**LAB 3**

**(Encryption)**

Step 1: Install Required Library

A screenshot of a computer

AI-generated content may be incorrect.

**Step 2: Encryption Program**

* Read a plaintext file.
* Encrypt the content using AES.
* Write the encrypted data to a new file.

**Step 3: Decryption Program ( my partners submission )**

* Read the encrypted file.
* Decrypt the content back into plaintext.
* Write the decrypted data to a third file.

**Step 4: Testing**

* Verify that the decrypted file matches the original plaintext file.

**Python program :**

from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes

from cryptography.hazmat.backends import default\_backend

import os

def encrypt\_file(input\_file, encrypted\_file, key, iv):

with open(input\_file, 'r') as file:

plaintext = file.read().encode('utf-8')

# Ensure the plaintext is a multiple of 16 bytes (AES block size)

while len(plaintext) % 16 != 0:

plaintext += b' '

cipher = Cipher(algorithms.AES(key), modes.CFB(iv), backend=default\_backend())

encryptor = cipher.encryptor()

encrypted\_data = encryptor.update(plaintext) + encryptor.finalize()

with open(encrypted\_file, 'wb') as file:

file.write(encrypted\_data)

# Generate a secure 32-byte key and 16-byte IV for AES encryption

key = os.urandom(32) # AES-256 key

iv = os.urandom(16) # Initialization Vector (IV)

# Save key and IV for the decryption process

with open('key\_iv.txt', 'wb') as file:

file.write(key + iv)

# Input/Output files

encrypt\_file('plaintext.txt', 'encrypted.txt', key, iv)

print("Encryption complete! Encrypted data written to 'encrypted.txt'")