WannaHide

A graduation project report submission

In partial fulfillment of the requirements for the award of the degree

Bachelor of Science

Submitted by:

Abdullah Riad	89683
Fouad Mostafa	89638
Abdelrahman Fathy	89393
Mahmoud Mohammed	89401

Under the supervision of Professor:

DR. Rania Elgohary

TA. Eng. Nehal A. Mohamed

TA. Eng. Islam Saied

Department of Computer Science - CS

Misr University for Science and Technology - MUST

College of Computers and Artificial Intelligence Technology – CAIT

June - 2023

ACKNOWLEDGEMENTS

Firstly, we are very grateful to Almighty Allah who gave us opportunity, strength, determination and wisdom to achieve our goal. Without her support this could not have been possible. Though only our names appear on the cover of this dissertation, a great many people have contributed to its production. We owe our gratitude to all those people who have made this dissertation possible and because of whom our graduate experience has been one that we will cherish forever.

Foremost, we would like to express our sincere gratitude to our adviser Prof. Dr. Rania ElGohary for the continuous support of our project study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped us in all the time of research and writing of this research. We appreciate her vast knowledge and skill in many areas, and her assistance in writing reports , which have on occasion made us "GREEN" with envy. Finally, most importantly, none of this would have been possible without the love and patience of our families. Our immediate families to whom this dissertation is dedicated to, has been a constant source of love, concern, support and strength all these years. We would like to express our heart-felt gratitude to our families. So, our extended families have aided and encouraged us throughout this endeavor.

DECLARATION

I hereby certify that this work, which I now submit for assessment on the programme of study leading to the award of Bachelor of Science in computer science is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others and to the extent that such work has been cited and acknowledged within the references section of this report.

Signed: ______________________________

Registration No.: ___________

Date: Day, Month Year.

ABSTRACT

It is generally noticeable that the digital communication whether it involves instant messaging, voice and video calls has become a daily need all over the globe thus protecting any type of data in this type of communication imposed new challenges to the security field which lead to utilizing encryption algorithms to ensure the confidentiality of the data along with its integrity and availability, since ensuring the medium was the first step to achieve that another level of security was added by encrypting the original data into a meaningless cipher using a hybrid encryption-scheme based on AES-256-CBC to secure its content using both symmetric and asymmetric encryption making sure the meaningful image is delivered.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS I

DECLARATION II

ABSTRACT III

Table of Contents IV

list of Figures VIII

list of tables X

Chapter one 1

1 INTRODUCTION 1

1.1 Overview 1

1.2 Chatting 2

1.3 Encryption 2

1.3.1 What is encryption? 2

1.3.2 History of encryption 3

1.3.3 Types of encryption: 4

1.3.4 Critical algorithms in history of encryption: 6

1.4 End-to-end encryption: 7

1.4.1 What is End-to-End Encryption? 7

1.4.2 The First Use of End-to-End Encryption: 7

1.4.3 The Significance of End-to-End Encryption: 7

1.5 Ratchet (Enhancing Security in Messaging with Forward Secrecy) : 8

1.5.1 Who Invented the Ratchet? 8

1.5.2 The First Use of Ratchet: 9

1.5.3 What is Ratchet? 9

1.5.4 what is Double-ratchet? 10

1.6 Diffie-Hellman: 10

1.6.1 Invention and Inventors: 10

1.6.2 First Usage and Early Adoption: 10

1.6.3 Theoretical Overview: 11

1.6.4 Theoretical Performance: 11

1.7 Elgamal algorithm: 12

1.7.1 Invention and Inventor: 12

1.7.2 First Usage and Early Adoption: 12

1.7.3 Elgamal's concept 12

1.7.4 Theoretical Overview: 13

1.7.5 Theoretical Performance: 13

1.8 Problem Definition 14

1.9 Project objectives 15

1.10 Software / Hardware Tools 16

1.10.1 Software Requirements 16

1.10.2 Hardware Requirements 17

1.11 RESEARCH ORGANIZATION 17

CHAPTER TWO 18

2 Related works 18

2.1 Whats App 18

2.1.1 Overview 18

2.1.2 Methodology 18

2.1.3 Advantages 18

2.1.4 Disadvantages 19

2.2 Telegram 19

2.2.1 Overview 19

2.2.2 Methodology 19

2.2.3 Advantages 19

2.2.4 Disadvantages 20

2.3 Signal 20

2.3.1 Overview 20

2.3.2 Methodology 20

2.3.3 Advantages 20

2.3.4 Disadvantages 20

2.4 Application Features 21

Chapter Three 22

3 System Analysis and Design 22

3.1 OVERVIEW 22

3.2 functional and non-functional requirement specification 23

3.2.1 Functional Requirements 23

3.2.2 Non-functional requirements 23

3.3 System Architecture 24

3.4 System Analysis 26

3.4.1 Context Diagram: 26

3.4.2 Data Flow Diagram 27

3.4.3 Use Case Diagram 28

3.4.4 State Diagram 29

3.4.5 Process Flow Diagram 30

3.4.6 Sequence Diagram 31

3.4.7 Class Diagram 32

3.4.8 Entity Diagram 33

CHAPTER FOUR 34

4 PROPOSED SYSTEM 34

4.1 Overview 34

4.2Methodologies 34

4.2.1 End-to-End Encryption 34

4.2.2 ElGamal Algorithm 34

4.2.3 Double Ratchet 35

4.2.4 Key Exchange: 35

4.2.5 Secure Communication 35

4.3 Algorithms 36

4.3.1 ELGAMAL algorithm 36

4.3.2 Double-ratchet Algorithm 37

4.3.3 AES 256-CBC encryption algorithm 39

4.4 Wannahide encryption scheme 41

4.5 PROPOSED APPLICATION 43

4.6 TEAM PROGRESS 46

CHAPTER FIVE 47

5 Results and discussion 47

5.1 Overview 47

5.2 Model flow 47

5.2.1 Choosing algorithm 47

5.2.2 Applying algorithm to specific data categories: 47

5.2.3 Building the entire encryption scheme 48

5.3 application results 48

5.4 Conclusion 48

6 References 49

LIST OF FIGURES

Figure 1 .*Encryption* 2

Figure 2 . *Symmetric Encryption* 4

Figure 3 . *Asymmetric Encryption* 4

Figure 4 . *Hybrid encryption* 5

Figure 5 . *Outline of the Research* 17

Figure 6 . *WhatsApp logo* 18

Figure 7 . *Telegram logo* 19

Figure 8 . *Signal logo* 20

Figure 9 . *System Architecture* 25

Figure 10 . *Context Diagram* 26

Figure 11 . *DFD diagram* 27

Figure 12 . *Use Case Diagram* 28

Figure 13 . *State Diagram* 29

Figure 14 . *Process Flow Diagram* 30

Figure 15 . *Sequence Diagram* 31

Figure 16 . *Class Diagram* 32

Figure 17 . *ERD* 33

Figure 18 . *Double ratchet mechanism* 38

Figure 19 . *Asymmetric Encryption mechanism in Wannahide* 42

Figure 20 . *WannaHide application homepage* 43

Figure 21 . *WannaHide application sign up form* 44

Figure 22 . WannaHide application sign in form 44

Figure 23 . WannaHide application chat page 45

Figure 24 . WannaHide application settings page 45

Figure 25 . Gantt chart 46

LIST OF TABLES

Table 1 . *Comparison between different chat applications* 21

Chapter one

1 INTRODUCTION

1.1 OVERVIEW

The increasing frequency and severity of data breaches in recent years have highlighted the critical importance of security and privacy in today's digital world. From financial institutions to social media platforms, no organization is immune to the risk of data breaches, which can lead to significant financial losses and reputational damage. This has led to a growing demand for secure communication channels, particularly in light of recent privacy scandals involving social media platforms. The need for a secure chat app that prioritizes user privacy and security has become increasingly evident. A secure chat app can provide end-to-end encryption, preventing unauthorized access to conversations, while ensuring the privacy of the users. In addition, such an app can offer features such as anonymous chat and secure file sharing, giving users control over their data and minimizing the risk of data breaches. The development of a secure chat app requires a comprehensive approach that includes risk assessment, threat analysis, and adherence to industry standards and best practices. By prioritizing security and privacy in the design and development of a chat app, individuals and organizations can communicate safely and confidently, without fear of data breaches or unauthorized access.

Furthermore the need for secure communication and collaboration in various contexts, such as business, government, and personal settings was obvious. With the rise of cyber threats and data breaches, there is a growing concern about the privacy and security of sensitive information transmitted through traditional communication channels, such as email and instant messaging. This problem is compounded by the fact that many existing communication tools lack sufficient encryption and security features to protect against these threats such as data leaks and breaches, which can result in the loss or exposure of sensitive or confidential information since it can occur through a variety of channels, including email, file sharing, messaging applications, and cloud storage, and can have serious consequences for individuals and organizations alike, including financial losses, legal liabilities, and damage to reputation.

By providing a practical and effective solution to protect messages transmission that aims to raise awareness of the importance of secure communication it encourages individuals and organizations to adopt more secure messaging tools to protect their sensitive information from unauthorized access and interception.

1.2 CHATTING

Instant messaging has become an integral part of our daily communication, allowing us to easily and quickly exchange messages with friends, family, and colleagues. With the increasing use of web applications, instant messaging has also become an essential feature of web chat applications, enabling users to communicate with each other in real-time. Web chat applications offer a convenient platform for users to communicate with each other from any location, as long as they have access to the internet. Unlike traditional messaging systems, web chat applications allow users to exchange messages in real-time, making communication faster and more efficient.

1.3 ENCRYPTION

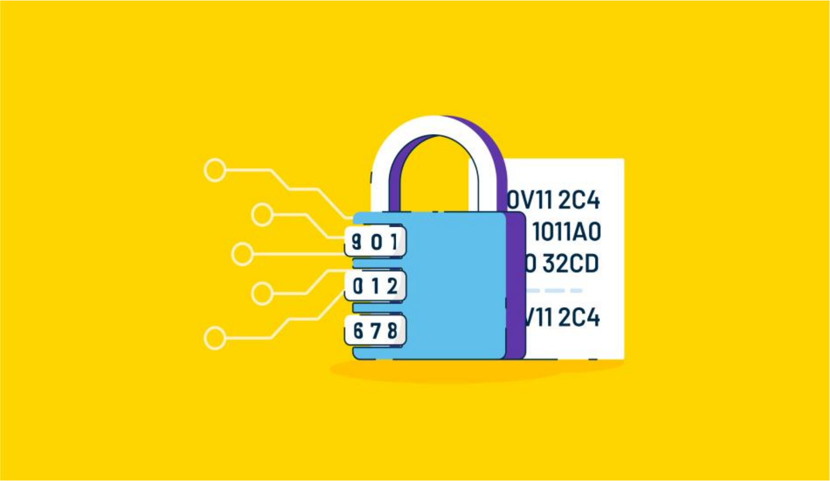


Figure 1.Encryption

1.3.1 What is encryption?

Encryption is a process of transforming plain-text (readable data) into cipher-text (encrypted data) using mathematical algorithms and keys. The primary objective of encryption is to protect the confidentiality and integrity of sensitive information, preventing unauthorized access or tampering. Encryption has a long history, dating back thousands of years, and has evolved significantly over time.

1.3.2 History of encryption

The earliest known use of encryption can be traced back to ancient civilizations. The ancient Egyptians used simple substitution ciphers around 1900 BCE, where each letter was replaced by another letter or symbol to obscure the original message. Similarly, the ancient Greeks employed encryption techniques such as the Spartan scytale, a cylinder with a strip of parchment wrapped around it, which served as a transposition cipher.

The Romans also made use of encryption during warfare. Julius Caesar, for instance, employed a substitution cipher known as the Caesar cipher, where each letter in the plaintext was shifted a fixed number of positions down the alphabet. Despite its simplicity, the Caesar cipher provided a degree of security against casual eavesdroppers.

Throughout history, encryption played a vital role in military and diplomatic communications. Notable examples include the use of encryption during the Renaissance, such as the Vigenère cipher invented by Giovan Battista Bellaso in the 16th century. The Vigenère cipher employed a keyword to determine the shifting pattern for each letter, making it more secure than earlier ciphers.

In the modern era, encryption has become increasingly sophisticated due to advancements in mathematics, computer science, and technology. In the 1970s, the Data Encryption Standard (DES) was developed by IBM and adopted by the U.S. government as a widely-used encryption standard. DES employed a symmetric key algorithm, using a 56-bit key to encrypt and decrypt data.

As technology advanced and the need for stronger encryption grew, the DES algorithm was eventually replaced by the Advanced Encryption Standard (AES) in 2001. AES utilizes symmetric key cryptography and supports key sizes of 128, 192, and 256 bits. It has become the de facto encryption standard for a wide range of applications.

Another significant development in encryption is the advent of asymmetric encryption or public-key cryptography. In 1976, Whitfield Diffie and Martin Hellman introduced the concept of public-key encryption, which allowed for secure communication without the need to share a secret key. The RSA algorithm, developed by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977, is one of the most well-known and widely used asymmetric encryption algorithms.

1.3.3 Types of encryption:

- Symmetric Encryption :

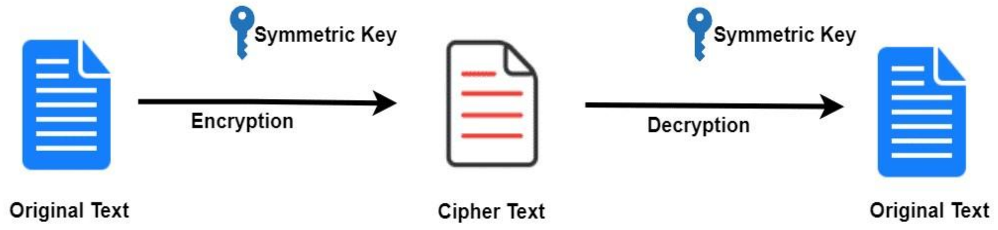


Figure 2. Symmetric Encryption

Also known as secret-key encryption, employs a single key for both encryption and decryption. Both the sender and recipient must possess and exchange the same secret key securely. Common symmetric encryption algorithms include Advanced Encryption Standard (AES) and Data Encryption Standard (DES).[1]

- Asymmetric Encryption :

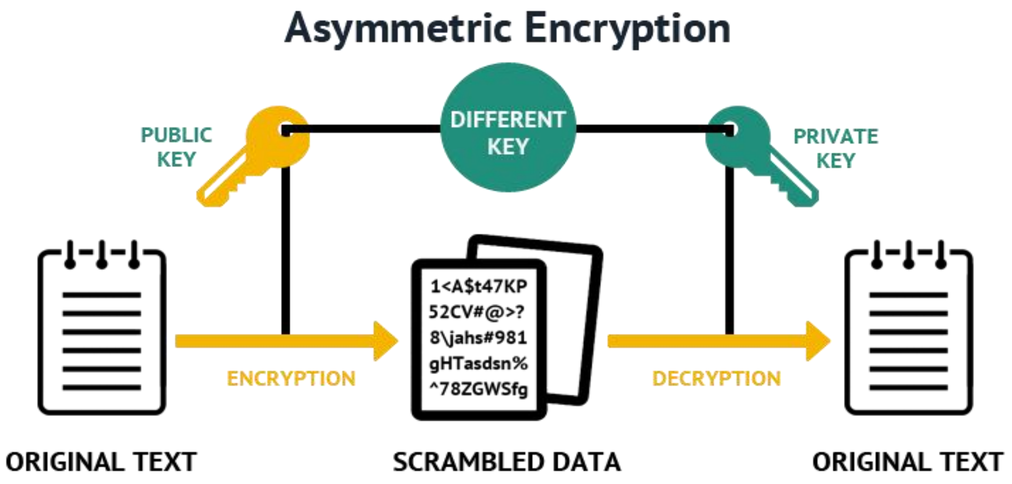


Figure 3. Asymmetric Encryption

Also referred to as public-key encryption, utilizes a pair of mathematically related keys: a public key for encryption and a private key for decryption. The public key can be openly shared, while the private key must be kept secret. Popular asymmetric encryption algorithms include RSA and Elliptic Curve Cryptography (ECC).[1]

- Hybrid encryption :

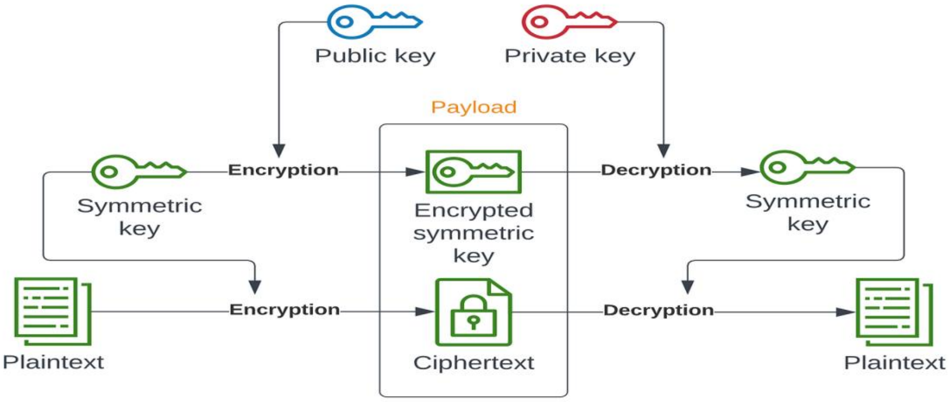


Figure 4. Hybrid encryption

It is a cryptographic technique that combines the strengths of symmetric and asymmetric encryption algorithms to provide a secure and efficient solution for data protection. It addresses the limitations of each encryption method by leveraging their unique characteristics, offering a practical approach to secure communication and data privacy in various domains. Hybrid encryption seeks to overcome these limitations by combining the benefits of both symmetric and asymmetric encryption techniques. It achieves this by utilizing a two-step process: first, a symmetric encryption algorithm is employed to encrypt the actual data, and then an asymmetric encryption algorithm is used to protect the symmetric key used for encryption. This approach allows for the secure exchange of the symmetric key using the public key infrastructure, while leveraging the efficiency of symmetric encryption for bulk data encryption.The concept of hybrid encryption can be visualized as a secure "envelope" for data transmission. The symmetric encryption algorithm creates a secure and efficient channel to protect the content of the envelope, while the asymmetric encryption algorithm safeguards the symmetric key that unlocks the envelope. This combination ensures both confidentiality and data integrity during transit, addressing the key exchange vulnerability of symmetric encryption. The concept of hybrid encryption can be visualized as a secure "envelope" for data transmission. The symmetric encryption algorithm creates a secure and efficient channel to protect the content of the envelope, while the asymmetric encryption algorithm safeguards the symmetric key that unlocks the envelope. This combination ensures both confidentiality and data integrity during transit, addressing the key exchange vulnerability of symmetric encryption.[1]

1.3.4 Critical algorithms in history of encryption:

- Development of Public-Key Cryptography: The invention of public-key cryptography by Whitfield Diffie and Martin Hellman in 1976 revolutionized encryption. Public-key cryptography introduced the concept of asymmetric encryption, where two mathematically related keys, a public key and a private key, are used for encryption and decryption. This breakthrough allowed for secure communication without the need to share a secret key, paving the way for more advanced encryption methods.[2]

- Data Encryption Standard (DES): The development of the Data Encryption Standard (DES) in the 1970s was a significant milestone in encryption history. DES, designed by IBM, became the first widely used encryption standard. It employed a symmetric key algorithm and a 56-bit key size. While DES eventually became vulnerable to brute-force attacks due to increasing computational power, its adoption laid the foundation for subsequent encryption standards.[2]

- Advanced Encryption Standard (AES): It was selected by the National Institute of Standards and Technology (NIST) in 2001 to replace DES as the encryption standard. AES, with its symmetric key algorithm, supports key sizes of 128, 192, and 256 bits, providing significantly stronger security. AES has become the de facto standard for encryption and is widely used in various applications and industries.[2]

- Diffie-Hellman Key Exchange: It introduced by Whitfield Diffie and Martin Hellman in 1976, revolutionized secure key exchange between parties over an insecure communication channel. It allows two parties to establish a shared secret key over a public channel, enabling secure communication without prior sharing of a secret key. The Diffie-Hellman key exchange became a foundational method in secure communication protocols.[2]

- Rivest-Shamir-Adleman (RSA) Encryption: developed by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977, is a widely used public-key encryption method. RSA is based on the computational difficulty of factoring large prime numbers. It enables secure key exchange, digital signatures, and confidentiality. RSA remains a vital encryption algorithm and forms the basis for various cryptographic protocols.[2]

- Elliptic Curve Cryptography (ECC): It is a modern asymmetric encryption technique that provides the same level of security as traditional algorithms but with smaller key sizes. ECC is based on the mathematics of elliptic curves, offering strong encryption while reducing computational requirements and bandwidth usage. ECC has gained popularity in recent years, especially in resource-constrained environments such as mobile devices and IOT devices.[2]
- Quantum-resistant Encryption:The advent of quantum computers poses a potential threat to traditional encryption methods. To address this challenge, research is being conducted to develop quantum-resistant encryption algorithms. These algorithms aim to withstand attacks from quantum computers by leveraging mathematical problems that are computationally hard for both classical and quantum computers. Post-Quantum Cryptography (PQC) is an active area of research to ensure encryption remains secure in the face of quantum computing advancements.[2]

1.4 END-TO-END ENCRYPTION:

1.4.1 What is End-to-End Encryption?

Intended recipients of a communication can access its content. It involves encrypting data at its source (sender) and decrypting it at its destination (receiver), with the encryption keys being exclusively in the hands of the communicating parties. This way, intermediaries and eavesdroppers, including service providers and hackers, are unable to access or decipher the encrypted End-to-end encryption (E2EE) is a security measure that ensures that only the data.[3][4]

1.4.2 The First Use of End-to-End Encryption:

End-to-end encryption was first introduced and popularized in the 1990s with the advent of Pretty Good Privacy (PGP). Phil Zimmermann developed PGP as an encryption software that enabled individuals to secure their email communications. PGP utilized a combination of symmetric and asymmetric encryption to provide end-to-end security for messages.[4]

1.4.3 The Significance of End-to-End Encryption:

There are several compelling reasons why end-to-end encryption is widely used and valued in various domains:

- Privacy Protection: End-to-end encryption ensures that only the sender and intended recipient can access the content of a message or data. It prevents unauthorized access, surveillance, and data breaches by hackers, governments, or even service providers. By protecting individual privacy, end-to-end encryption fosters trust and empowers users to have control over their own information.[5][6]

- Mitigating Data Interception: With end-to-end encryption, data transmitted between users is encrypted at the source and can only be decrypted by the intended recipient. This prevents interception and tampering of data by malicious actors during transit, as the encrypted data is meaningless without the proper decryption keys.[6]

- Securing Cloud Storage: End-to-end encryption is crucial for safeguarding data stored in the cloud. By encrypting data on the client-side before uploading it to the cloud service, the data remains confidential even if the cloud provider's servers are compromised.[6]

- Protection against Service Provider Vulnerabilities: End-to-end encryption mitigates the risks posed by vulnerabilities or breaches within service providers' systems. Even if a service provider's servers are compromised or if an insider attack occurs, the encrypted data remains unreadable without the decryption keys held only by the communicating parties.[7][8]

- Ensuring Confidentiality in Communication: End-to-end encryption is particularly valuable in messaging applications, ensuring that conversations remain private and secure. It prevents unauthorized access to message content, including text, images, voice recordings, and attachments.[7][8][9]

1.5 RATCHET (ENHANCING SECURITY IN MESSAGING WITH FORWARD SECRECY) :

1.5.1 Who Invented the Ratchet?

The concept of the ratchet was pioneered by Trevor Perrin, Moxie Marlinspike, and others during the development of the Signal Protocol. The Signal Protocol is widely adopted as the backbone of end-to-end encryption in popular messaging apps like Signal, What's App, and Facebook Messenger.[10]

1.5.2 The First Use of Ratchet:

The Signal Protocol, incorporating the ratchet mechanism, was first introduced in the Signal messaging app (formerly known as Text Secure) in 2013. Signal, known for its emphasis on privacy and security, was one of the first messaging platforms to employ ratchet-based encryption.[10][11]

1.5.3 What is Ratchet?

The ratchet mechanism operates based on several theoretical principles, enabling enhanced security in messaging applications. Here's an overview of its theoretical performance:

- Perfect Forward Secrecy (PFS):

The ratchet provides Perfect Forward Secrecy, which ensures that the compromise of a long-term encryption key does not expose the confidentiality of past messages. With the ratchet, ephemeral keys are regularly updated, and each new key is used to encrypt a subset of messages. This way, if one key is compromised, it only affects a limited portion of the conversation, preserving the security of other messages.[12]

- Diffie-Hellman Key Exchange:

The ratchet leverages the Diffie-Hellman key exchange protocol, allowing two parties to establish a shared secret key over an insecure channel. This shared secret is used to derive session keys for encryption and decryption. The ratchet ensures that these session keys are updated regularly, limiting the impact of potential key compromise.[12]

- Key Derivation and Updating:

When a message is sent, the ratchet mechanism derives a new encryption key from the previous key. This derivation process utilizes cryptographic functions to generate a fresh key, ensuring that future messages are encrypted with different keys from previous messages. This key updating mechanism enhances security and prevents an attacker from gaining access to multiple message keys even if one key is compromised.[12]

- Key Expiration and Deletion:

To further enhance security, the ratchet may include a key expiration or deletion mechanism. This ensures that older keys are no longer used after a certain period or a specific number of messages. By expiring or deleting older keys, the exposure of messages is limited, even if a key is compromised.[12]

1.5.4 what is Double-ratchet?

The double ratchet end-to-end encryption protocol is a cryptographic technique used to secure communication between two parties, such as in messaging applications. The protocol utilizes the double ratchet model to provide forward secrecy and message integrity.[12]

In this protocol, both parties generate a pair of public and private keys. The public key is shared between the parties and used for encryption, while the private key is kept secret and used for decryption. The parties also generate a shared secret key that is used to encrypt and decrypt messages.[12]

The double ratchet model is then used to establish a session key that is used to encrypt messages. The first ratchet is used to generate a new key for each message, while the second ratchet is used to update the shared secret key.[12]

1.6 DIFFIE-HELLMAN:

1.6.1 Invention and Inventors:

The Diffie-Hellman algorithm was invented by Whitfield Diffie, an American cryptographer, and Martin Hellman, an American electrical engineer. In 1976, Diffie and Hellman published their seminal paper titled "New Directions in Cryptography," which introduced the concept of public-key cryptography and the Diffie-Hellman key exchange algorithm. Their innovative work challenged the traditional notions of secure key exchange and sparked a new era in cryptographic research.[14][15]

1.6.2 First Usage and Early Adoption:

Although the Diffie-Hellman algorithm was groundbreaking, its initial implementation faced challenges due to limited computational resources at the time. However, the algorithm's practical usage began to gain momentum in the 1990s with the advent of more powerful computers. It quickly became a fundamental component in various cryptographic protocols, including secure communication protocols, digital signatures, and secure key establishment.[14][15]

1.6.3 Theoretical Overview:

The Diffie-Hellman algorithm is based on the concept of modular exponentiation within finite cyclic groups. The algorithm allows two parties, traditionally referred to as Alice and Bob, to agree on a shared secret key over an insecure communication channel. The key exchange is achieved without directly transmitting the secret key itself, providing a secure method for key establishment.

Key Generation: Alice and Bob agree on a common public domain parameters, which include a large prime number and a primitive root modulo the prime. Each party generates a private key and calculates the corresponding public key using modular exponentiation.

Key Exchange: Alice and Bob exchange their public keys over the insecure channel. Using their own private keys and the received public keys, they independently calculate a shared secret key using modular exponentiation.

Shared Key: The shared secret key obtained by both Alice and Bob is identical, allowing them to securely communicate using symmetric encryption algorithms.[16][17]

1.6.4 Theoretical Performance:

The Diffie-Hellman algorithm's security is based on the computational complexity of the discrete logarithm problem. Breaking the algorithm requires solving the discrete logarithm problem, which is considered computationally infeasible for large prime numbers and well-chosen parameters. The strength of the algorithm lies in the difficulty of deriving the private key from the public key.

From a theoretical perspective, the Diffie-Hellman algorithm provides a secure method for key exchange. However, its performance can be influenced by factors such as the size of the prime number and the efficiency of modular exponentiation algorithms. The algorithm's computational complexity increases with larger prime numbers, potentially impacting the execution time. Various optimizations and improvements have been developed to enhance the algorithm's performance, including the use of faster modular exponentiation algorithms and elliptic curve-based variations of Diffie-Hellman.[16][17]

1.7 ELGAMAL ALGORITHM:

1.7.1 Invention and Inventor:

The ElGamal algorithm was invented by Taher Elgamal, an Egyptian cryptographer and computer scientist. Elgamal introduced the algorithm in 1984 while working at the Stanford Research Institute. His innovative approach to public-key encryption laid the foundation for secure communication in the digital age.[18][19]

1.7.2 First Usage and Early Adoption:

The ElGamal algorithm quickly gained recognition within the cryptographic community for its effectiveness in securing sensitive information. Although its initial implementation faced some challenges, it became widely adopted in the early 1990s. The algorithm's first major usage was in the Pretty Good Privacy (PGP) software, an encryption program developed by Phil Zimmermann, which allowed users to send encrypted messages securely over email.[18][19]

1.7.3 Elgamal's concept

ElGamal algorithm is based on the Diffie-Hellman algorithm. In fact, the ElGamal algorithm can be seen as an extension or variation of the Diffie-Hellman key exchange protocol, while Both algorithms share a similar foundation in the concept of modular exponentiation within finite cyclic groups, they rely on the computational complexity of the discrete logarithm problem to provide secure key exchange or encryption.

The Diffie-Hellman algorithm focuses specifically on key exchange, allowing two parties to establish a shared secret key over an insecure channel. On the other hand, the ElGamal algorithm extends this concept to encryption by incorporating the principles of public-key cryptography.

In the ElGamal algorithm, the sender encrypts a message using the recipient's public key, and the recipient decrypts the message using their private key. The encryption process involves modular exponentiation and modular multiplication operations, similar to the Diffie-Hellman algorithm.

While both algorithms are based on similar mathematical principles and the discrete logarithm problem, the ElGamal algorithm introduces additional steps to enable encryption and decryption. By incorporating public-key cryptography, ElGamal provides a secure method for encrypting and decrypting messages while ensuring confidentiality and authenticity, therefore, one can consider the ElGamal algorithm as an extension of the Diffie-Hellman algorithm, where the key exchange concept is expanded to include encryption and decryption functionalities.[20][21]

1.7.4 Theoretical Overview:

The ElGamal algorithm is based on the computational complexity of solving the discrete logarithm problem, which forms the core of its security. The algorithm operates in a multiplicative group of integers modulo a prime number. It employs mathematical operations such as modular exponentiation and modular multiplication to encrypt and decrypt messages.

- Key Generation: The algorithm involves generating a public-private key pair. The public key is derived from the private key and is used for encryption, while the private key remains secret and is used for decryption.
- Encryption: To encrypt a message, the sender selects a random value, known as a session key, and performs modular exponentiation on it using the recipient's public key. This operation produces the ciphertext, which is then combined with the plaintext message.
- Decryption: The recipient, possessing the corresponding private key, can decrypt the ciphertext by performing modular exponentiation on the received data. This process retrieves the original plaintext message.[21]

1.7.5 Theoretical Performance:

From a theoretical standpoint, the ElGamal algorithm offers strong security guarantees. Its strength lies in the computational complexity of solving the discrete logarithm problem, which is considered computationally infeasible for large prime numbers. The security of the algorithm relies on the assumption that breaking the encryption requires solving this mathematical problem efficiently.

However, it's important to note that the ElGamal algorithm is generally slower compared to symmetric encryption algorithms due to its reliance on complex mathematical operations. The performance of the algorithm can be affected by factors such as key size, choice of prime numbers, and the efficiency of modular exponentiation algorithms. Efforts have been made to optimize the algorithm's performance by implementing faster algorithms for modular exponentiation.[22]

1.8 PROBLEM DEFINITION

Chatting became an essential part of our life whether if its work, family or friends related, through multiple applications sending photos and text messages without caring - for most users - about the security of the application used or the privacy of their data if it's at risk of being leaked or hacked by an attacker in some specific method solving that problem will take a place by designing and developing a chat web application that utilizes end-to-end encryption to ensure secure and private communication between users. The problem we aim to address is the lack of privacy and security in online communication, which can lead to the unauthorized access of personal information and sensitive data.

Existing chat applications utilize various security measures, such as Transport Layer Security (TLS), to protect the transmission of data between users. However, these security measures only protect the data in transit and do not ensure that the messages exchanged are protected from unauthorized access by the service providers or other intermediaries.

End-to-end encryption is a cryptographic technique that provides a solution to this problem. It ensures that the messages exchanged between users are encrypted on the sender's device and can only be decrypted by the intended recipient, with no intermediaries having access to the unencrypted message.

The challenge in implementing end-to-end encryption in a chat web application lies in ensuring that the encryption and decryption process is seamless and transparent to the users, while also providing a user-friendly interface and maintaining the performance and scalability of the application.

Therefore, the objective of this project is to design and develop a chat web application that utilizes end-to-end encryption to provide secure and private communication between users, while also ensuring that the application is user-friendly, performant, and scalable. The project will involve researching and implementing suitable cryptographic algorithms, designing and building a user interface that supports end-to-end encryption, and testing the application's security and performance.

1.9 PROJECT OBJECTIVES

The main objective of this graduation project is to design and develop a chat web application that utilizes end-to-end encryption to ensure secure and private communication between users. To achieve this goal, the following specific objectives will be pursued:

- Research and select appropriate cryptographic algorithms: The project will involve researching and selecting cryptographic algorithms suitable for implementing end-to-end encryption in the chat web application. This will include evaluating the security and performance of different algorithms and selecting the most suitable ones.

- Design and build a user interface that supports end-to-end encryption: The project will involve designing and building a user interface that supports end-to-end encryption, while also providing a user-friendly experience for users. The interface should include features such as message encryption and decryption and user authentication.

- Implement end-to-end encryption in the chat web application: The project will involve implementing end-to-end encryption in the chat web application, using the selected cryptographic algorithms. The encryption and decryption process should be seamless and transparent to the users, and should not affect the performance or scalability of the application.

- Test the security and performance of the chat web application: The project will involve testing the security and performance of the chat web application, to ensure that it is secure and performs well under different conditions. This will include testing its performance under high traffic and load conditions.

- Evaluate the effectiveness of the chat web application: The project will involve evaluating the effectiveness of the chat web application in providing secure and private communication between users.

1.10 SOFTWARE / HARDWARE TOOLS

1.10.1 Software Requirements

- MongoDB stores data in documents, which are JSON like structures that can have nested fields and arrays. It also uses a flexible schema, allowing data to be added or removed without the need for predefined tables and columns. One of the key features of MongoDB is that it supports horizontal scaling through sharing, which is the process of splitting data across multiple servers or shards.
- IndexedDB: is an API (Application Programming Interface) that provides a way for web applications to store and retrieve large amounts of structured data, such as JSON objects or binary data, on the client-side. It is a NoSQL, transactional database that is native to modern web browsers, such as Chrome, Firefox, and Edge. IndexedDB is a key-value store, where each data item is associated with a unique key. It provides a way to store and retrieve data asynchronously, which means that data can be accessed and manipulated without blocking the main thread of the web application. This is particularly useful for web applications that need to work with large datasets or perform complex data operations. IndexedDB is widely used in web applications that require client-side storage of large datasets, such as email clients, task managers, and note-taking applications. It provides a powerful and flexible way to store and retrieve data on the client-side, without the need for a server-side database.[26][27]
- Node.js: It is a cross-platform, open-source server environment that can run on Windows, Linux, Unix, macOS, and more. Node.js is a back-end JavaScript runtime environment, runs on the V8 JavaScript Engine, and executes JavaScript code outside a web browser.[25]
- Web browser: A web browser is an application for accessing websites. When a user requests a web page from a particular website, the browser retrieves its files from a web server and then displays the page on the user's screen.
- Visual studio code: Also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

1.10.2 Hardware Requirements

There is no hardware required for this project.

1.11 RESEARCH ORGANIZATION

This research consists of 5 chapters:

- Chapter one presents an introduction and background of security, encryption, some of the most common algorithms used and their history.
- Chapter two introduces a review of related works that have been implemented before and a comparative study between them.
- Chapter three explained all requirements such as user requirements, system requirements, functional and non-functional requirements, …etc., and all your UML diagrams such as state diagram, frequency diagram, use case diagram, …etc.
- Chapter four describes in depth how every aspect of the project was done, compiled, or created. This includes system methodology, all algorithms used, and implemented systems functions.
- Chapter five contains summarized and elaborated results of the project, in addition to functional evaluation and discussion about the business value of the project.

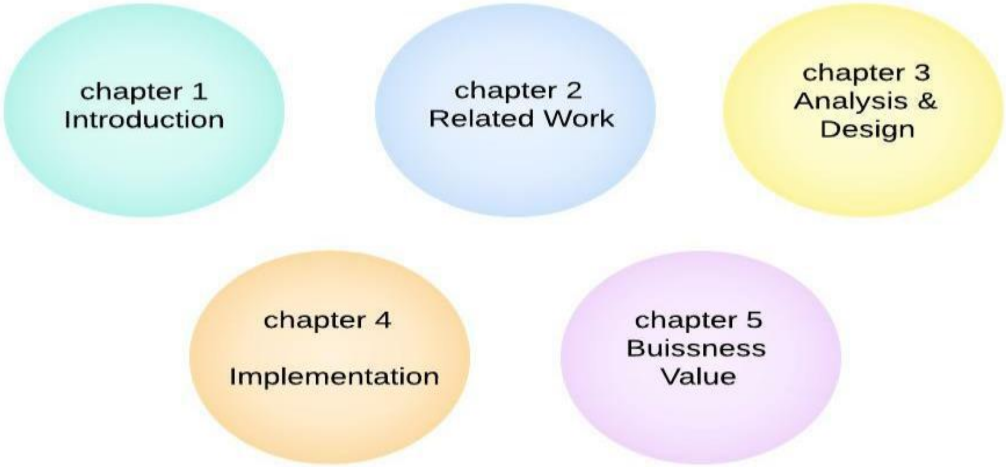


Figure 5. Outline of the Research

CHAPTER TWO

2 RELATED WORKS

The WannaHide app aims to provide users with a secure and private chat experience by leveraging advanced encryption algorithms. To better understand its significance in the landscape of secure chat applications, it is important to explore related works in the field. This article provides an overview of popular secure chat applications, including WhatsApp, Telegram, Signal, and Messenger. It examines their methodologies, advantages, and disadvantages in comparison to the WannaHide app.

2.1 WHATS APP

Figure 6. WhatsApp logo

2.1.1 Overview

WhatsApp is a widely used chat application available on various platforms. It offers end-to-end encryption for messages, voice calls, and video calls, ensuring that only the intended recipients can access the content. [32]

2.1.2 Methodology

WhatsApp employs the Signal Protocol for secure communication. It uses the Signal Protocol's double ratchet algorithm for message encryption and implements secure key exchange through the Diffie-Hellman key agreement protocol. [32]

2.1.3 Advantages

- WhatsApp benefits from a large user base
- cross-platform availability

- seamless integration with phone contacts.
- offers features like group chats, multimedia sharing, and voice/video calls.

2.1.4 Disadvantages

- WhatsApp provides strong encryption.
- It is owned by Facebook, raising concerns about data privacy and potential data sharing with the parent company.

2.2 TELEGRAM

Figure 7. Telegram logo

2.2.1 Overview

Telegram is a cloud-based chat application that emphasizes speed and security. It offers end-to-end encryption for secret chats and incorporates self-destructing messages for enhanced privacy. [31]

2.2.2 Methodology

Telegram uses a proprietary encryption protocol called MTProto, which combines symmetric and asymmetric encryption algorithms. It also allows users to verify encryption keys and offers features like cloud storage for media files.[31]

2.2.3 Advantages

- User-friendly interface, fast message delivery, and extensive customization options.
- Features self-destructing messages and cloud storage for media files.

2.2.4 Disadvantages

- Default mode lacks end-to-end encryption for all chats.
- Security auditability and protocol implementation concerns raised by some experts.

2.3 SIGNAL

Figure 8. Signal logo

2.3.1 Overview

Signal is an open-source secure chat application that places a strong emphasis on privacy and security. It is known for its commitment to protecting user data and maintaining minimal data collection. [28]

2.3.2 Methodology

Signal utilizes the Signal Protocol, which provides end-to-end encryption for all communications, including text messages, voice calls, and video calls. It employs the double ratchet algorithm for secure key exchange and encrypted messaging. [30]

2.3.3 Advantages

- Signal offers robust security measures, including forward secrecy, secure group chats, and the ability to verify encryption keys.
- It places a strong focus on user privacy, with minimal data collection and a transparent approach to security.

2.3.4 Disadvantages

- Signal's user base may be smaller compared to other chat applications, which can limit the network effect and the number of contacts using the platform.

2.4 APPLICATION FEATURES

Features	Wannahide	WhatsApp	Signal	Telegram
Trust-On-First-Use	X	✓	✓	X
End-to-end encryption	✓	✓	✓	✓
Account verification	✓	✓	✓	✓
Edit profile	✓	✓	✓	✓
Notification about key changes	X	X	✓	X
Blocking messages	X	✓	✓	✓
Changing chain key in message transmission	✓	✓	✓	X
Type of Hybrid encryption	Elgamal with double ratchet	Diffie-Hellman with double ratchet	Diffie-Hellman with double ratchet	MTProto Protocol
Price	Free	Free	Free	Free
Database	MongoDB	MySQL or PostgreSQL	SQLite	TDLib

Table 1. Comparison between different chat applications

Chapter Three

3 SYSTEM ANALYSIS AND DESIGN

3.1 OVERVIEW

System analysis and design is the process of understanding, planning, and designing a software system. This process involves several steps that help developers to create a high-quality and efficient system that meets the user's needs. Here are the six steps involved in system analysis and design:

- Requirements gathering: This involves gathering information about the system's purpose, user requirements, and constraints. This step involves interviews with stakeholders, surveys, and user feedback.
- System analysis: This step involves analyzing the gathered requirements to identify the system's components, processes, and data flows. This step also involves identifying any potential problems or risks that may arise during development.
- System design: This step involves designing the system's architecture, data structures, and user interfaces. This step also involves selecting the appropriate programming language, framework, and database system.
- Implementation: This step involves coding the system based on the design specifications. This step also involves testing the system to ensure it meets the required functionality.
- Testing: This step involves testing the system to ensure it meets the required functionality and performance. Testing includes unit testing, integration testing, and system testing.
- Maintenance: This step involves maintaining the system after it's deployed. This step includes fixing any bugs, updating the system, and adding new features.[34][35][36]

3.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENT SPECIFICATION

3.2.1 Functional Requirements

Functional requirements describe the specific features and capabilities that a software system should have to meet the needs of its users. These requirements are typically defined through a process of gathering and analyzing user needs and expectations. Here are some examples of functional requirements:

- User registration and authentication: The chat app should allow users to register for an account and authenticate themselves using a username and password.
- Chat: The chat app should allow users to send and receive messages to other users one-on-one, in real-time.
- Message delivery and read receipts: The chat app should provide notifications to users when messages are delivered and read by the recipient.
- Media sharing: The chat app should allow users to share images with their friends.
- User profile: The chat app should allow users to create and customize their profiles with information and photos.
- Add Friends: The chat app should allow users allow users to add friends using their username.

3.2.2 Non-functional requirements

Non-functional requirements describe the characteristics of a software system that are not directly related to its functionality, but are still critical to its overall success. These requirements typically specify how the system should perform, how it should be designed, and how it should be maintained over time. Here are some examples of non-functional requirements:

- Performance: The system should be fast and responsive, with minimal lag when performing tasks or loading data.
- Usability: The system should be easy to use and navigate, with a clean and intuitive user interface.

- Security: The system should use encryption and other security measures to protect user data and prevent unauthorized access.
- Reliability: The system should be stable and reliable, with minimal downtime and errors.
- Scalability: The system should be able to handle a large number of users and data without slowing down or crashing.
- Maintainability: The system should be easy to maintain and update over time, with well-documented code and clear instructions for making changes.

3.3 SYSTEM ARCHITECTURE

The systems architecture process is where the concepts that will be the backbone of the actual system are developed. It is a conceptual model that describes the structure and behavior of the proposed system or of an existing system . The model could include the technical framework, end user requirements, and a list of system components (hardware and software).

The key decisions that need to be made during the systems architecture process are:

- The attributes of the new system
- The style of architecture
- Type of software used (custom or off-the-shelf)
- Types of technologies used.
- How the system will be deployed

At this point in the systems engineering life cycle, an operational need has been expressed and turned into a concept and set of operational requirements (refer to “Concept Development” topic).

They are then analyzed and transformed into a set of system requirements. The next step is to develop an architecture (or update an existing architecture for fielded systems) as a basis or foundation to guide design and development.[35][36]

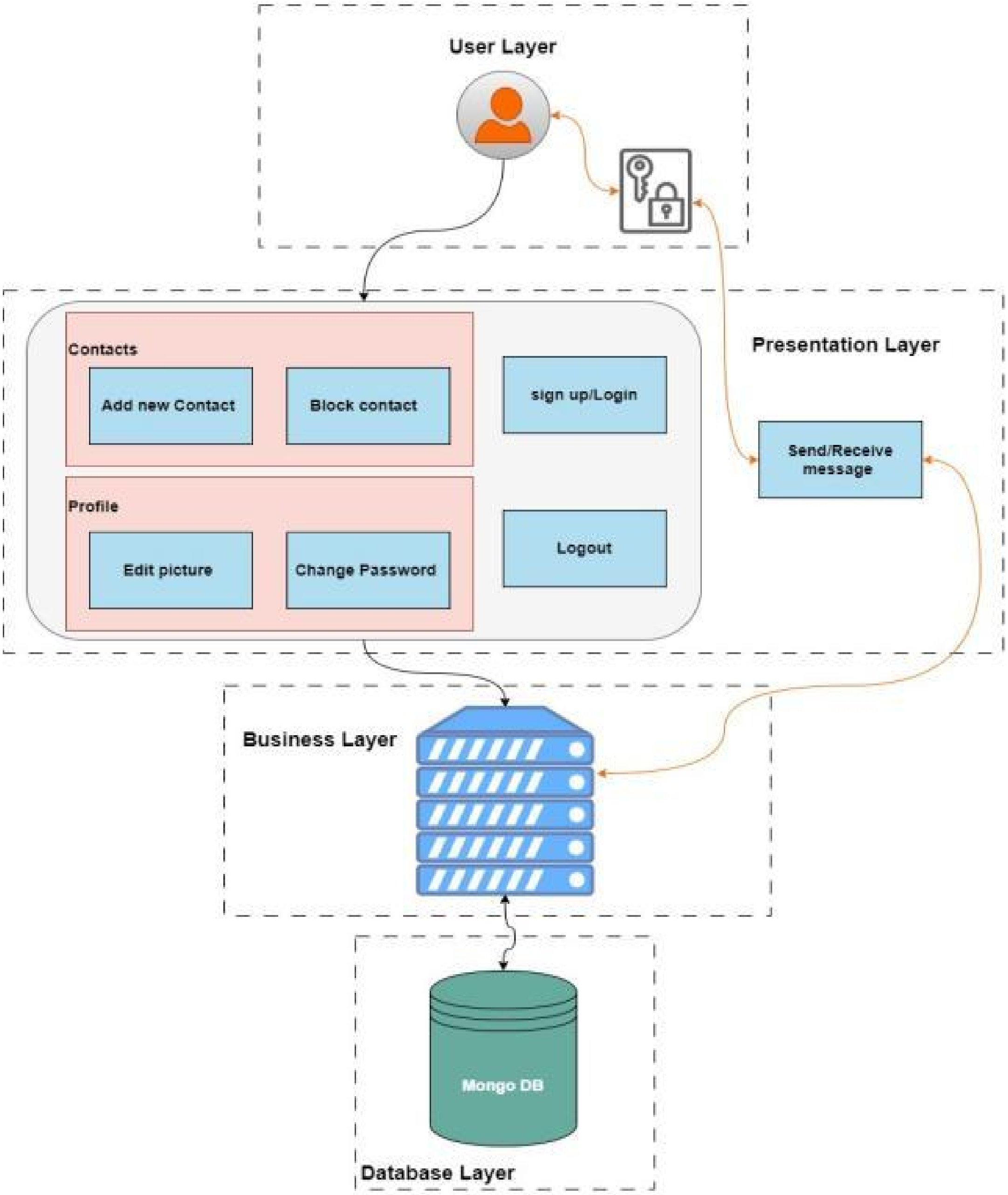


Figure 9. System Architecture

3.4 SYSTEM ANALYSIS

3.4.1 Context Diagram:

A context diagram is a visual representation of a system that shows the system's boundaries, external entities, and the interactions between them. It is a high-level diagram that provides an overview of the system and its relationship with other systems or entities.

The purpose of a context diagram is to provide a clear and concise view of the system or process and its interactions with external entities. It helps to identify the scope and boundaries of the system or process, and to ensure that all stakeholders have a shared understanding of the system's purpose and functionality.[36]

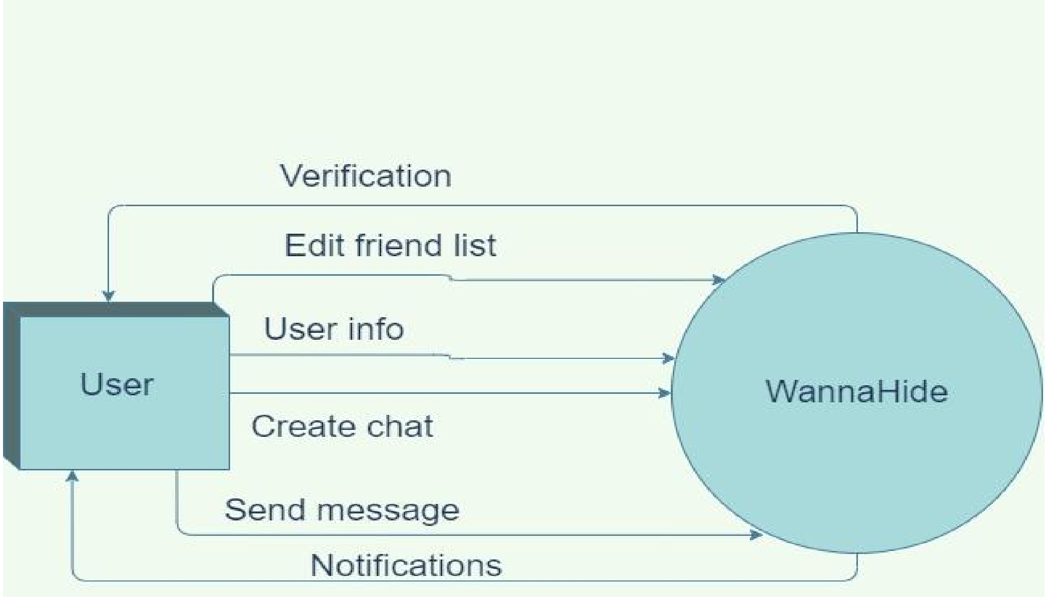


Figure 10. Context Diagram

3.4.2 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of a system or process that shows how data flows through various processes and data stores. It is used to model the flow of data in a system, from input to processing to output.

The purpose of a DFD is to provide a clear and concise representation of the system's overall data flow. It helps to identify potential areas of improvement or optimization in the system's design and operation.[36]

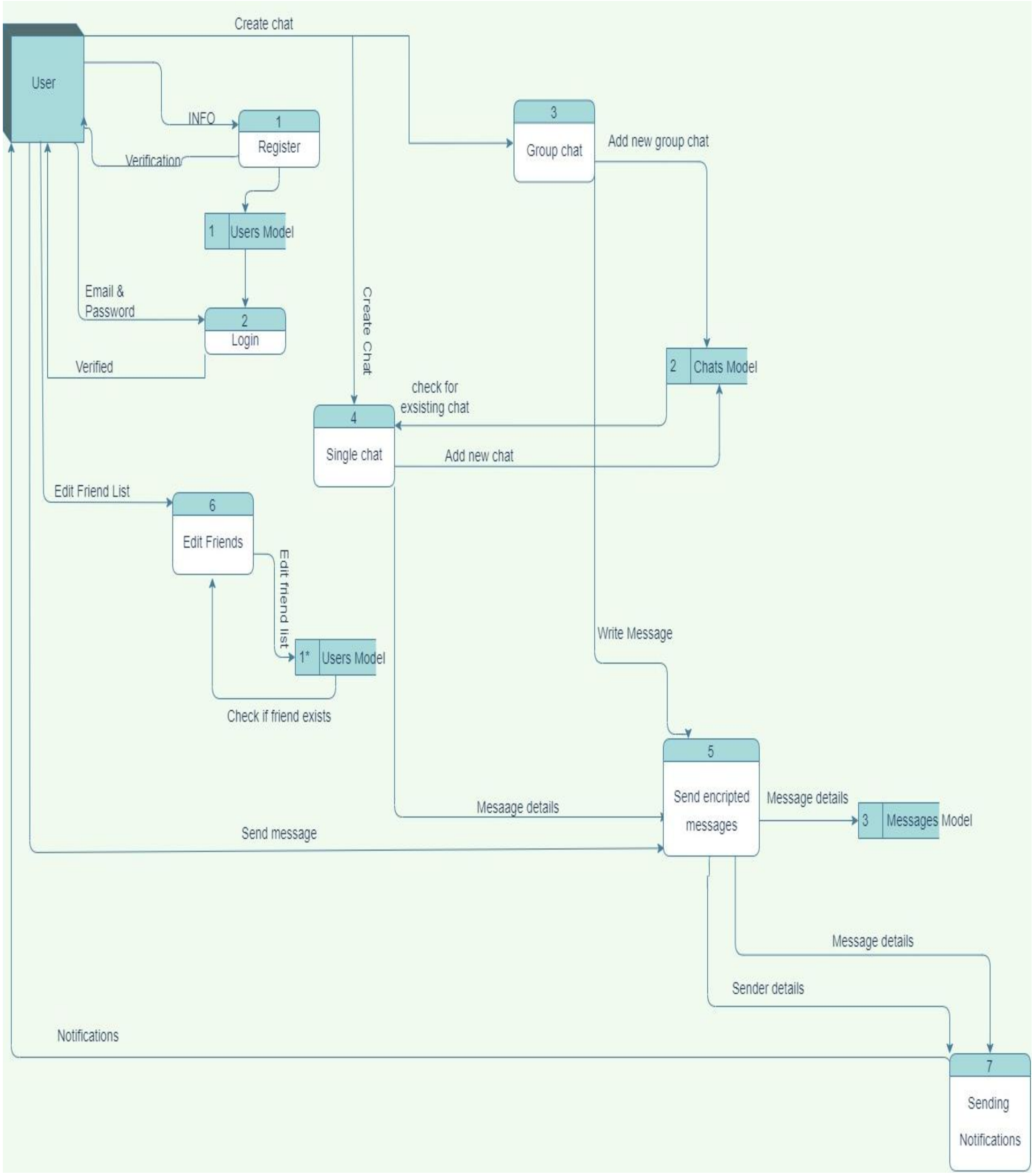


Figure 11. DFD diagram

3.4.3 Use Case Diagram

A use case diagram is a type of behavioral diagram in the Unified Modeling Language (UML) that represents the interactions between actors and a system or software application. It is used to illustrate the user's goals and requirements from the system or application.

The purpose of a use case diagram is to provide a high-level view of the system's functionality and the interactions between the user and the system. It is used to identify the user's goals, requirements, and expectations, and to ensure that the system meets these requirements.[36]

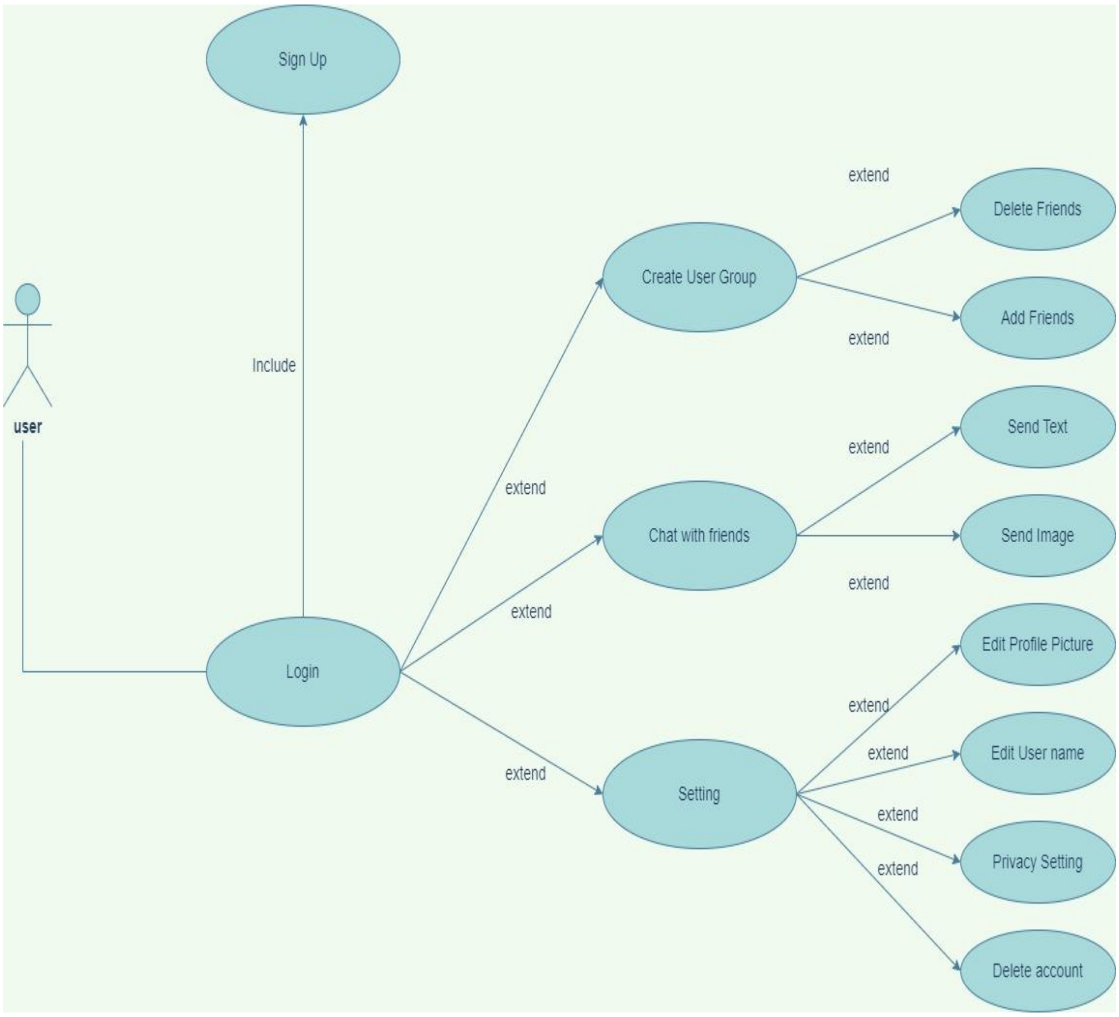


Figure 12. Use Case Diagram

3.4.4 State Diagram

A state diagram is a type of behavioral diagram in the Unified Modeling Language (UML) that represents the behavior of a system or object over time. It is used to model the life cycle of an object or system, showing how it responds to events and changes in its environment.

The purpose of a state diagram is to provide a visual representation of the system's behavior, showing how it responds to different events or stimuli. It is used to model complex systems or objects, such as software applications, hardware devices, or business processes.[36]

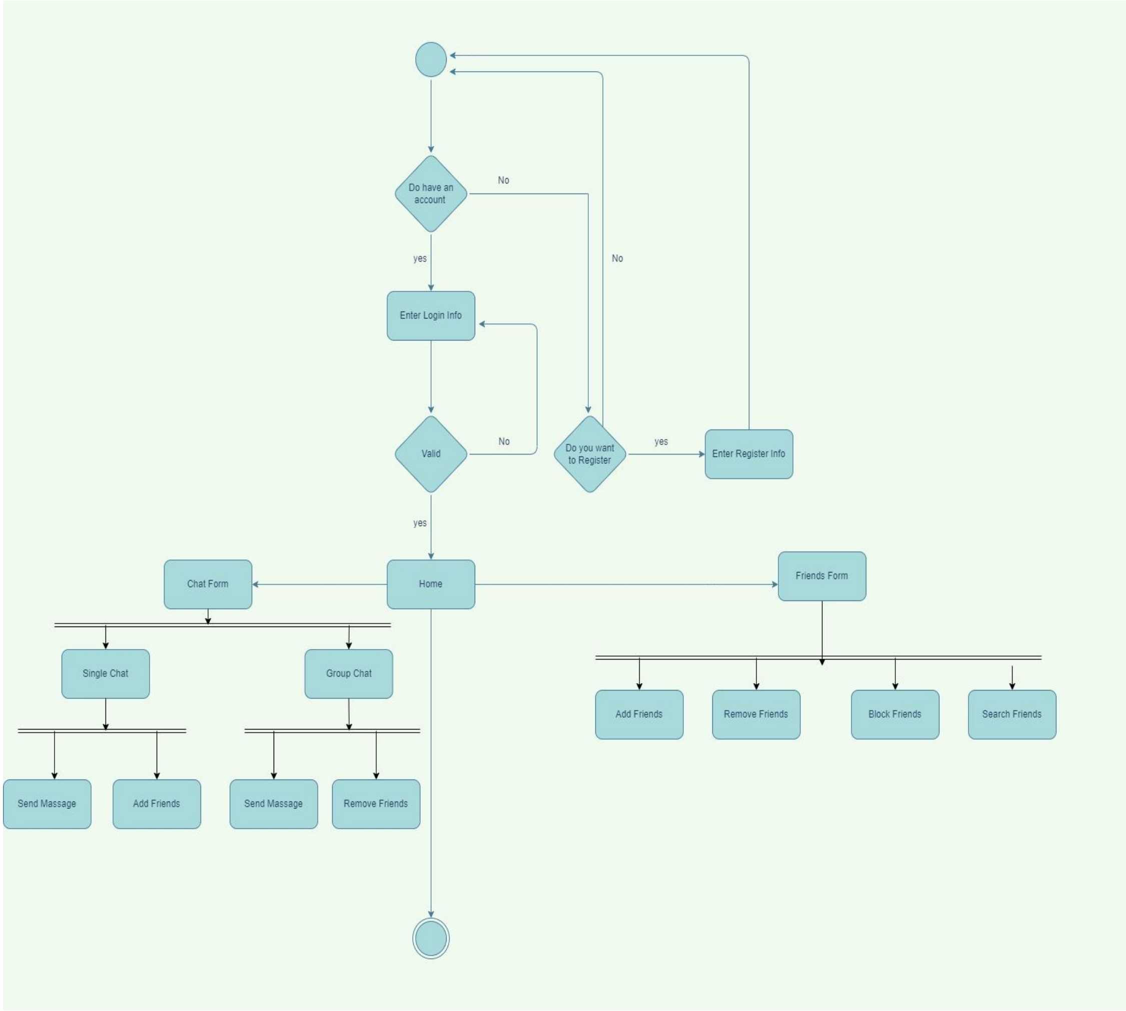


Figure 13. State Diagram

3.4.5 Process Flow Diagram

Process flow diagrams are useful tools for identifying inefficiencies, bottlenecks, or other issues in a process. They help to streamline processes, eliminate unnecessary steps, and increase efficiency and productivity. Process flow diagrams are used in a variety of industries, including manufacturing, health-care, finance, and information technology.[36]

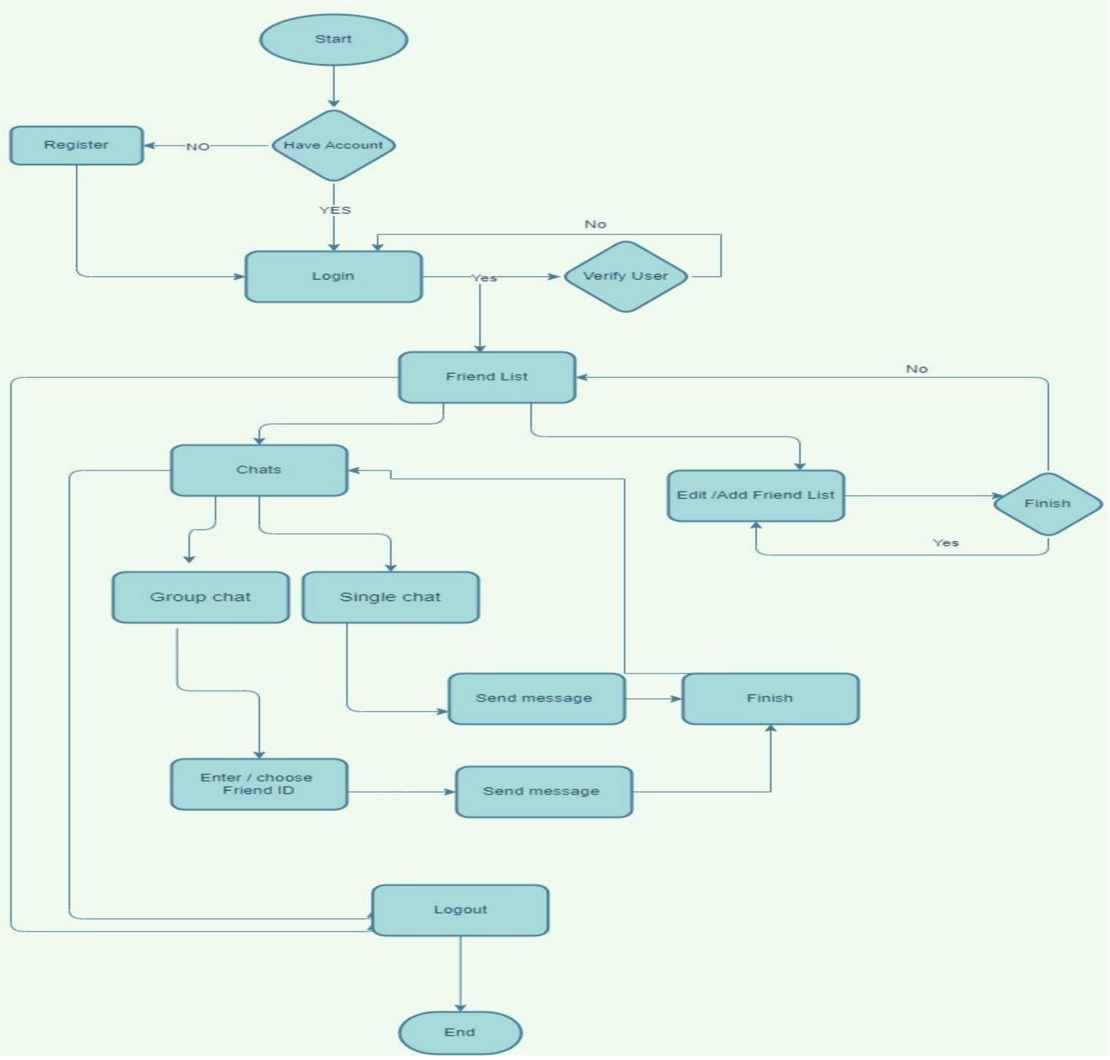


Figure 14. Process Flow Diagram

3.4.6 Sequence Diagram

A sequence diagram is a type of interaction diagram in the Unified Modeling Language (UML) that shows the interactions between objects or components in a system over time. It is used to model the flow of messages or events between objects or components and to illustrate the order of these interactions.

The purpose of a sequence diagram is to provide a visual representation of the system's behavior, showing how objects or components interact with each other and in what order. It is used to model complex systems or processes, such as software applications, web services, or business processes.[36]

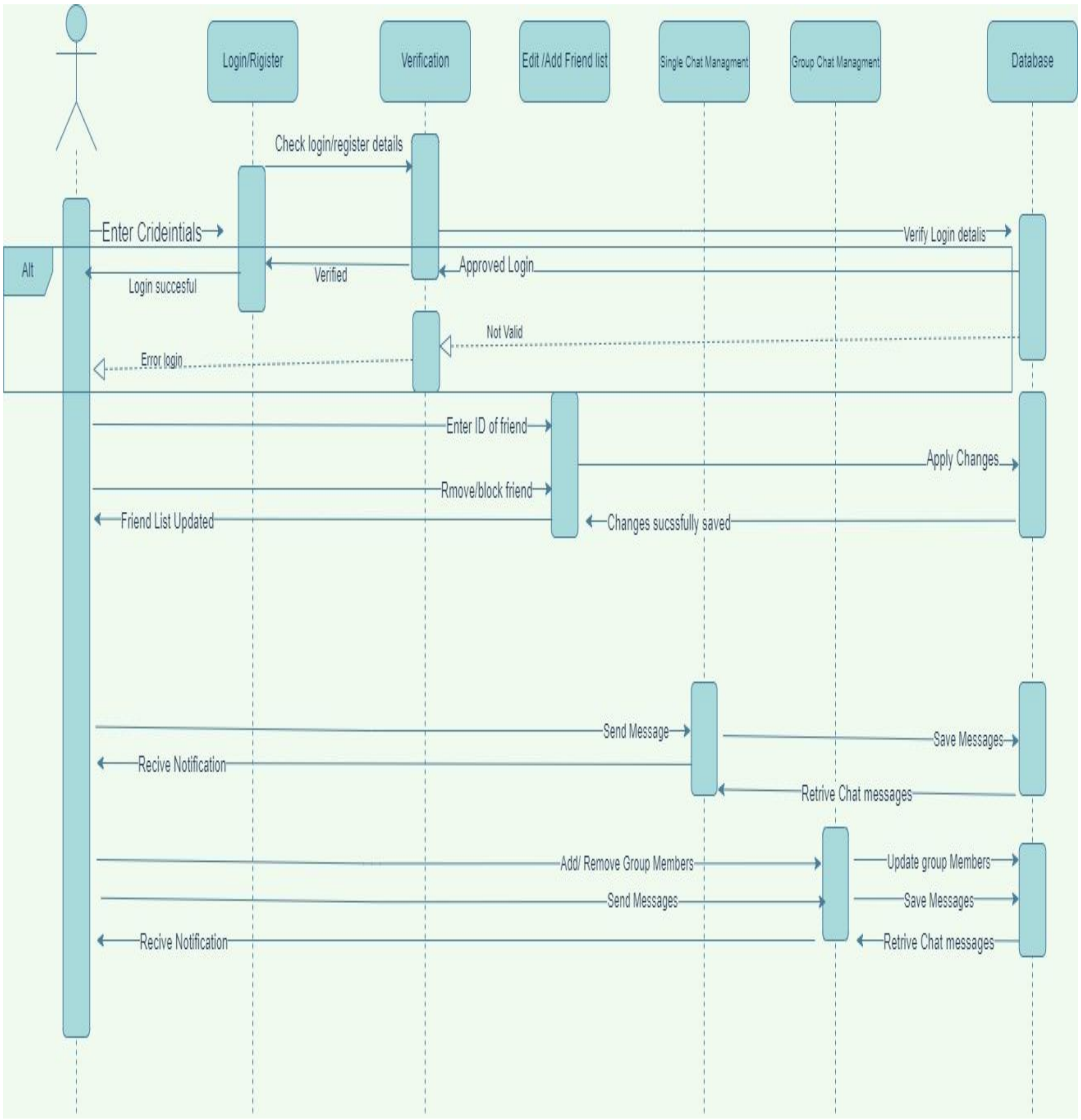


Figure 15. Sequence Diagram

3.4.7 Class Diagram

A class diagram is a type of static structure diagram in the Unified Modeling Language (UML) that represents the structure and relationships of classes in a system or software application. It is used to model the classes and objects in a system, as well as their attributes, methods, and relationships.

The purpose of a class diagram is to provide a visual representation of the system's structure and relationships between classes. It is used to model complex systems or software applications, and to ensure that the system is designed and implemented to the highest standards of quality and usability.[36]

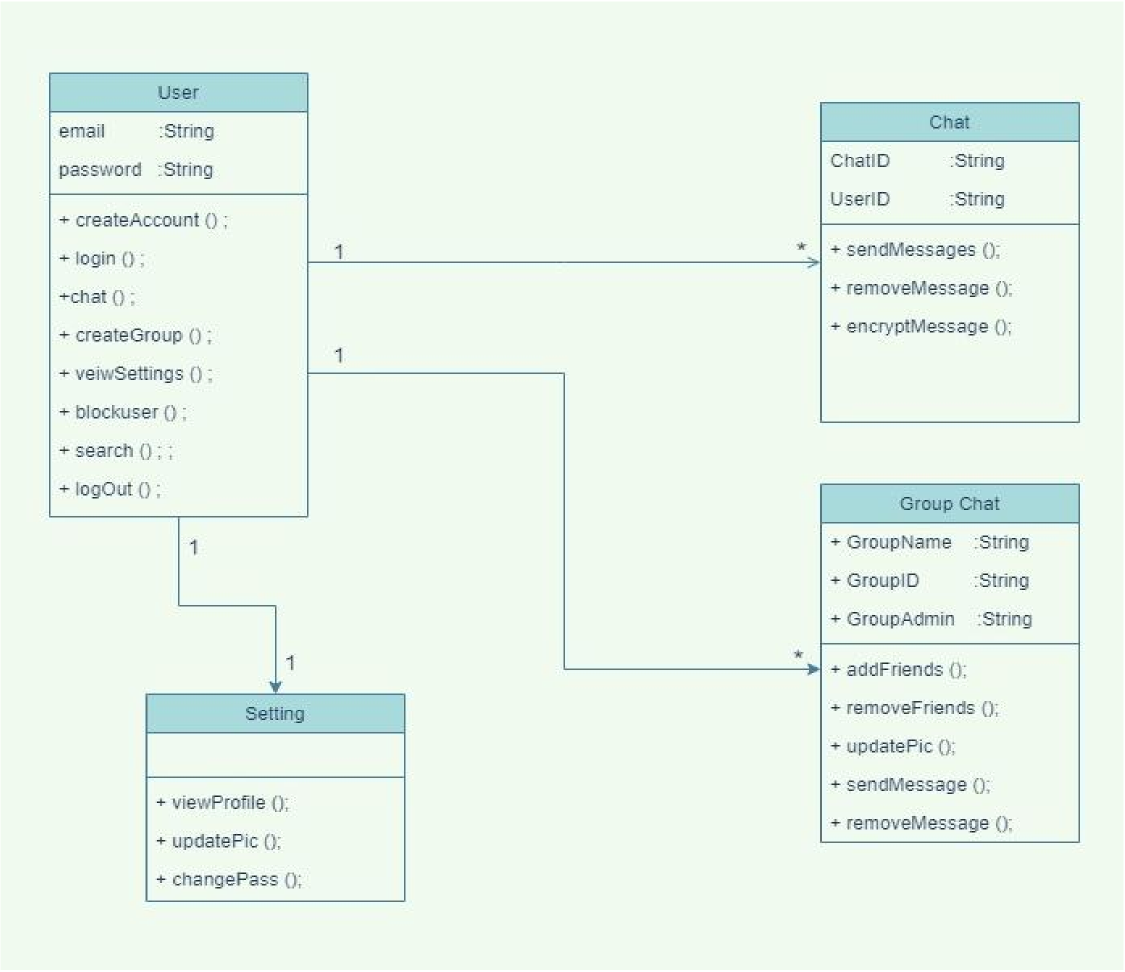


Figure 16. Class Diagram

3.4.8 Entity Diagram

An entity-relationship (ER) diagram, also known as an entity-relationship model, is a type of data modeling diagram that represents entities and their relationships to each other. It is used to model the structure of a database and the relationships between the entities within it.

The purpose of an ER diagram is to provide a visual representation of the structure of a database, showing the entities and their relationships to each other. It is used in database design and development to ensure that the database is well-organized and optimized for the intended use.[35]

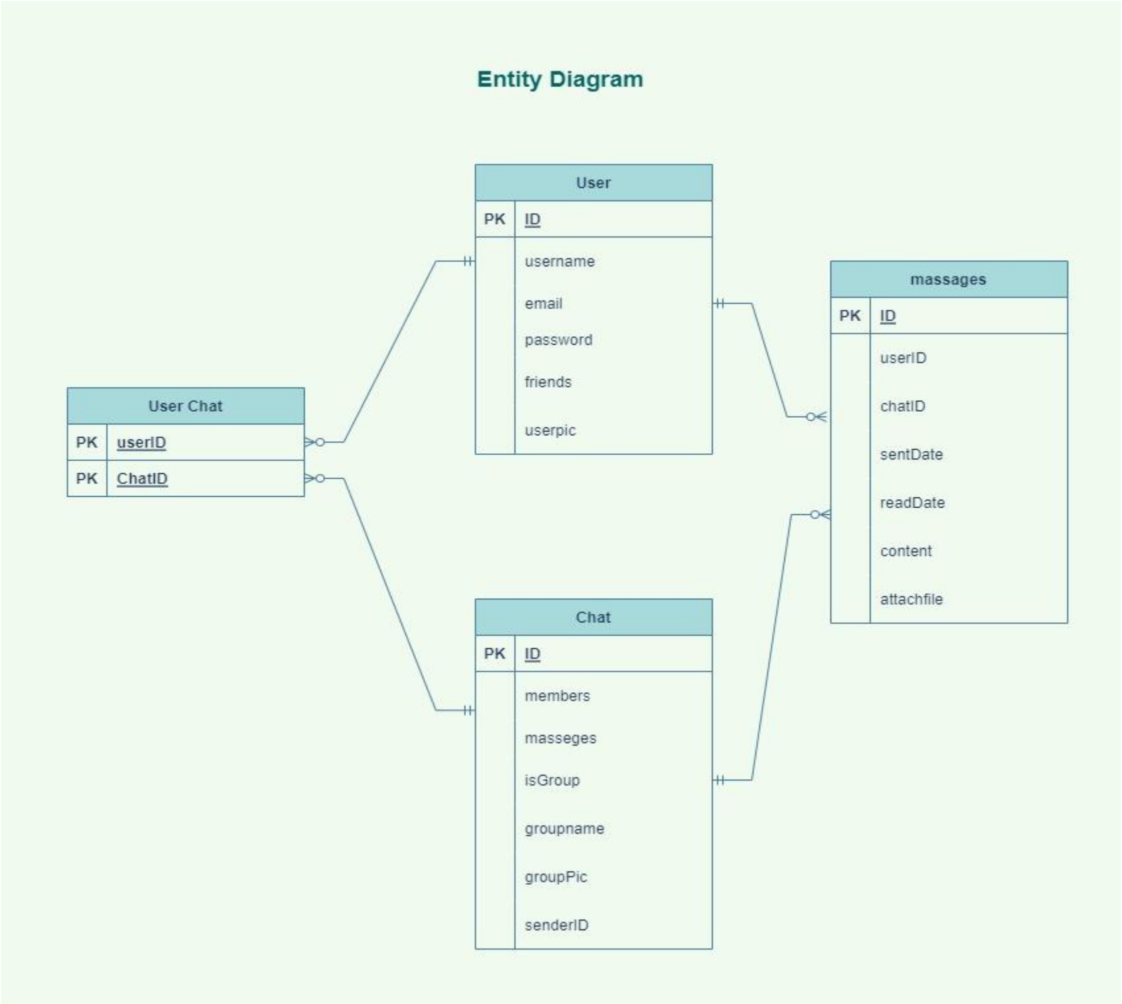


Figure 17. ERD

Chapter Four

4 PROPOSED SYSTEM

4.1 OVERVIEW

In the previous chapter, we discussed the analysis phase of the proposed system ‘Wannahide’, and its procedural diagrams. This chapter will give all the details of the software or application. This includes system methodology, all algorithms used, implemented systems functions, and presenting a draft timeline plan to finish.

4.2 METHODOLOGIES

4.2.1 End-to-End Encryption

WannaHide app utilizes end-to-end encryption to secure all communication between users.

End-to-end encryption ensures that only the intended recipients can access and decrypt the messages exchanged.

Encryption and decryption processes are performed exclusively on the user's device, protecting the messages from unauthorized access during transit.

4.2.2 ElGamal Algorithm

The ElGamal encryption algorithm is employed by WannaHide app for secure key exchange and message encryption.

The algorithm is based on the computational complexity of the Discrete Logarithm Problem in a finite field.

The sender and receiver each generate their public and private key pairs using the ElGamal algorithm.

The sender's public key is used to encrypt the message, while the receiver's private key is used for decryption.

ElGamal encryption offers a high level of security, as it relies on the mathematical properties of discrete logarithms, which are considered computationally difficult to solve.[18]

4.2.3 Double Ratchet

WannaHide app incorporates the Double Ratchet algorithm to provide forward secrecy and secure messaging.

The Double Ratchet algorithm is a key management protocol that ensures each message has a unique encryption key.

It generates new session keys for each message and updates them regularly, providing perfect forward secrecy.

Forward secrecy guarantees that even if one session key is compromised, previous and future messages remain secure.

The Double Ratchet algorithm also facilitates asynchronous communication, allowing users to receive and decrypt messages even if they were offline during the message exchange.[10]

4.2.4 Key Exchange:

The WannaHide app implements a secure key exchange mechanism for establishing shared secret keys between users.

The Diffie-Hellman key exchange protocol is utilized, which enables secure key generation without transmitting the keys over the network.

Each user generates their Diffie-Hellman public and private keys.

During a key exchange, users exchange their public keys and use them to compute a shared secret key without revealing their private keys.

The shared secret key is then used to derive session keys for encryption and decryption using the ElGamal algorithm.

4.2.5 Secure Communication

WannaHide app ensures that all messages, voice calls, and video calls are protected with end-to-end encryption.

The combination of the ElGamal algorithm and the Double Ratchet algorithm guarantees secure and private communication.

The encryption keys are only known to the communicating parties, ensuring confidentiality and integrity of the exchanged data.

WannaHide app prioritizes user privacy and security by preventing unauthorized access to user messages and minimizing the risk of data breaches.

4.3 ALGORITHMS

4.3.1 ELGAMAL algorithm

The ElGamal encryption algorithm is a public-key cryptosystem that allows secure communication over insecure channels by encrypting messages using the recipient's public key. The algorithm consists of two main parts: key generation and encryption/decryption.

In the key generation process, the sender (Alice) generates a pair of keys: a private key and a public key. The private key is kept secret and used to decrypt messages, while the public key is shared with others and used to encrypt messages. The key generation process involves the following steps:

- Choose a large prime number p and a generator g of the multiplicative group of integers modulo p.
- Choose a random secret number a, such that 1 < a < p - 1.
- Compute A = g^a mod p and publish (p, g, A) as the public key ,keep a as the private key.

In the encryption process, the sender (Bob) encrypts a message M using Alice's public key (p, g, A) as follows:

Choose a random secret number k, such that 1 < k < p - 1 and gcd(k, p - 1) =1.

- Compute B = g^k mod p and C = AM^k mod p.
- Send (B, C) as the encrypted message.

In the decryption process, Alice uses her private key a to decrypt the message (B, C) as follows:

- Compute D = B^a mod p.
- Compute M = CD^-1 mod p.

As has been demonstrated its based on the computational hardness of the discrete logarithm problem, which is the difficulty of computing the discrete logarithm of a given number modulo a prime. The security of the algorithm depends on the difficulty of computing the private key a from the public key (p, g, A) and the encrypted message (B, C). If the key size and other parameters are chosen appropriately.[20][21][22]

4.3.2 Double-ratchet Algorithm

The double ratchet end-to-end encryption protocol is a cryptographic technique used to secure communication between two parties, such as in messaging applications. The protocol utilizes the double ratchet model to provide forward secrecy and message integrity.

In this protocol, both parties generate a pair of public and private keys. The public key is shared between the parties and used for encryption, while the private key is kept secret and used for decryption. The parties also generate a shared secret key that is used to encrypt and decrypt messages.

The double ratchet model is then used to establish a session key that is used to encrypt messages. The first ratchet is used to generate a new key for each message, while the second ratchet is used to update the shared secret key.

When a message is sent, the sender generates a new message key, denoted as mk, and encrypts the message using the session key derived from the shared secret key and the message key. This is represented by the following equation:

- $C = E(Sk, E(mk, M))$.

where C is the ciphertext, Sk is the session key derived from the shared secret key, E is the encryption function, mk is the message key, and M is the message.

The session key, Sk, used in the double ratchet protocol is derived from the shared secret key, SS, and a message key, mk, as follows:

- $Sk = HKDF(SS, mk)$.

The sender then uses the first ratchet to generate a new message key, denoted as mk', which is used for the next message. This is represented by the following equation:

$mk' = HKDF(ck, 1)$

where HKDF is a key derivation function and ck is the chaining key.

The receiver then uses the second ratchet to update the shared secret key. The receiver first decrypts the ciphertext using the session key derived from the shared secret key and the message key, and then uses the decrypted message key to update the shared secret key. This is represented by the following equations:

- mk = E(Sk, E(mk', M'))
- Sk' = HKDF(Sk, ck)

where M' is the decrypted message, Sk' is the updated session key derived from the updated shared secret key, and ck is the updated chaining key.

The sender and receiver then repeat this process for each subsequent message, generating new message keys and updating the shared secret key, ensuring forward secrecy and message integrity.

To sum up , the double ratchet end-to-end encryption protocol utilizes the double ratchet model to provide secure communication between two parties. The protocol provides forward secrecy and message integrity by generating a new message key for each message and updating the shared secret key using the double ratchet model. This protocol provides a secure communication channel for messaging applications and is widely used in modern end-to-end encrypted messaging services.[12][11]

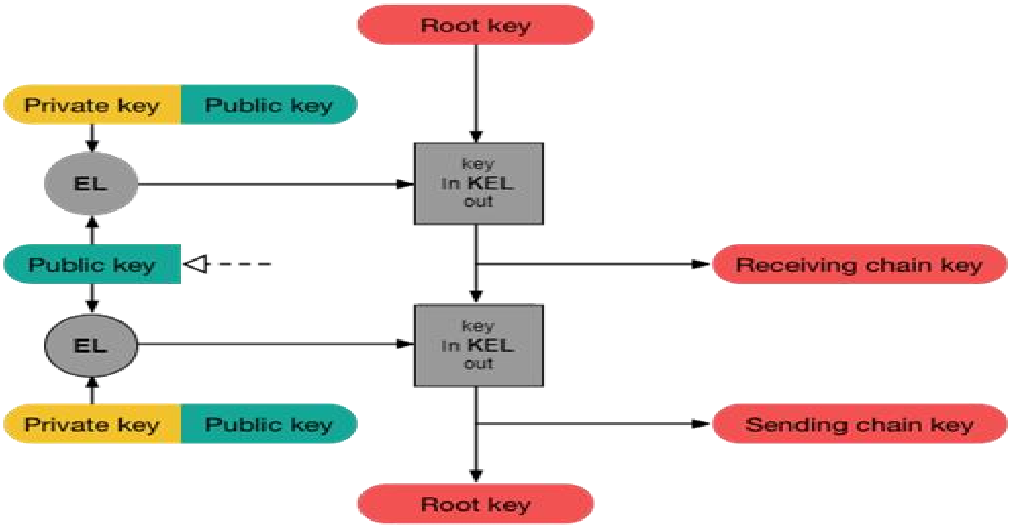


Figure 18. Double ratchet mechanism

4.3.3 AES 256-CBC encryption algorithm

AES-256-CBC refers to a specific encryption algorithm and mode of operation used for symmetric encryption. Let's break down its components:

AES: AES stands for Advanced Encryption Standard, which is a widely used symmetric encryption algorithm. It was selected by the U.S. National Institute of Standards and Technology (NIST) as a replacement for the older Data Encryption Standard (DES). AES operates on fixed-size blocks of data, typically 128 bits, and supports key sizes of 128, 192, and 256 bits.

256: The number 256 indicates the key size used in AES-256. It means that the encryption algorithm employs a 256-bit key for encrypting and decrypting data. AES-256 is considered highly secure and suitable for protecting sensitive information.

CBC: CBC stands for Cipher Block Chaining, which is a mode of operation for block ciphers like AES. In CBC mode, each block of plain-text is combined with the previous cipher-text block before encryption. This chaining process adds randomness and increases the security of the encryption. CBC requires an initialization vector (IV) that serves as the initial input to the encryption algorithm.

To encrypt data using AES-256-CBC:

1. Generate a 256-bit encryption key.

2. Generate a random 128-bit initialization vector (IV).

3. Divide the plain text into fixed-size blocks (usually 128 bits).

4. XOR (exclusive OR) the first plain text block with the IV.

5. Encrypt the XOR result using AES-256 with the encryption key.

6. XOR the resulting cipher text block with the next plain text block.

7. Encrypt the XOR result using AES-256 with the encryption key.

8. Repeat the XOR and encryption steps for each subsequent block, using the previous cipher text block in the XOR operation.

9. The final output is the encrypted cipher text.

To decrypt data using AES-256-CBC, the process is reversed:

1. Retrieve the encryption key and IV.

2. Divide the cipher text into fixed-size blocks.

3. Decrypt the first cipher text block using AES-256 and the encryption key.

4. XOR the decrypted block with the IV (for the first block) or the previous ciphertext block (for subsequent blocks).

5. Decrypt the next cipher text block using AES-256 and the encryption key.

6. XOR the decrypted block with the previous ciphertext block.

7. Repeat the XOR and decryption steps for each subsequent block.

8. The final output is the decrypted plain text.

To sum up, AES-256-CBC is a widely used encryption algorithm that provides strong security for data at rest and in transit. It is a symmetric-key block cipher that uses a fixed-length key of 256 bits to encrypt and decrypt data. In CBC mode, the algorithm uses a block cipher to encrypt plain-text in blocks of fixed size and applies an initialization vector (IV) to the first block to increase randomness and prevent patterns from being detected in the encrypted data.

The security of AES-256-cbc encryption depends on the strength of the symmetric key and the randomness of the initialization vector. The key size of 256 bits provides a very large key space, making brute-force attacks impractical. The use of a random initialization vector ensures that each block of cipher-text is unique, making it difficult to detect patterns in the encrypted data.[7][8]

4.4 WANNAHIDE ENCRYPTION SCHEME

The ElGamal algorithm is a public-key encryption algorithm that allows two parties to exchange messages securely without having to share a secret key in advance. In this algorithm, each party generates a public and private key pair. The public key can be shared with anyone, while the private key must be kept secret.

To initiate a secure conversation, the two parties exchange their public keys. Each party then uses the other party's public key to encrypt a secret key, which is used for symmetric encryption of the actual messages. This secret key is then exchanged securely between the two parties using the asymmetric encryption mechanism.

Once the secret key is shared, the double ratchet algorithm is used for symmetric encryption of the messages. The double ratchet algorithm is a key management algorithm that generates a new symmetric key for each message sent. This ensures that if one key is compromised, only a single message is affected.

The double ratchet algorithm works by maintaining two key chains, one for sending messages and one for receiving messages. Each chain consists of a series of keys, and a new key is generated for each message sent or received. The sender and receiver synchronize their key chains at the beginning of each message exchange to ensure that they are using the same key.

The double ratchet algorithm also provides forward secrecy, which means that even if an attacker gains access to a previous key, they cannot use it to decrypt future messages.

In summary, end-to-end encryption in Wannahide using hybrid encryption involves the use of the ElGamal algorithm for asymmetric encryption of a secret key, which is then used in conjunction with the double ratchet algorithm for symmetric encryption of the messages. This approach provides strong security guarantees and ensures that even if one key is compromised, only a limited number of messages are affected.

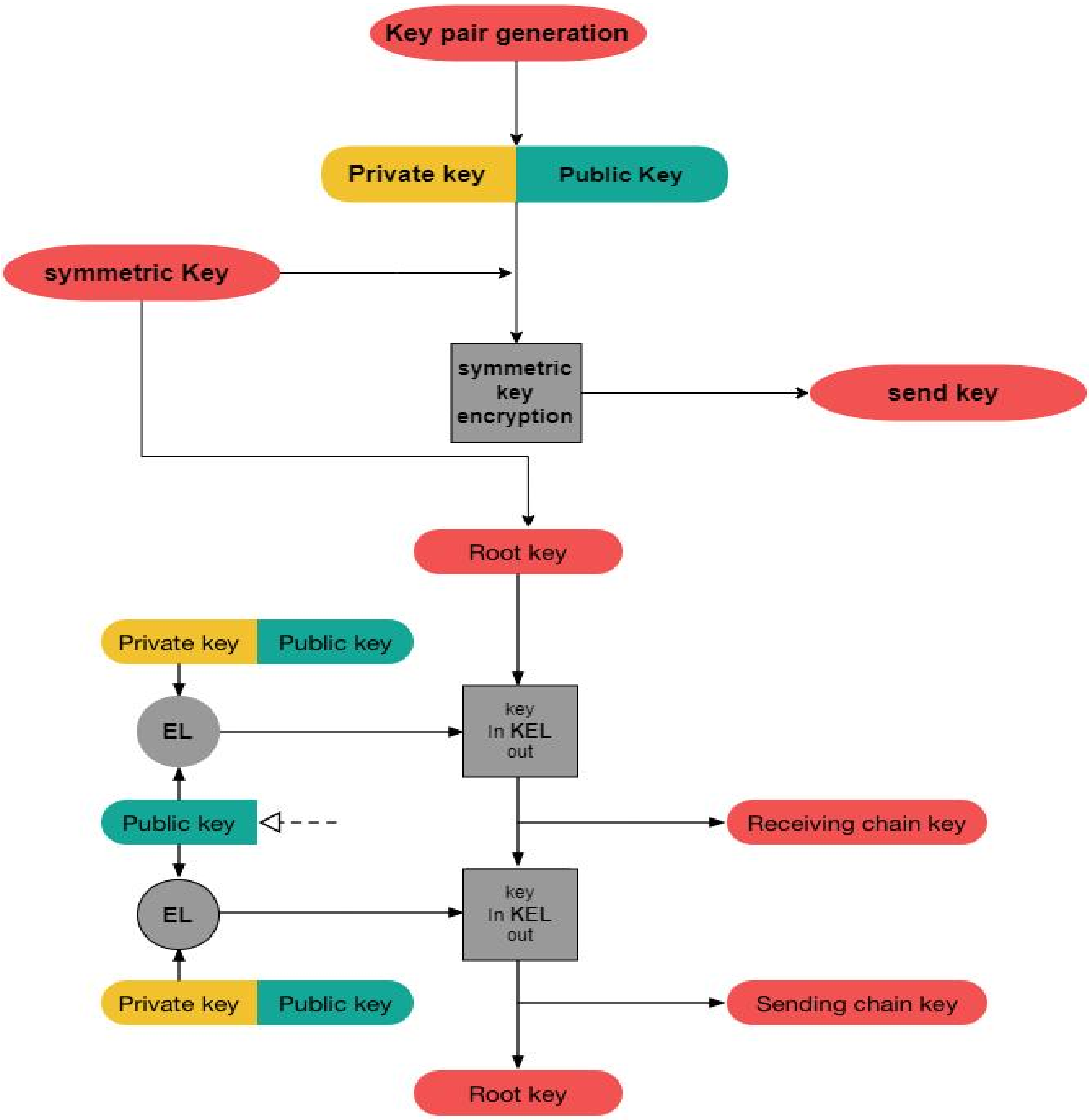


Figure 19. Asymmetric Encryption mechanism in Wannahide

4.5 PROPOSED APPLICATION

The aim of wannahide app is to provide an innovative chat application designed to prioritize your privacy and security. First of all the user has to sign up to the application, as soon as he fills the needed information, a confirmation email is sent to him at the email that is provided in the sign up form. Once the account is activated he can freely sign in and use the application, the user can communicate with other users by adding them, by the time the other user accepts the request a chat session is opened between the two parties.

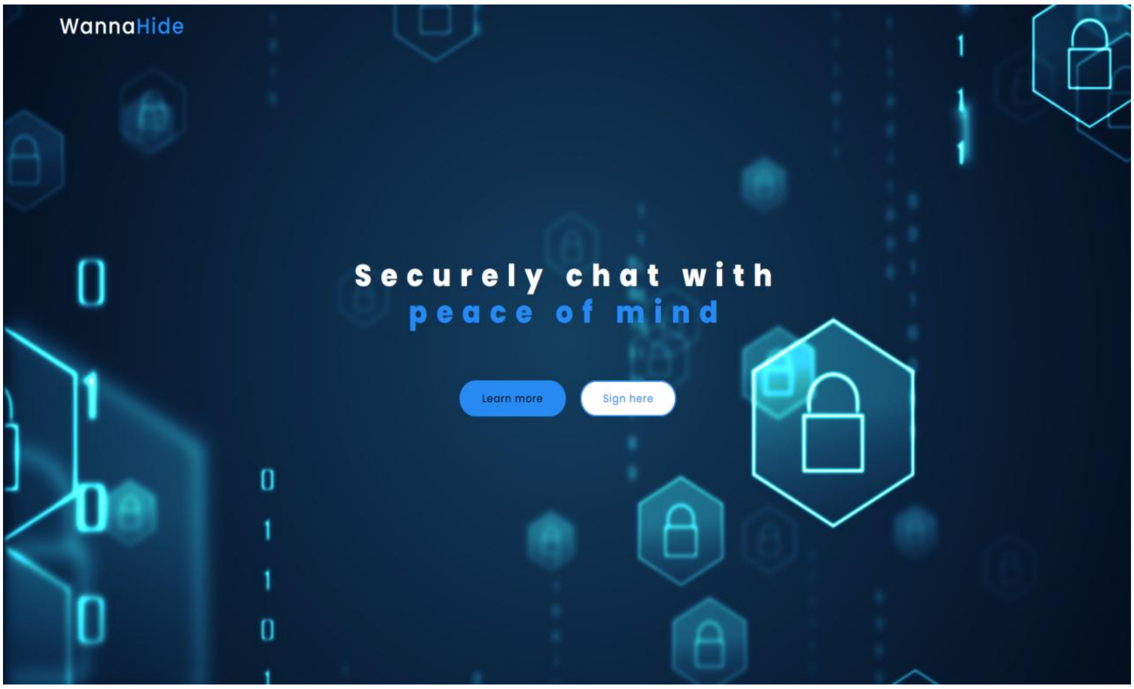


Figure 20. WannaHide application homepage

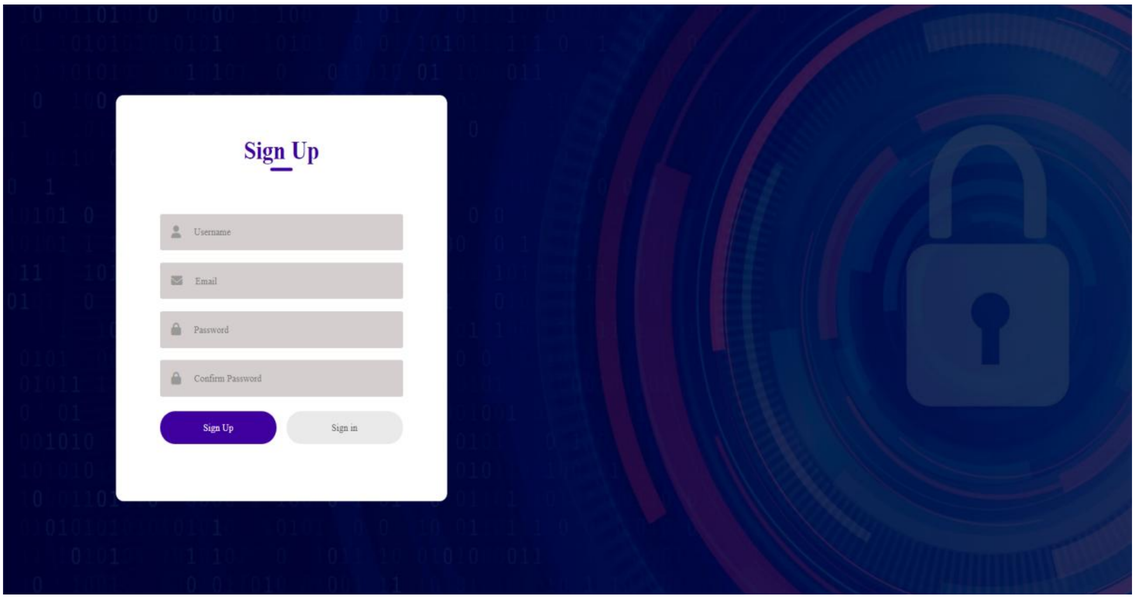


Figure 21. WannaHide application sign up form

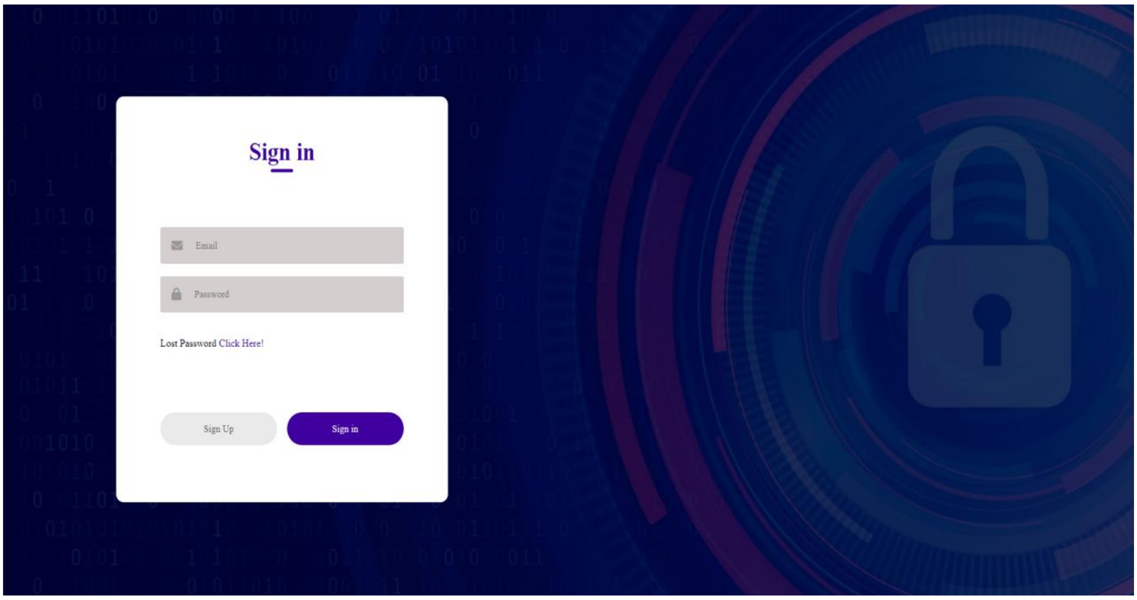
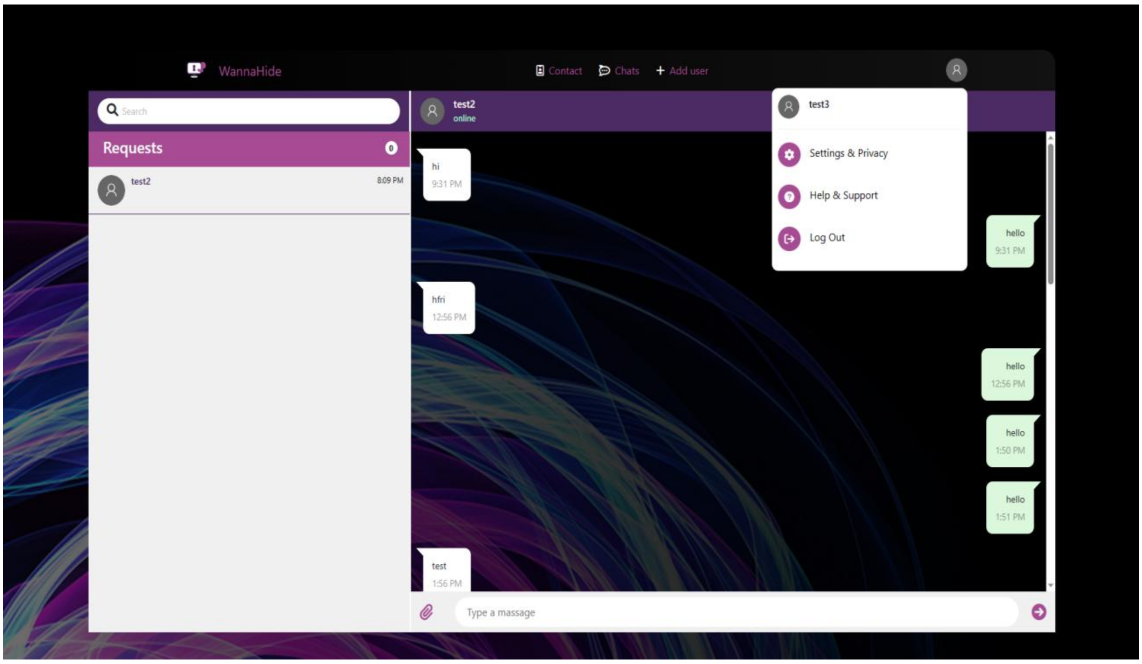


Figure 22. WannaHide application sign in form



test2 8:09 PM

test2
online

hi
9:31 PM

hello
9:31 PM

hfri
12:56 PM

hello
12:56 PM

hello
1:50 PM

hello
1:51 PM

test
1:56 PM

Figure 23. WannaHide application chat page

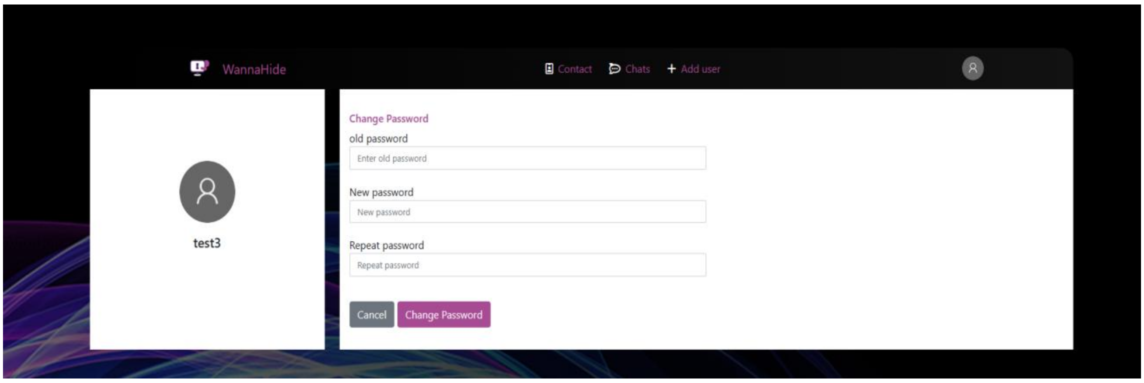
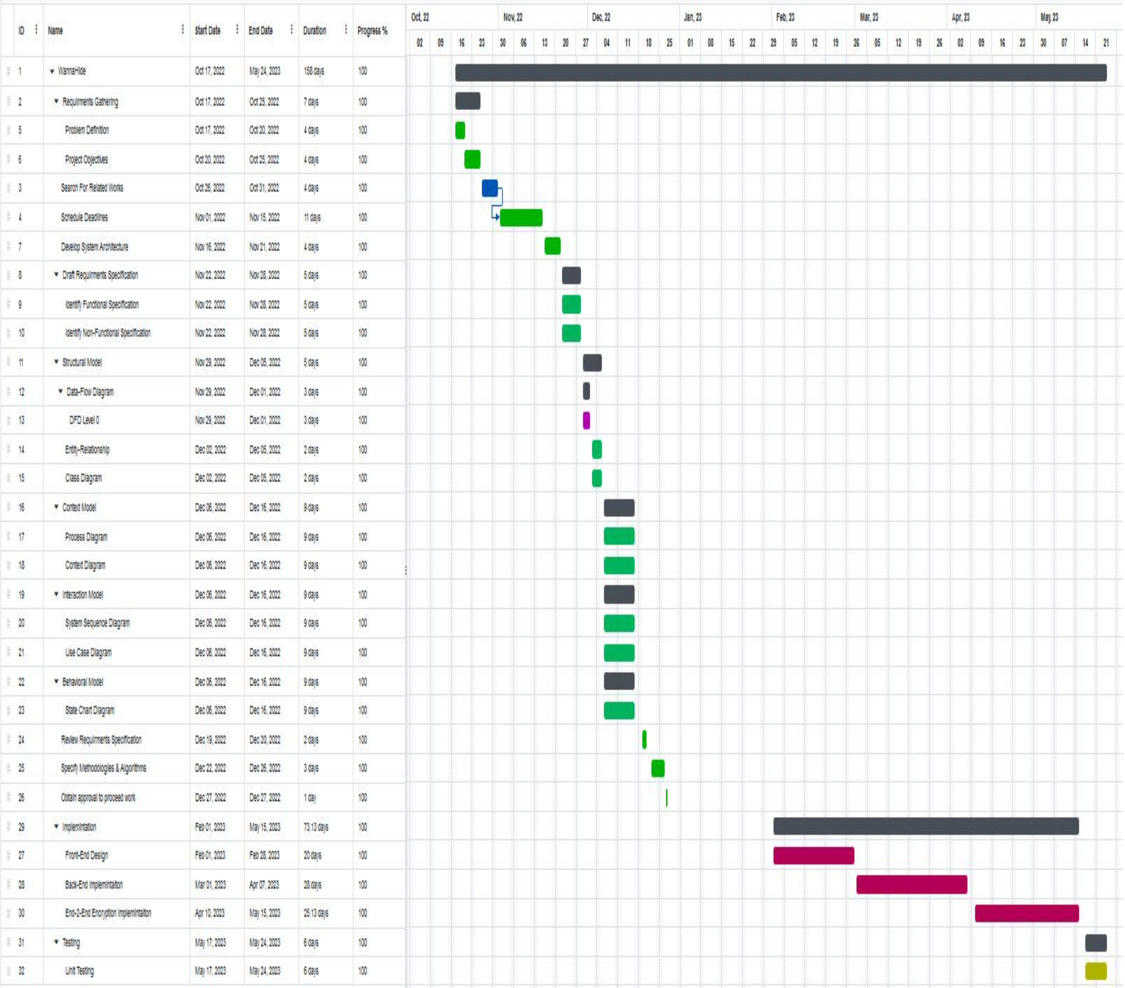


Figure 24. WannaHide application settings page

4.6 TEAM PROGRESS

The following Gantt Chart describes the progress of the project by itemizing the team's accomplishments to date and the related tasks that are still in process. For each task, indicate the date set for its completion and the dependencies that depend on the competition of that task.



ID	Name	Start Date	End Date	Duration	Progress %
1	▼ WannaHide	Oct 17, 2022	May 24, 2023	158 days	100
2	▼ Requirments Gathering	Oct 17, 2022	Oct 25, 2022	7 days	100
5	Problem Definition	Oct 17, 2022	Oct 20, 2022	4 days	100
6	Project Objectives	Oct 20, 2022	Oct 25, 2022	4 days	100
3	Search For Related Works	Oct 26, 2022	Oct 31, 2022	4 days	100
4	Schedule Deadlines	Nov 01, 2022	Nov 15, 2022	11 days	100
7	Develop System Architecture	Nov 16, 2022	Nov 21, 2022	4 days	100
8	▼ Draft Requirments Specification	Nov 22, 2022	Nov 28, 2022	5 days	100
9	Identify Functional Specification	Nov 22, 2022	Nov 28, 2022	5 days	100
10	Identify Non-Functional Specification	Nov 22, 2022	Nov 28, 2022	5 days	100
11	▼ Structural Model	Nov 29, 2022	Dec 05, 2022	5 days	100
12	▼ Data-Flow Diagram	Nov 29, 2022	Dec 01, 2022	3 days	100
13	DFD Level 0	Nov 29, 2022	Dec 01, 2022	3 days	100
14	Entity-Relationship	Dec 02, 2022	Dec 05, 2022	2 days	100
15	Class Diagram	Dec 02, 2022	Dec 05, 2022	2 days	100
16	▼ Context Model	Dec 06, 2022	Dec 16, 2022	9 days	100
17	Process Diagram	Dec 06, 2022	Dec 16, 2022	9 days	100
18	Context Diagram	Dec 06, 2022	Dec 16, 2022	9 days	100
19	▼ Interaction Model	Dec 06, 2022	Dec 16, 2022	9 days	100
20	System Sequence Diagram	Dec 06, 2022	Dec 16, 2022	9 days	100
21	Use Case Diagram	Dec 06, 2022	Dec 16, 2022	9 days	100
22	▼ Behavioral Model	Dec 06, 2022	Dec 16, 2022	9 days	100
23	State Chart Diagram	Dec 06, 2022	Dec 16, 2022	9 days	100
24	Review Requirments Specification	Dec 19, 2022	Dec 20, 2022	2 days	100
25	Specify Methodologies & Algorithms	Dec 22, 2022	Dec 26, 2022	3 days	100
26	Obtain approval to proceed work	Dec 27, 2022	Dec 27, 2022	1 day	100
29	▼ Implemintation	Feb 01, 2023	May 15, 2023	73.13 days	100
27	Front-End Design	Feb 01, 2023	Feb 28, 2023	20 days	100
28	Back-End Implemintaiton	Mar 01, 2023	Apr 07, 2023	28 days	100
30	End-2-End Encryption Implemintaiton	Apr 10, 2023	May 15, 2023	25.13 days	100
31	▼ Testing	May 17, 2023	May 24, 2023	6 days	100
32	Unit Testing	May 17, 2023	May 24, 2023	6 days	100

Figure 25. Gantt chart

5 RESULTS AND DISCUSSION

5.1 OVERVIEW

In the previous chapter, we discussed the system methodology and all algorithms used in the ‘Kindergarten Educational app’. This chapter will give all the results of the application. This includes the classification model flow and discussion of the results including a comparative study with the previously explained related works to finish.

5.2 MODEL FLOW

5.2.1 Choosing algorithm

In light of the multitude of well-known chat applications available, each with its own security features, extensive research and analysis have led us to select the Elgamal encryption algorithm as the foundation for ensuring data integrity within our application. This strategic decision was driven by the algorithm's ability to provide a heightened level of security against potential threats, including brute force attacks and key compromise. Moreover, Elgamal's underlying strength lies in its reliance on the computational complexity of the Discrete Logarithm Problem, which is widely recognized as more formidable than both the Integer Factorization Problem (RSA) and the Discrete Logarithm Problem in a multiplicative group (DH). Elgamal algorithm main purpose is to provide both encryption, decryption and digital signature, whereas Diffie-Hellman is primarily used for key exchange rather than data integrity purposes.

5.2.2 Applying algorithm to specific data categories:

According to statistics provided by WhatsApp in the year 2022 , text messages were the most common type of messages exchanged between users.[29]

Choosing the most common types was our second step to determine what kind of data will be encrypted and sent in Wannahide. For text messages we utilize both Elgamal encryption algorithm using Asymmetric encryption and double ratchet that completes the encryption process with symmetric encryption.

As for images its converted to base64 string and repeat the same process that happens in the text encryption process. Ensuring that data categories mentioned can be encrypted and decrypted for both sender and receiver.

5.2.3 Building the entire encryption scheme

To initiate a secure conversation, the two parties exchange their public keys. Each party then uses the other party's public key to encrypt a secret key, which is used for symmetric encryption of the actual messages. This secret key is then exchanged securely between the two parties using the asymmetric encryption mechanism.

Once the secret key is shared, the double ratchet algorithm is used for symmetric encryption of the messages. The double ratchet algorithm is a key management algorithm that generates a new symmetric key for each message sent. This ensures that if one key is compromised, only a single message is affected.

The double ratchet algorithm works by maintaining two key chains, one for sending messages and one for receiving messages. Each chain consists of a series of keys, and a new key is generated for each message sent or received. The sender and receiver synchronize their key chains at the beginning of each message exchange to ensure that they are using the same key.

5.3 APPLICATION RESULTS

The first step after implementing our application was to try it and experience its performance for small group of users, getting their feedback helped us improve our implementation to enhance user experience.

5.4 CONCLUSION

Purpose of Wannahide is to make instant messaging service more secure through double ratchet end-to-end encryption based on enhancement of Diffie-Hellman algorithm called Elgamal crypto system that made encryption process more secure, faster, and less vulnerable to practical attacks that threats the Confidentiality of the communication process.

6 REFERENCES

[1] Alisawi, W. C., Oleiwi, Z. C., Alawsi, W. A., Alfoudi, A. S., & Hadi, N. K. (2019). Improvement of classical cipher algorithm based on a new model of timed-released encryption. International Journal of Applied Engineering Research, 14(16), 3531-3536.

[2] Oleiwi, Z. C., Abdallah, W., & Alisawi, W. C. (2023, june 27). publication/342521740_Overview_and_Performance_Analysis_of_Encryption_Algorithms. Retrieved from researchgate: https://www.researchgate.net/publication/342521740_Overview_and_Performance_Analysis_of_Encryption_Algorithms

[3] Ringcentral Corporate. (2023, March 31). Introducing dynamic end-to-end encryption for RingCentral Video. Retrieved from ringcentral: https://www.ringcentral.com/us/en/blog/dynamic-end-to-end-encryption/

[4] De Luca, A., Das, S., Ortlieb, M., Ion, I., & Laurie, B. (2016). Expert and non-expert attitudes towards (secure) instant messaging.

[5] Abu-Salma, R., Sasse, M. A., Bonneau, J., Danilova, A., Naiakshina, A., & Smith, M. (2017, May). Obstacles to the adoption of secure communication tools. In 2017 IEEE Symposium on Security and Privacy (SP) (pp. 137-153). IEEE.

[6] Vaziripour, E., Wu, J., O'Neill, M., Whitehead, J., Heidbrink, S., Seamons, K., & Zappala, D. (2017). Is that you, Alice? A usability study of the authentication ceremony of secure messaging applications. In Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017) (pp. 29-47).

[7] Wu, J., & Zappala, D. (2018, August). When is a Tree Really a Truck? Exploring Mental Models of Encryption. In SOUPS@ USENIX Security Symposium (pp. 395-409).

[8] Tan, J., Bauer, L., Bonneau, J., Cranor, L. F., Thomas, J., & Ur, B. (2017, May). Can unicorns help users compare crypto key fingerprints?. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 3787-3798).

[9] Dechand, S., Naiakshina, A., Danilova, A., & Smith, M. (2019, June). In encryption we don’t trust: The effect of end-to-end encryption to the masses on user perception. In 2019 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 401-415). IEEE.

[10] Marlinspike, M. (2016, November 20). The Double Ratchet Algorithm. Retrieved from whispersystems: https:// whispersystems.org/docs/specifications/doubleratchet/doubleratchet.pdf

[11] Marlinspike, M. (04 11 ,2016). The X3DH Key Agreement Protocol. Retrieved from signal: https://signal.org/docs/specifications/x3dh/x3dh.pdf

[12] Mihir Bellare, Asha Camper Singh, Joseph Jaeger, Maya Nyayapati, and Igors Stepanovs. Ratcheted encryption and key exchange: The security of messaging. In Jonathan Katz and Hovav Shacham, editors, CRYPTO 2017, Part III, volume 10403 of LNCS, pages 619–650, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[13] Alwen, J., Coretti, S., & Dodis, Y. (2019, April). The double ratchet: security notions, proofs, and modularization for the signal protocol. In Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I (pp. 129-158). Cham: Springer International Publishing.

[14] Diffie and M. Hellman, New directions in cryptography, Information Theory, IEEETransactions on, vol. 22, no. 6, pp. 644 - 654, 1976.

[15] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, CT-RSA 2001, volume 2020 of LNCS, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany

[16] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing, 33(1):167–226, November 2003.

[17] Bao, F., Deng, R. H., & Zhu, H. (2003). Variations of diffie-hellman problem. In Information and Communications Security: 5th International Conference, ICICS 2003, Huhehaote, China, October 10-13, 2003. Proceedings 5 (pp. 301-312). Springer Berlin Heidelberg.

[18] ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory, 31(4), 469-472.

[19] Ramzi A. Haraty, H. O.-N.-K. (2004, January). A Comparative Study of Elgamal Based Cryptographic Algorithms. Retrieved from researchgate: https://www.researchgate.net/publication/220709516_A_Comparative_Study_of_Elgamal_Based_Cryptographic_Algorithms

[20] Awad, Y., El-Kassar, A. N., & Kadri, T. (2018, August). Rabin public-key cryptosystem in the domain of Gaussian Integers. In 2018 International Conference on Computer and Applications (ICCA) (pp. 1-340). IEEE.

[21] Ramzi A. Haraty, A.-N. E.-K. (2006, August). A Comparative Study of Elgamal Based Digital Signature Algorithms. Retrieved from researchgate: https://www.researchgate.net/publication/4257126_A_Comparative_Study_of_Elgamal_Based_Digital_Signature_Algorithms

[22] Diffie and M.E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory IT-22 (1978),472–492

[23] R.Merlin Shalini & Mr.S.Dhamodharan, "Performance and Scaling Com-parison Study of RDBMS and NoSQL (MongoDB", in COMPUSOFT, An inter-national journal of advanced computer technology, 3 (11), November-2014 (Vol-ume-III, Issue-XI)

[24] Hema Krishnan, M. E. (2016, May). MongoDB – a comparison with NoSQL databases. Retrieved from researchgate: https://www.researchgate.net/publication/327120267_MongoDB_-_a_comparison_with_NoSQL_databases

[25] Zhang Zhaoyuan, "A preliminary study on the back-end technology of web Node.js [J]", Small and Medium Enterprise Management and Technology, no. 22, pp. 193-194, 2020.

[26] Naseem, S. Z., & Majeed, F. (2013, September). Extending HTML5 local storage to save more data; efficiently and in more structured way. In Eighth International Conference on Digital Information Management (ICDIM 2013) (pp. 337-340). IEEE.

[27] Stefan Kimak, J. E. (2015, December). The role of HTML5 IndexedDB, the past, present and future. Retrieved from researchgate: https://www.researchgate.net/publication/304410447_The_role_of_HTML5_IndexedDB_the_past_present_and_future

[28] Judge, S. M. (2018). Mobile forensics: Analysis of the messaging application Signal. University of Central Oklahoma.

[29] Whatsapp. (n.d.). About WhatsApp. Retrieved from whatsapp: https://www.whatsapp.com/about/

[30] Marlinspike, M. (2016, November 20). The Double Ratchet Algorithm. Retrieved from signal: https://signal.org/docs/specifications/doubleratchet/

[31] Telegram. (n.d.). MTProto Mobile Protocol. Retrieved from telegram: https://core.telegram.org/mtproto.

[32] Panghal, A. (2018, October 06). WhatsApp's End to End Encryption, How does it work? Retrieved from medium: https://medium.com/@panghalamit/whatsapp-s-end-to-end-encryption-how-does-it-work-80020977caa0

[33] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing, 33(1):167–226, November 2003.

[34] Davis, Feinstein, Gorgone, Longenecker and Valacich, (2002), IS 2002 Information Systems Model Curriculum, College of Business and Economics, Washington State University, Pullman, WA.

[35] Dennis, A. Wixom B., Tegarden D., (2012), Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach, 4 th edition; Chapters 4, 5, and 6 (pages 153 - 247).

[36] Naren. (2019, January 12). Introduction to System Architecture Design. Retrieved from medium: https://medium.com/backendarmy/introduction-to-system-architecture-design-fcd4f327b6c9

المستخلص

بشكل عام, يلاحظ أن الاتصال الرقمي - سواء كان عبارة عن الرسائل الفورية او مكالمات الصوت والفيديو - أصبح حاجة يومية في جميع أنحاء العالم .

ومع ذلك, فإن حماية اي نوع من البيانات في هذه الأنواع من الاتصالات يشكل تحديات جديدة في مجال أمن المعلومات, ولهذا السبب تم استخدام خوارزميات التشفير لضمان سرية البيانات وسلامتها وتوفرها.

حيث كانت الخطوة الأولى التي يجب اتباعها هي أن يتم تأمين وسيلة الاتصال نفسها كخطوة أساسية لتحقيق ذلك.
بالإضافة إلى تشفير البيانات الأصلية لتحويلها

الى نص غير مفهوم باستخدام نظام تشفير هجين يعتمد على :

AES-256-CBC

حيث يتم استخدام التشفير التناظري والغير تناظري في هذا النظام لضمان محتوى بيانات آمن سواء كان نص عادي أو صورة وتأكيد تسليم المذكرو بشكل مفهوم وصحيح.

تطبيق محادثات آمن

مقدم من :

فؤاد مصطفى منير *89638*

عبدالله رياض الأعرج *89638*

عبدالرحمن فتحي شعبان *89393*

محمود محمد *89401*

تحت إشراف:

أ.د/ رانيا الجوهري

م. إسلام سعيد

م. نهال محمد

جامعة مصر للعلوم والتكنولوجيا - *MUST*

كلية الحاسبات وتقنيات الذكاء الاصطناعي - *CAIT*

قسم علوم الحاسب - *CS*