import os

from base64 import b64encode, b64decode

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

from cryptography.hazmat.backends import default\_backend

from dotenv import load\_dotenv

# Load encryption key from environment variable

load\_dotenv()

AES\_KEY = os.getenv("NOT\_MY\_KEY").encode() # Must be 32 bytes for AES-256

def encrypt\_private\_key(private\_key: str) -> str:

iv = os.urandom(16) # Generate a random IV (16 bytes)

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

encryptor = cipher.encryptor()

ciphertext = encryptor.update(private\_key.encode()) + encryptor.finalize()

return b64encode(iv + ciphertext).decode()

def decrypt\_private\_key(encrypted\_data: str) -> str:

decoded\_data = b64decode(encrypted\_data)

iv = decoded\_data[:16]

ciphertext = decoded\_data[16:]

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

decryptor = cipher.decryptor()

return (decryptor.update(ciphertext) + decryptor.finalize()).decode()