC.2015.33.

[43] Bernard J Jansen. “The graphical user interface”. In: *ACM SIGCHI Bulletin* 30.2 (1998), pp. 22–26.

[44] *JavaScript With Syntax For Types*. https://www.typescriptlang.org/.

[45] Carole S Jordan. *Guide to Understanding Discretionary Access Control in Trusted Systems*. DIANE Publishing, 1987.

[46] Atul Kahate. *Cryptography and network security*. Tata McGraw-Hill Education, 2013, pp. 38–40.

[47] A.A.E. Kalam et al. “Organization based access control”. In: *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks*. 2003, pp. 120–131. DOI: 10.1109/POLICY.2003.1206966.

[48] Louiza Khati, Nicky Mouha, and Damien Vergnaud. “Full disk encryption: bridging theory and practice”. In: *Cryptographers’ Track at the RSA Conference*. Springer. 2017, pp. 241–257.

[49] Joseph Migga Kizza, Wheeler Kizza, and Wheeler. *Guide to computer network security*. Springer, 2013.

[50] Prakash Kuppuswamy and Saeed QY Al-Khalidi. "Hybrid encryption/decryption technique using new public key and symmetric key algorithm". In: *MIS Review: An International Journal* 19.2 (2014), pp. 1–13.

[51] James SH Kwok and Sheng Gao. "Knowledge sharing community in P2P network: a study of motivational perspective". In: *Journal of knowledge Management* (2004).

[52] Theo Lippeveld et al. *Design and implementation of health information systems*. World Health Organization, 2000.

[53] Peter A Loscocco et al. "The inevitability of failure: The flawed assumption of security in modern computing environments". In: *Proceedings of the 21st national information systems security conference*. Vol. 10. 1998, pp. 303–314.

[54] Richard WC Lui et al. "Role activation management in role based access control". In: *Australasian Conference on Information Security and Privacy*. Springer. 2005, pp. 358–369.

[55] Xiaopu Ma et al. "Specifying and enforcing the principle of least privilege in role-based access control". In: *Concurrency and Computation: Practice and Experience* 23.12 (2011), pp. 1313–1331.

[56] Lakmini P Malasinghe, Naeem Ramzan, and Keshav Dahal. "Remote patient monitoring: a comprehensive study". In: *Journal of Ambient Intelligence and Humanized Computing* 10.1 (2019), pp. 57–76.

[57] Ahmad Kamran Malik et al. "A Comparison of Collaborative Access Control Models". In: *International Journal of Advanced Computer Science and Applications* 8.3 (2017).

[58] Rob Manson. *Getting Started with WebRTC*. Packt Publishing Ltd, 2013.

[59] Simreen Kaur Matharu. "Exploiting SSL/TLS Vulnerabilities in Modern Technologies". In: (2020).

[60] *medintux-about*. https://medintux.org/about. [Online; accessed 2022-09-13].

[61] D. Meyer and G. Zobrist. "TCP/IP versus OSI". In: *IEEE Potentials* 9.1 (Feb. 1990), pp. 16–19.

[62] Evgeny Milanov. "The RSA algorithm". In: *RSA laboratories* (2009), pp. 1–11.

[63] Vijay Kumar Mitali and Arvind Sharma. "A survey on various cryptography techniques". In: *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)* 3.4 (2014), pp. 307–312.

[64] Jonathan D Moffett. "Control principles and role hierarchies". In: *Proceedings of the third ACM workshop on Role-based access control*. 1998, pp. 63–69.

[65] Kamil Mysliwiec. *Documentation | NestJS - A progressive Node.js framework*. https://docs.nestjs.com/. [Online; accessed 2022-09-09].

[66] Haroon Shakirat Oluwatosin. "Client-server model". In: *IOSR Journal of Computer Engineering* 16.1 (2014), pp. 67–71.

[67] *OpenEMR Features - OpenEMR Project Wiki*. [Online; accessed 2022-07-26].

[68] Rolf Oppliger. *SSL and TLS: Theory and Practice*. Artech House, 2016.

[69] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009.

[70] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009, pp. 259–289.

[71] H. Pang, K.-L. Tan, and X. Zhou. "StegFS: a steganographic file system". In: *Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405)*. 2003, pp. 657–667. DOI: 10.1109/ICDE.2003.1260829.

[72] Radia Perlman, Charlie Kaufman, and Mike Speciner. *Network security: private communication in a public world*. Pearson Education India, 2016, pp. 88–92.

[73] Radia Perlman, Charlie Kaufman, and Mike Speciner. *Network security: private communication in a public world*. Pearson Education India, 2016, pp. 92–96.

[74] Leonard Ponzi and Michael Koenig. "Knowledge management: another management fad". In: *Information research* 8.1 (2002), pp. 8–1.

[75] . *PostgreSQL: About*. https://www.postgresql.org/about/. [Online; accessed 2022-09-09].

[76] Bart Preneel. "Cryptographic hash functions". In: *European Transactions on Telecommunications* 5.4 (1994), pp. 431–448.

[77] *React – A JavaScript library for building user interfaces*. https://reactjs.org/.

[78] Pierangela Samarati and Sabrina Capitani de Vimercati. "Access control: Policies, models, and mechanisms". In: *International School on Foundations of Security Analysis and Design*. Springer. 2000, pp. 137–196.

[79] Ravi Sandhu, David Ferraiolo, Richard Kuhn, et al. "The NIST model for role-based access control: towards a unified standard". In: *ACM workshop on Role-based access control*. Vol. 10. 344287.344301. 2000.

[80] *Sass: Syntactically Awesome Style Sheets*. https://sass-lang.com/.

[81] Christian Schlögl. "Information and knowledge management: dimensions and approaches." In: *Information Research: An International Electronic Journal* 10.4 (2005), n4.

[82] Paul G Shekelle, Sally C Morton, and Emmett B Keeler. "Costs and benefits of health information technology". In: *Evidence report/technology assessment* 132 (Apr. 2006), pp. 1–71. ISSN: 1530-4396. DOI: 10.23970/ahrqepcerta132. URL: http://europepmc.org/books/NBK37988.

[83] Mark Siegler. *Confidentiality in medicine—a decrepit concept*. 1982.

[84] Karan Singh et al. *System z Crypto and TKE Update*. IBM Redbooks, 2011.

[85] Mahima Singh and Deepak Garg. "Choosing Best Hashing Strategies and Hash Functions". In: *2009 IEEE International Advance Computing Conference*. 2009, pp. 50–55. DOI: 10.1109/IADCC.2009.4808979.

[86] *Storybook*. https://storybook.js.org/.

[87] Radwan Tahboub and Yousef Saleh. “Data Leakage/Loss Prevention Systems (DLP)”. In: *2014 World Congress on Computer Applications and Information Systems (WCCAIS)*. 2014, pp. 1–6. DOI: 10.1109/WCCAIS.2014.6916624.

[88] Zahir Tari and Shun-Wu Chan. “A role-based access control for intranet security”. In: *IEEE Internet Computing* 1.5 (1997), pp. 24–34.

[89] John J Trinckes Jr. *The definitive guide to complying with the HIPAA/HITECH privacy and security rules*. CRC Press, 2012.

[90] Minne Van der Haak et al. “Data security and protection in cross-institutional electronic patient records”. In: *International journal of medical informatics* 70.2-3 (2003), pp. 117–130.

[91] *Visual Studio Code - Code Editing. Redefined.* https://code.visualstudio.com/.

[92] Karen A Wager, Frances W Lee, and John P Glaser. *Health care information systems: a practical approach for health care management*. John Wiley & Sons, 2021.

[93] Vanessa Wang, Frank Salim, and Peter Moskovits. *The definitive guide to HTML5 WebSocket*. Vol. 1. Springer, 2013.

[94] Etienne Wenger et al. “Communities of practice: Learning as a social system”. In: *Systems thinker* 9.5 (1998), pp. 2–3.

[95] *What is Prisma?* https://www.prisma.io/docs/concepts/overview/what-is-prisma. [Online; accessed 2022-09-09].

[96] R CHESWICK WILLIAM and M BELLOVIN STEVEN. “Firewalls and Internet Security: Repelling the Wily Hacker”. In: *Reading* (1994).

[97] Tom D Wilson et al. “The nonsense of knowledge management”. In: *Information research* 8.1 (2002), pp. 8–1.