**DEPARTMENT OF CYBER SECURITY SCIENCE**

**FACULTY OF COMPUTING AND INFORMATICS**

**LADOKE AKINTOLA UNIVERSITY OF TECHNOLOGY,**

**OGBOMOSHO, OYO STATE NIGERIA**

**PROJECT ON**

**FCI SWEP 200**

**GROUP 7**

**File encryption and decryption system**

**Lecturer:**

**MR OJO**

|  |  |  |
| --- | --- | --- |
| S/N | GROUP MEMBERS | MATRIC. NO |
| 1. | Kolade Hammed Olaniyi | 2022005464 |
| 2. | Ajala Praise Moyosoluwa | 2022003835 |
| 3. | Peter Olakanmi | 2022006557 |
| 4. | Oyelade Anuoluwa Boluwatife | 2022002090 |
| 5. | Ewuola Olwanifemi Seun | 2022010147 |
| 6. | Akintunde Oluwaseyi David | 2022002447 |
| 7. |  | 2022004228 |
| 8. |  | 2022006612 |
| 9. |  | 2022006486 |
| 10. |  | 2022002454 |

DEDICATION

This SWEP report is dedicated to God almighty, whose overflowing love and infinite mercy helped us through this project, and also to all the lecturers in the department of Cyber Security Science, the Faculty of Computing and Informatics and to the school Ladoke Akintola University of Technology (LAUTECH).

# ACKNOWLEDGEMENT

We place on our deep sense of gratitude to God for his love, mercy, favor, and protection during the Student Work Experience Program (SWEP). We are also grateful to my honorable supervisor MR OJO.

We thank him for being the great person assigned to this wonderful project. Lastly, we express our appreciation to every member of Group 7 for their collective efforts and contributions that led to the fulfillment of this task.

ABSTRACT

File encryption and decryption are essential processes in securing sensitive data from unauthorized access. This paper presents a system for file encryption and decryption using the Advanced Encryption Standard (AES) algorithm, one of the most secure and widely adopted cryptographic methods. AES, a symmetric key encryption algorithm, encrypts data by converting plaintext into ciphertext using a secure key, and the same key is used to decrypt the ciphertext back into plaintext.

The system provides a robust solution for safeguarding data, ensuring both confidentiality and data integrity during storage or transmission. It is implemented using Python and the PyCryptodome library, offering functionalities for key generation, file encryption, and file decryption. The AES algorithm operates efficiently with 128-bit, 192-bit, or 256-bit key lengths, making it suitable for a wide range of applications that require high security.

This paper also addresses the importance of secure key management and explores the implementation challenges, including performance impacts on large files and user handling of encryption keys. The system demonstrates that AES provides a secure, fast, and reliable method for file encryption and decryption, ensuring compliance with data protection regulations and protecting against potential data breaches.

The findings suggest that AES encryption is a practical solution for both personal and organizational data security, offering a balance between high performance and strong cryptographic security.

**TABLE OF CONTENT**

|  |  |
| --- | --- |
| Chapter 1: | Introduction To Python For Cryptography |
| Chapter 2: | File Encryption and Decryption |
| Chapter 3: | Methodology |
| Chapter 4: | Result And Output |
| Chapter 5: | Conclusion |
|  |  |

**CHAPTER ONE**

**INTRODUCTION TO PYTHON FOR CRYPTOGRAPHY**

**What is Python?**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python is a dynamically-typed and garbage-collected programming language.

Guido van Rossum crafted Python between 1985 and 1990, and it is characterized as a dynamically-typed and garbage-collected language. Similar to Perl, Python's source code is available under the GNU General Public License (GPL), ensuring availability on major operating systems such as Linux, Windows, and Mac OS.

Due to its interpreted-based nature, Python is recognized for its relative ease of learning compared to other mainstream languages. It boasts dynamic typing and features intuitive data types. Python is an open-source, cross-platform language, governed by the Python Software Foundation

Python supports multiple programming paradigms, including Procedural, Object Oriented and Functional programming languages.

## Uses & Application Of Python

Python, with its versatility and simplicity, finds extensive applications across various domains. Some notable uses and applications of Python include:

1. Web Development: Python is widely employed for building dynamic websites and web applications.

Frameworks like Django and Flask simplify the development process, making it a popular choice among developers.

1. Data Science and Analytics: Python is a go-to language for data scientists and analysts. Libraries such as NumPy, pandas, and scikit-learn facilitate data manipulation, analysis, and machine learning tasks.
2. Artificial Intelligence (AI) and Machine Learning (ML): Python's extensive libraries, like TensorFlow and PyTorch, make it a preferred language for AI and ML development. It is widely used in developing and training machine learning models.
3. Scientific Computing: Python is embraced in scientific research and engineering for numerical computations and simulations. Libraries like SciPy and Matplotlib support scientific computing and data visualization.
4. Automation and Scripting: Python's simplicity makes it ideal for automation and scripting tasks. It is commonly used for writing scripts to automate repetitive tasks and streamline workflows.
5. Game Development: Python is employed in game development, often for prototyping and scripting. Popular game engines like Pygame provide a platform for creating simple games.
6. Desktop GUI Applications: Python can be used to develop graphical user interface (GUI) applications. Libraries like Tkinter and PyQt enable developers to create desktop applications with ease.
7. Network Programming: Python's robust support for networking makes it suitable for developing network-related applications and tools. Frameworks like Twisted facilitate asynchronous network programming.
8. Cybersecurity: Python is utilized in cybersecurity for tasks such as penetration testing and developing security tools. Its simplicity and extensive libraries make it a valuable resource in this field.

10.Education: Python's readability and simplicity make it an excellent choice for teaching programming. Many educational institutions use Python as an introductory language for programming courses.

The widespread adoption of Python across diverse fields is a testament to its adaptability and effectiveness, making it a go-to language for both beginners and experienced developers.

**CHAPTER TWO**

**FILE ENCRYPTION AND DECRYPTION**

Cryptography The prefix "crypt-" means "hidden" or "vault" -- and the suffix "-graphy" stands for "writing". Cryptography or cryptology is the practice and study of techniques for secure communication in the presence of “adversaries” (the third parties). More generally, cryptography is about burning the message in such a way that prevents third parties or the public from reading private messages.

Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it. There are three types of cryptography method.

1. Symmetric

2. Asymmetric

3. Hybrid

Symmetric Key Cryptography In symmetric key cryptography (also known as private-key cryptography) a secret key may be held by one person or exchanged between the sender and the receiver of a message. If private key cryptography is used to send secret messages between two parties, both the sender and receiver must have a copy of the secret key.

Asymmetric Key Cryptography In the two-key system (also known as the public key system) one key encrypts the information and another, mathematically related key decrypts it. The computer sending an encrypted message uses a chosen private key that is never shared and so is known only to the sender. Using this public-key cryptographic method, the sender and receiver are able to authenticate one another as well as protect the secrecy of the message

File encryption and decryption are fundamental processes in data security, designed to protect sensitive information from unauthorized access. Encryption transforms readable data (plaintext) into an unreadable format (ciphertext) using specific algorithms and keys. Decryption is the reverse process that converts the ciphertext back into plaintext, allowing authorized users to access the original data.

**Importance of File Encryption**

Data Protection: Encryption ensures that sensitive information, such as personal data, financial records, and confidential business information, remains secure from unauthorized access or theft.

Compliance and Regulations: Many industries are subject to regulations (e.g., GDPR, HIPAA) that require the protection of sensitive information. Encryption helps organizations comply with these legal requirements.

Data Integrity: Encryption not only secures data but also helps maintain its integrity by preventing unauthorized modifications.

Confidentiality: Encryption ensures that only authorized users can access and read the data, providing confidentiality and privacy for individuals and organizations.

**How File Encryption Works**

File encryption involves the use of cryptographic algorithms and keys. Here are the key components of the process:

**Encryption Algorithms:**

Algorithms are mathematical procedures used to encrypt and decrypt data. Common encryption algorithms include:

AES (Advanced Encryption Standard): A widely used symmetric encryption algorithm known for its security and efficiency.

RSA (Rivest-Shamir-Adleman): An asymmetric encryption algorithm often used for secure data transmission.

Blowfish: A fast block cipher that is often used in software applications.

Keys: A key is a piece of information used by the encryption algorithm to perform the transformation. Keys can be:

Symmetric Keys: The same key is used for both encryption and decryption. This method is faster but requires secure key management.

Asymmetric Keys: A pair of keys is used: a public key for encryption and a private key for decryption. This method enhances security but is generally slower.

**Modes of Operation:**

For block ciphers like AES, different modes of operation determine how data is encrypted and decrypted. Common modes include:

ECB (Electronic Codebook): Simplest mode, but less secure due to identical plaintext blocks producing identical ciphertext blocks.

CBC (Cipher Block Chaining): Each block of plaintext is XORed with the previous ciphertext block, improving security.

CFB (Cipher Feedback): Allows for encryption of data in smaller increments, suitable for stream encryption.

**The File Encryption Process**

Select an Algorithm: Choose a suitable encryption algorithm based on the security requirements and performance considerations.

Generate or Obtain a Key: For symmetric encryption, generate a secure key. For asymmetric encryption, obtain a public/private key pair.

Prepare the Data: Identify the files or data to be encrypted and prepare them for processing.

Encrypt the Data: Use the selected algorithm and key to encrypt the data, producing ciphertext.

Store or Transmit the Encrypted Data: Save the encrypted file securely or transmit it over a network.

**Challenges in File Encryption**

Key Management: Securely generating, distributing, and storing encryption keys is crucial for maintaining data security.

Performance: Encryption can introduce latency, especially with large files or when using complex algorithms.

Compatibility: Different encryption standards may not be compatible with all systems, requiring careful consideration when sharing encrypted files.

Data Loss: If keys are lost or forgotten, accessing encrypted data can become impossible, leading to potential data loss.

Best Practices for File Encryption

Use Strong Algorithms: Opt for well-established encryption algorithms, such as AES, with sufficient key lengths (e.g., 256 bits).

Implement Robust Key Management: Utilize secure methods for key generation, storage, and access controls. Consider using hardware security modules (HSM) or dedicated key management systems (KMS).

Regularly Update Encryption Methods: Stay updated with the latest cryptographic standards and practices to ensure ongoing security.

Backup Encrypted Data: Ensure that backups of encrypted data are maintained securely and that encryption keys are also backed up in a secure manner.

Educate Users: Provide training to users on the importance of encryption and secure handling of encryption keys.

# CHAPTER THREE

# Methodology

# How to Encrypt/Decrypt

* Open the App and Click SELECT FILE Button and select your file e.g. "abc.jpg".
* Enter your Secret Key (This can be any alphanumeric letters). Remember this so you can Decrypt the file later.
* Click ENCRYPT Button to encrypt. A new encrypted file with ".kryp" extention e.g. "abc.jpg.kryp" will be created in the same directory where the "abc.jpg" is.

When you want to Decrypt a file you, will select the file with the "E/D" extention and Enter your Secret Key which you chose at the time of Encryption.

* Click DECRYPT Button to decrypt. The decrypted file will be of the same name as before with the suffix "dekrypted" e.g. "abc\_\_dekrypted\_\_.jpg".
* Click RESET Button to reset the input fields and status bar.

You can also Click CANCEL Button during Encryption/Decryption to stop the process.

**What is Tkinter**

**Tkinter** is the standard GUI (Graphical User Interface) toolkit for Python. It provides a powerful and easy way to create graphical user interfaces for applications. Tkinter is a wrapper around the Tcl/Tk GUI toolkit and is included with most Python installations, making it readily available for developers.

Key Features of Tkinter

Cross-Platform:

Tkinter applications can run on various operating systems, including Windows, macOS, and Linux, without requiring significant modifications.

Simple and Intuitive:

Tkinter has a straightforward API that allows developers to create windows, dialogs, buttons, labels, and other GUI components easily. Its simplicity makes it a popular choice for beginners and quick application development.

Widget Support:

Tkinter supports various widgets (GUI elements) such as:

* Labels: Display static text or images.
* Buttons: Trigger actions when clicked.
* Entry Fields: Allow user input.
* Text Areas: Provide multi-line text input.
* Frames: Organize widgets within a container.
* Menus: Create dropdown or pop-up menus.
* Canvas: Draw shapes, lines, and images.

Event Handling:

Tkinter supports event-driven programming, allowing applications to respond to user actions (such as mouse clicks, keyboard inputs, etc.) through event binding.

Geometry Management:

Tkinter provides several geometry management options (pack, grid, and place) to control the placement and organization of widgets within the application window.

Theming and Styling:

Tkinter allows customization of widget styles and themes, enabling developers to create visually appealing interfaces.

**What is Pycryptodome**

The PyCryptodome library is a self-contained Python package that provides cryptographic services. It is designed to replace the older PyCrypto library and offers a wide range of cryptographic functionalities that can be used for secure data handling and communication. Here are the main uses and features of the PyCryptodome library:

**Key Features and Uses**

Symmetric Encryption:

PyCryptodome supports various symmetric encryption algorithms, such as:

* AES (Advanced Encryption Standard)
* DES (Data Encryption Standard)
* Triple DES
* Blowfish

These algorithms can be used to encrypt and decrypt data with the same key.

Asymmetric Encryption:

The library supports asymmetric encryption algorithms, including:

* RSA (Rivest-Shamir-Adleman)
* DSA (Digital Signature Algorithm)

**USES OF PYCRYPTODOME**

The library supports HMAC (Hash-based Message Authentication Code), which combines hashing with a secret key to ensure the integrity and authenticity of a message.

**Random Number Generation:**

PyCryptodome includes a secure random number generator, which is essential for creating secure cryptographic keys and nonces.

**Digital Signatures:**

The library supports the creation and verification of digital signatures using asymmetric algorithms, allowing users to verify the authenticity and integrity of messages.

**Key Derivation Functions:**

It offers functions like PBKDF2 (Password-Based Key Derivation Function 2), which is used to derive secure keys from passwords.

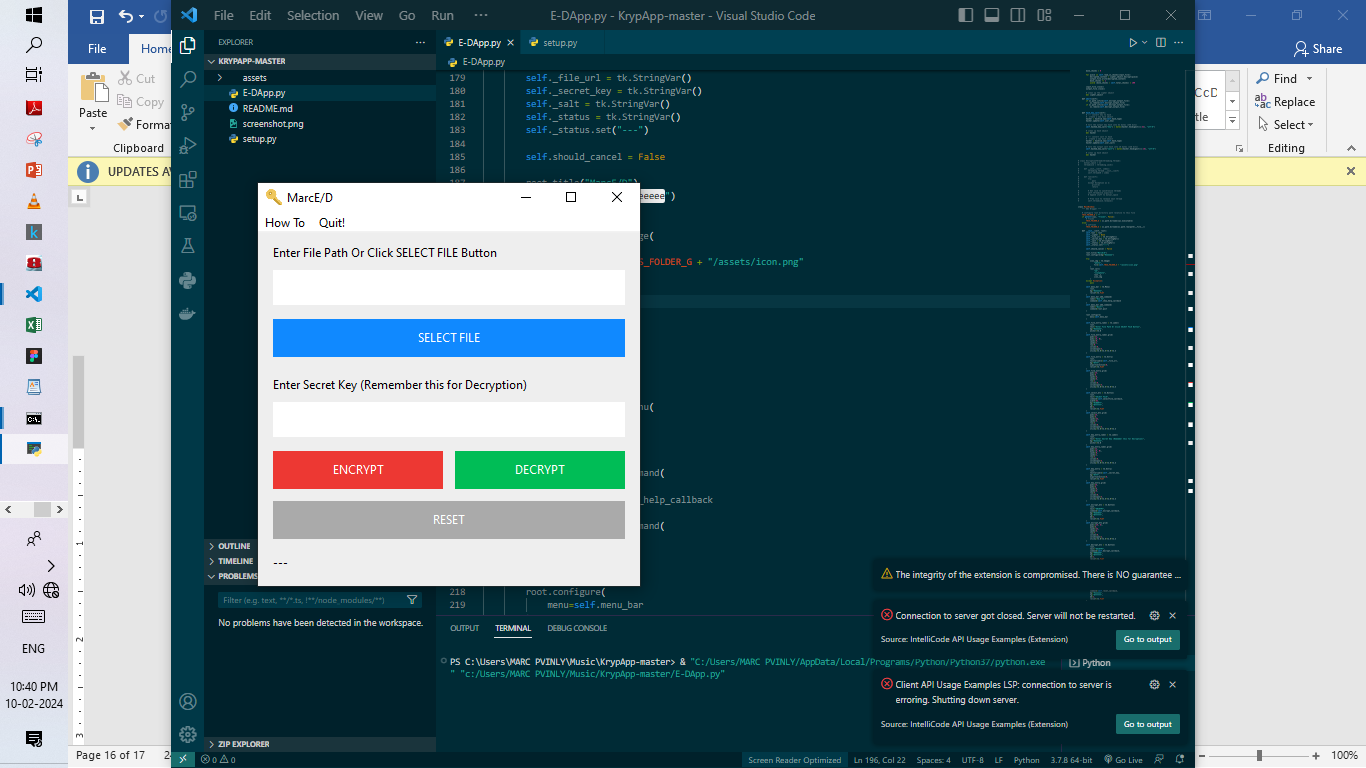
**Public Key Infrastructure (PKI):**

PyCryptodome can be used to work with X.509 certificates and support operations related to public key infrastructure.

**CHAPTER FOUR**

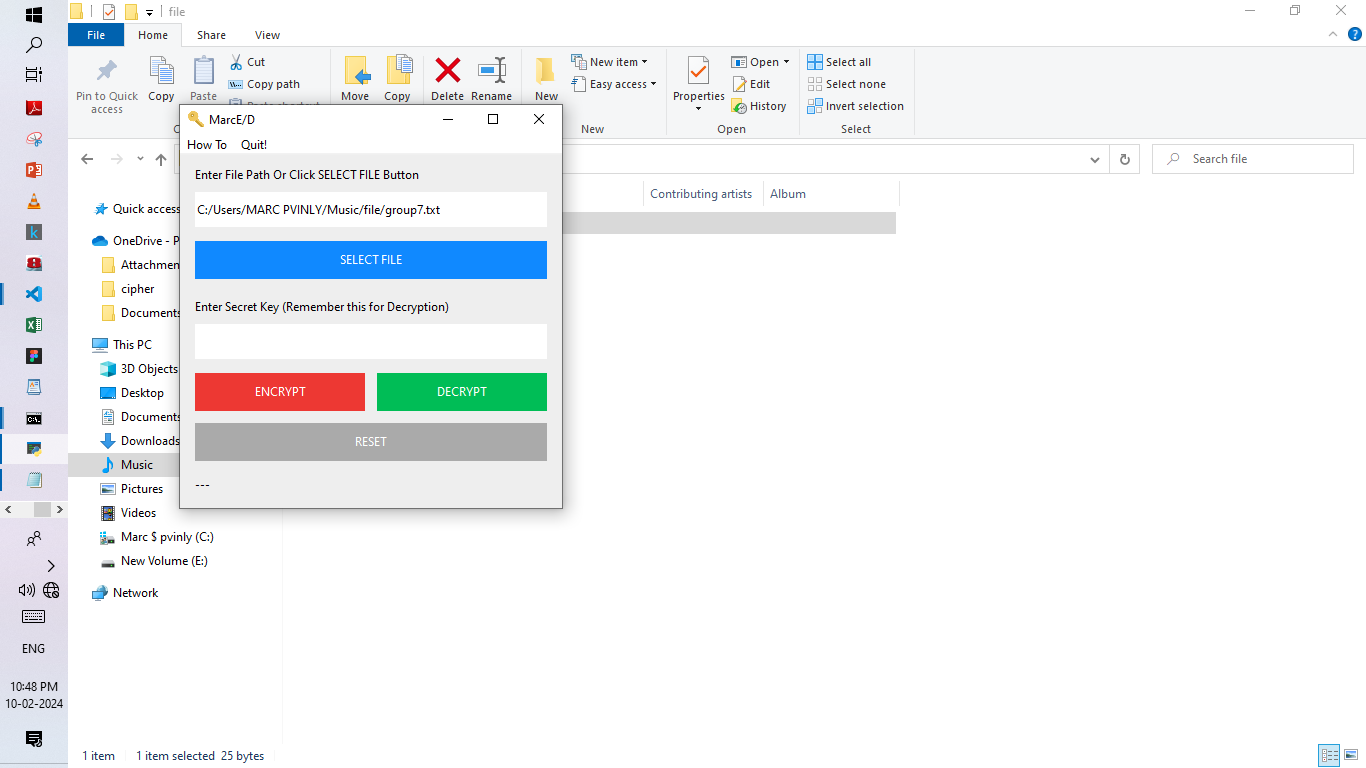
# RESULT AND OUTPUT

**Application GUI (Graphical User Interface)**

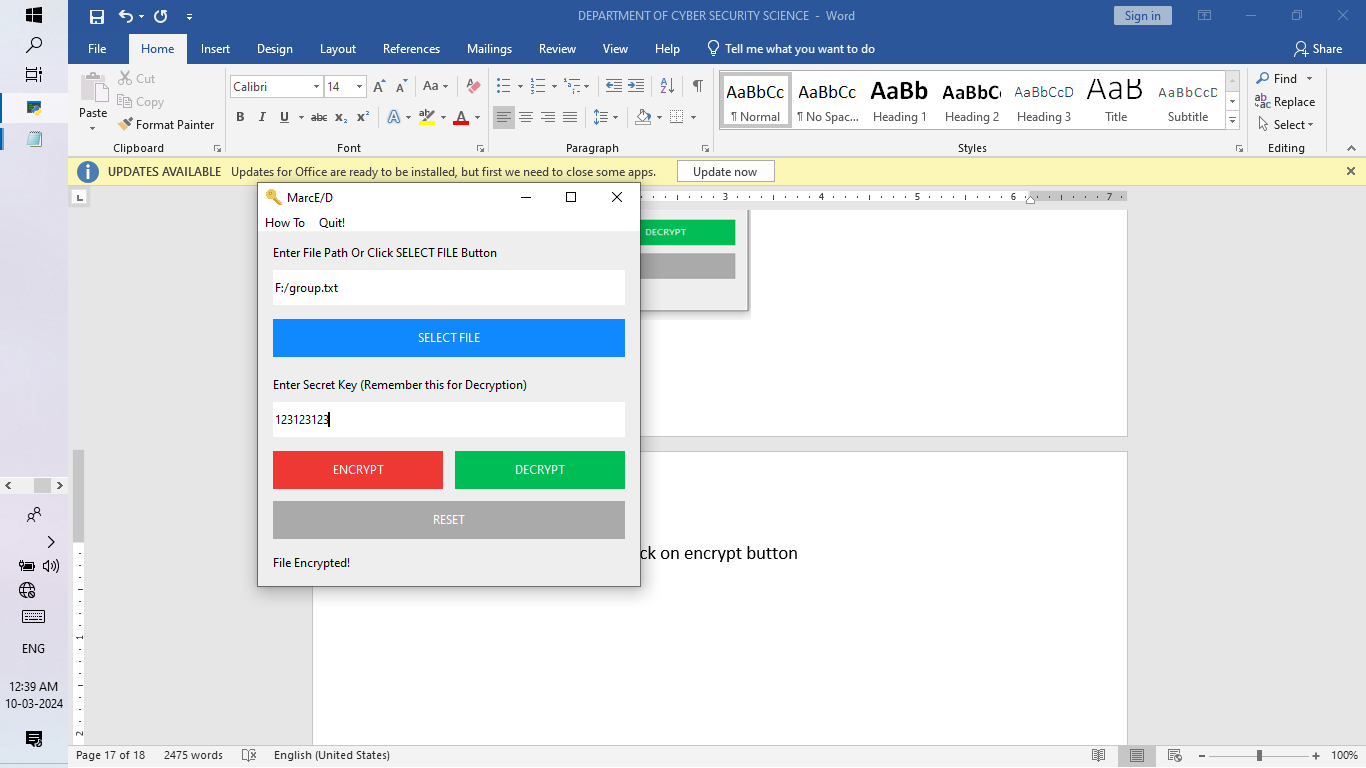


**How to encrypt a file**

Click on select file to pick the file you want to encryption

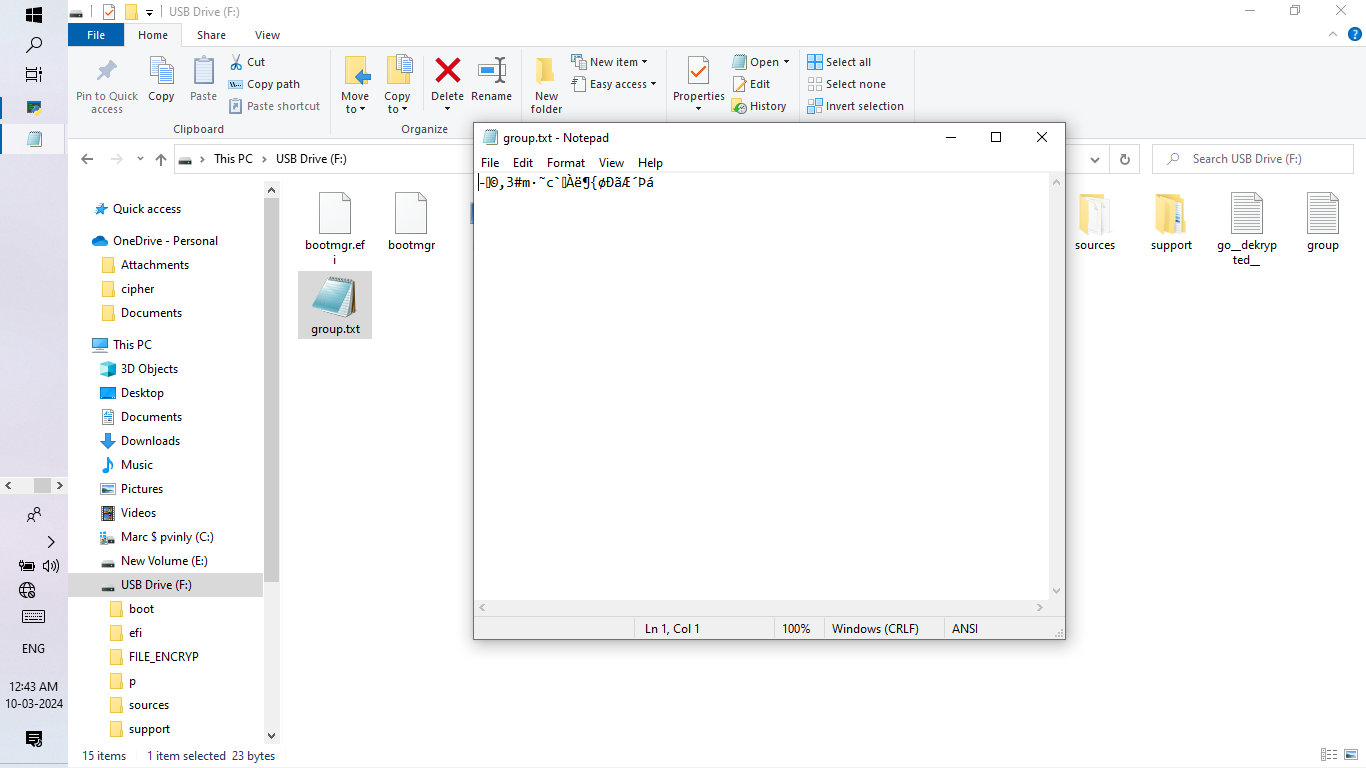


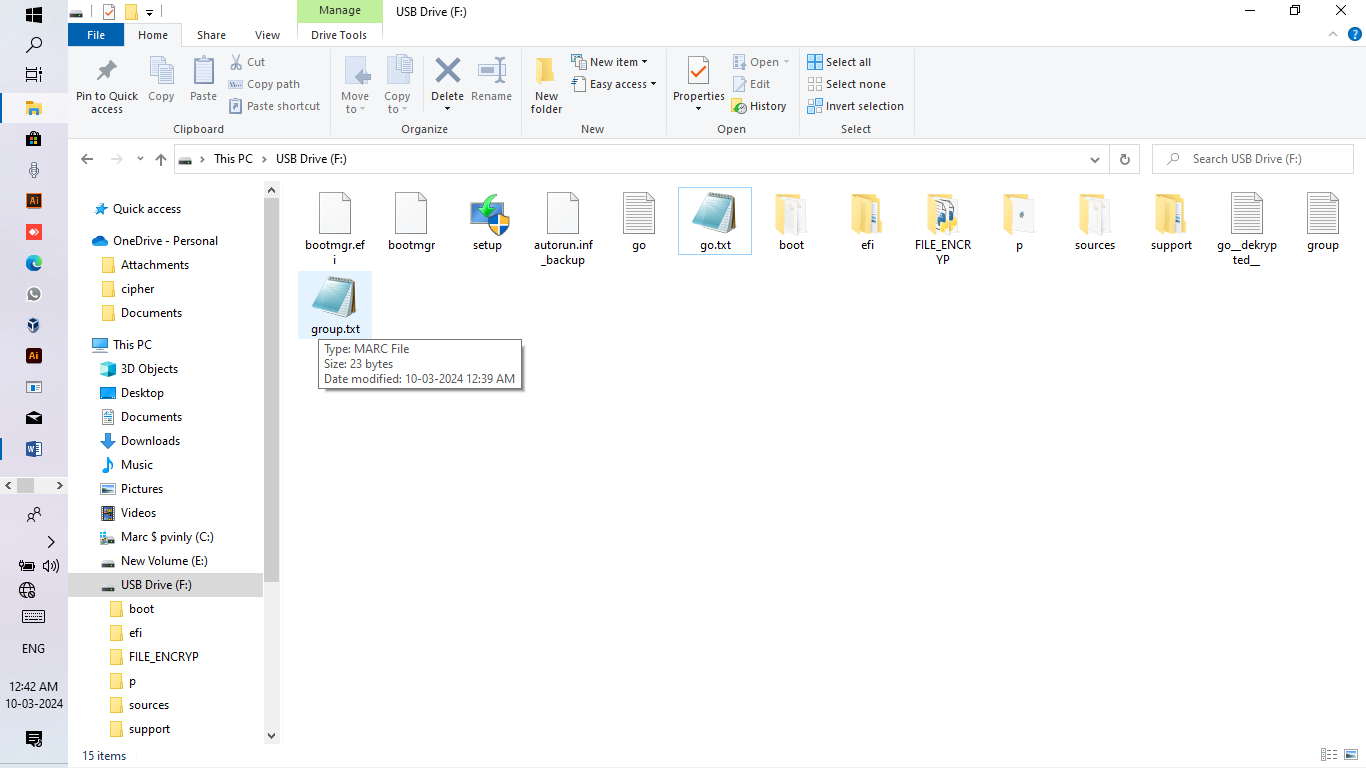
Provide a password and click on encrypt button



It will then provide a message “file Encrypted”.

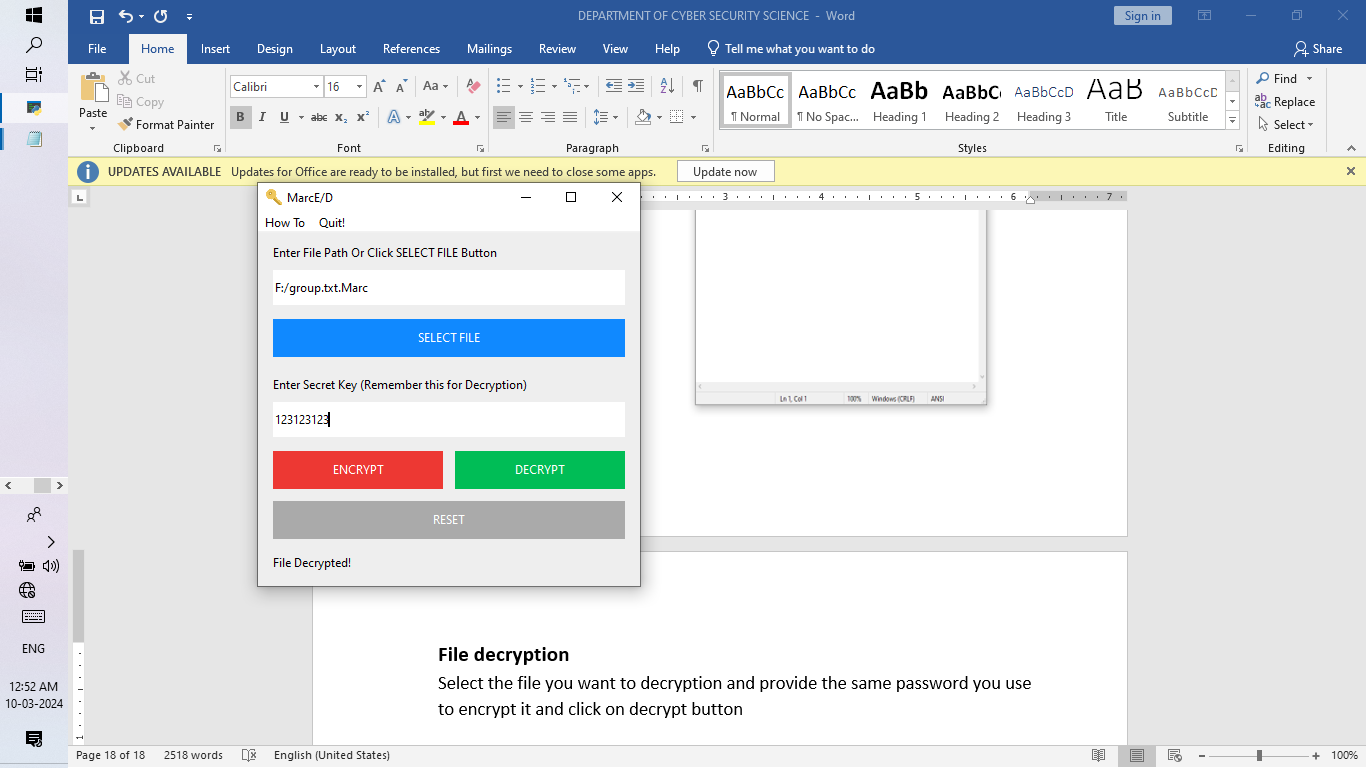
**Image of Encrypted file**



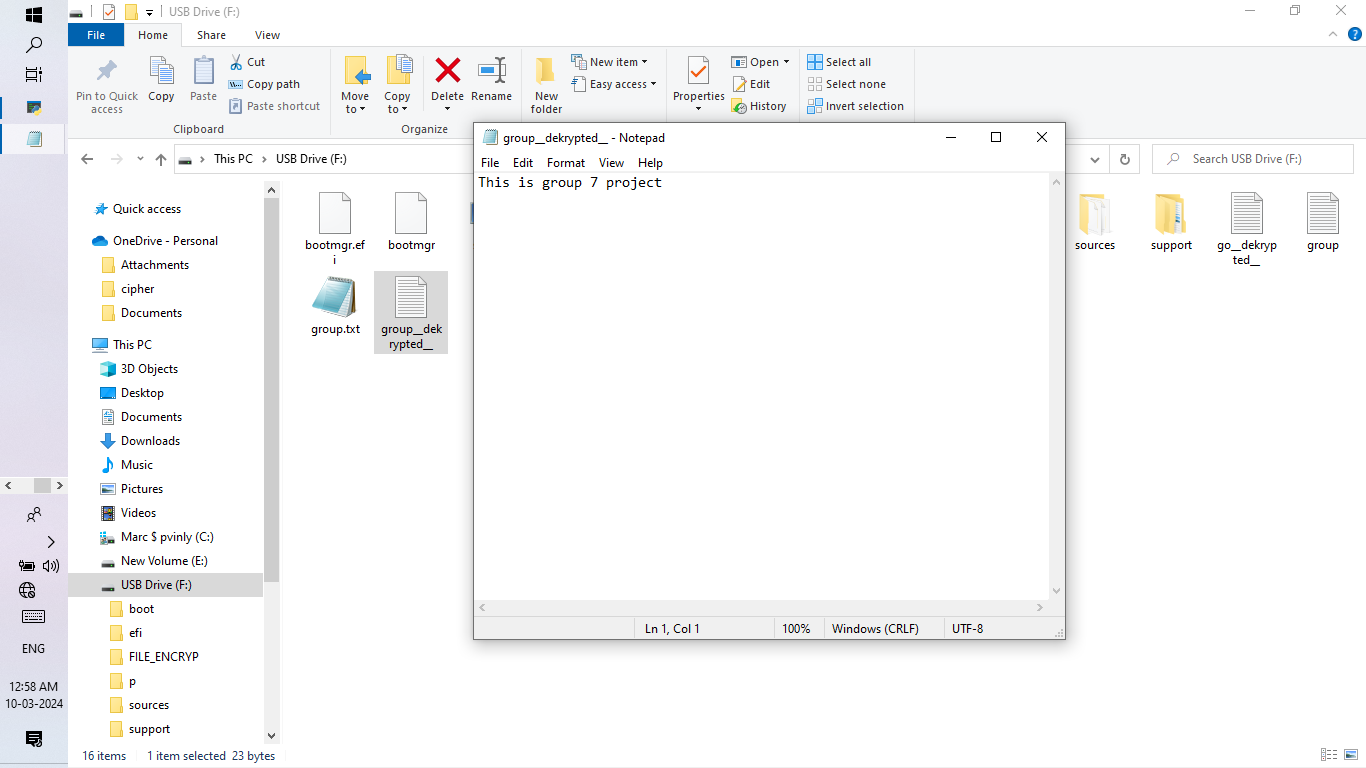


**File decryption**

Select the file you want to decryption and provide the same password you use to encrypt it and click on decrypt button



**Image of Decrypted file**



**CHAPTER FIVE**

**CONCLUTION**

File encryption and decryption are vital components of modern data security strategies. By understanding the principles of encryption, organizations and individuals can protect sensitive information from unauthorized access, maintain compliance with regulations, and ensure the integrity and confidentiality of their data. As cyber threats evolve, the importance of robust encryption practices will continue to grow, making it essential for everyone to stay informed and proactive about data protection measures.