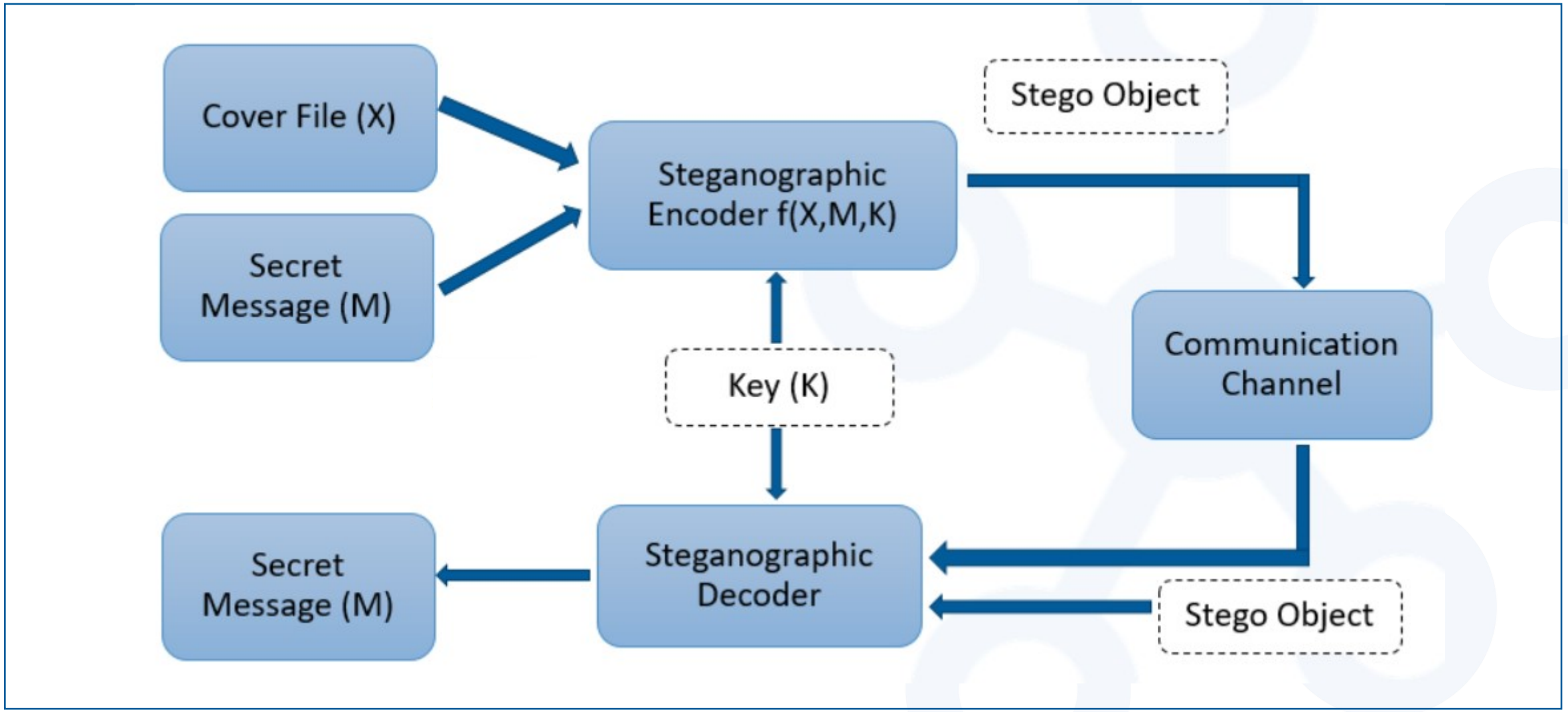
Team & Challenge Details

Team Name	:	TECHI TECH
Team ID	:	GZN1209
Team Leader Name	:	Kopur Prathyusha
Team Member Name	:	Guvvala Hima Bindu
Challenge ID	:	120
Challenge Name	:	Hidden Shield: Secret Data Encryption

Steganography: Hiding Secrets in Plain Sight

- **Definition**: Hiding secret information within a non-secret message, image, or audio
- **How it works**: Embedding secret message into a cover medium using steganography algorithms
- **Types**: Image, Text, Audio Steganography
- **Techniques**: LSB Substitution, Frequency Domain, Spread Spectrum
- **Applications**: Secure Communication, Digital Watermarking, Covert Communication
- **Challenges**: Detection, Capacity, Robustness

Flowchart Diagram



Process

1. The secret message is first encrypted to protect its confidentiality.
2. The encrypted message is then converted into a binary format suitable for hiding within an image.
3. The binary data is embedded into the image by modifying pixel values.
4. The resulting image, along with the encryption key, is used to recover the original message.
5. This process combines encryption and steganography to provide a secure way to hide information within an seemingly ordinary image.

Characters per Pixel Image

Image Size	Pixels	Bits (3/channels/pixel	Potential character (8 bits/character)	Approximate words (5 characters/word)
320 X 240	76800	230400	28800	5760
640 X 480	307200	921600	115200	23040
1204 X 768	786432	2359296	294912	58982
1920 X 1080	2073600	6220800	777600	155520

Commands for Encryption and Encoding

- from cryptography.fernet import Fernet: Import Fernet class for encryption
- from PIL import Image: Import Image class for image processing
- import streamlit as st: Import Streamlit library for web application
- import base64: Import base64 library for encoding and decoding
- import io: Import io library for in-memory binary streams
- import zipfile: Import zipfile library for creating ZIP files
- def write_key(): Define function to generate encryption key
- def encrypt_message(message: str, key: bytes) -> bytes: Define function to encrypt message
- def text_to_bits(text, encoding='utf-8', errors='surrogatepass'): Define function to convert text to binary bits
- def encode_image(image, message): Define function to encode encrypted message into image

Commands for Decryption and Decoding

- from PIL import Image: Import Image class for image processing
- from cryptography.fernet import Fernet: Import Fernet class for decryption
- import streamlit as st: Import Streamlit library for web application
- def bits_to_text(bits, encoding='utf-8', errors='surrogatepass'): Define function to convert binary bits to text
- def decode_image(image): Define function to decode encrypted message from image
- def decrypt_message(encrypted_message: str, key: bytes) -> str: Define function to decrypt message

Commands for Streamlit Application

- import streamlit as st: Import Streamlit library for web application
- from encrypt_and_encode import app as encrypt_app: Import encrypt_and_encode application function
- from decode_and_decrypt import app as decrypt_app: Import decode_and_decrypt application function
- st.set_page_config(page_title="Steganography App", page_icon="", layout="wide"): Set page configuration for Streamlit app
- tabs = st.tabs(["Encrypt and Encode", "Decode and Decrypt"]): Create two tabs in Streamlit app
- with tabs[0]: encrypt_app(): Run encrypt_and_encode application function in first tab
- with tabs[1]: decrypt_app(): Run decode_and_decrypt application function in second tab

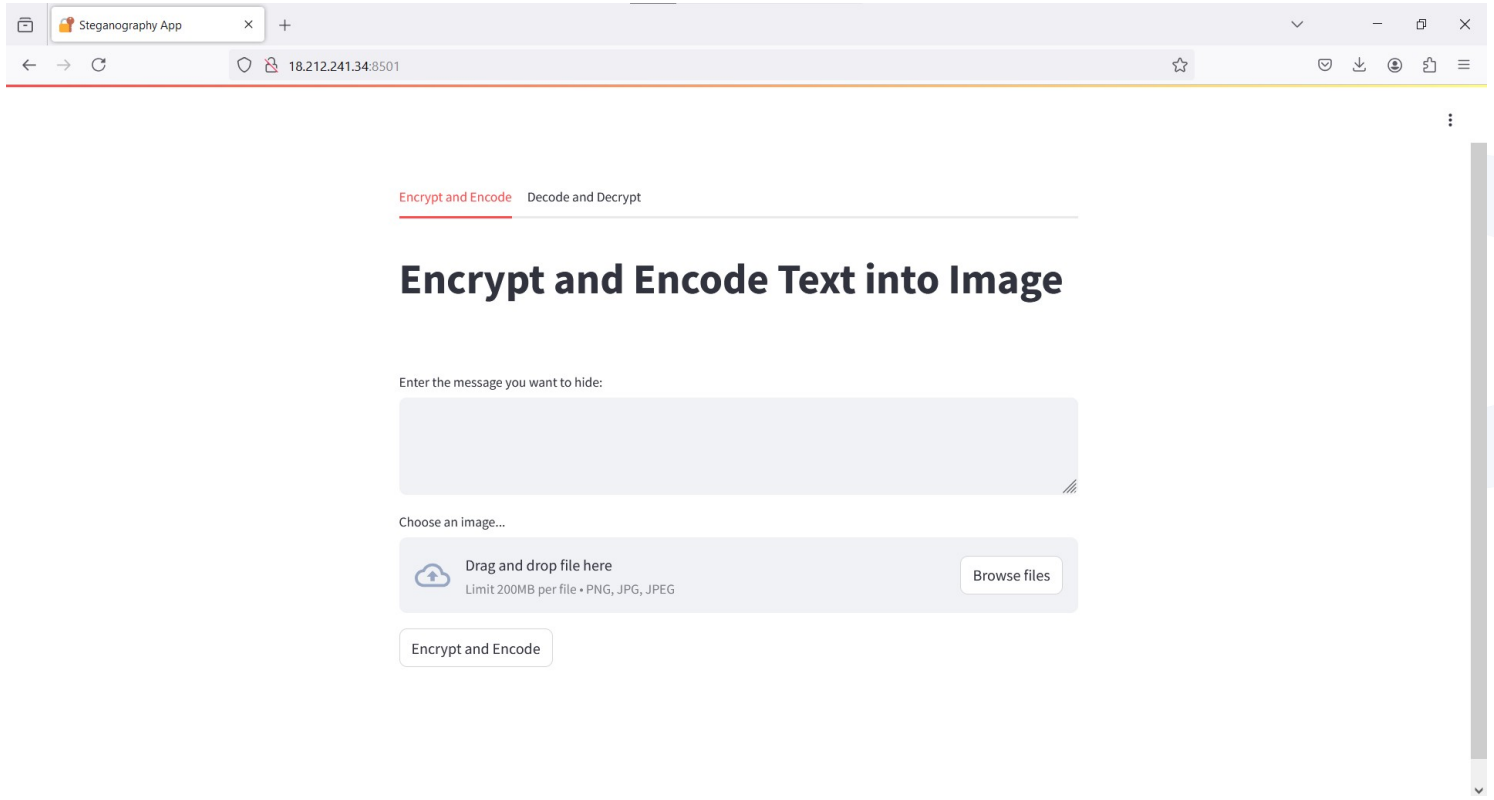
1024 X 768 Images

Before Encoding and Encryption

After Encryption

Steganography App: Encrypt and Encode Panel

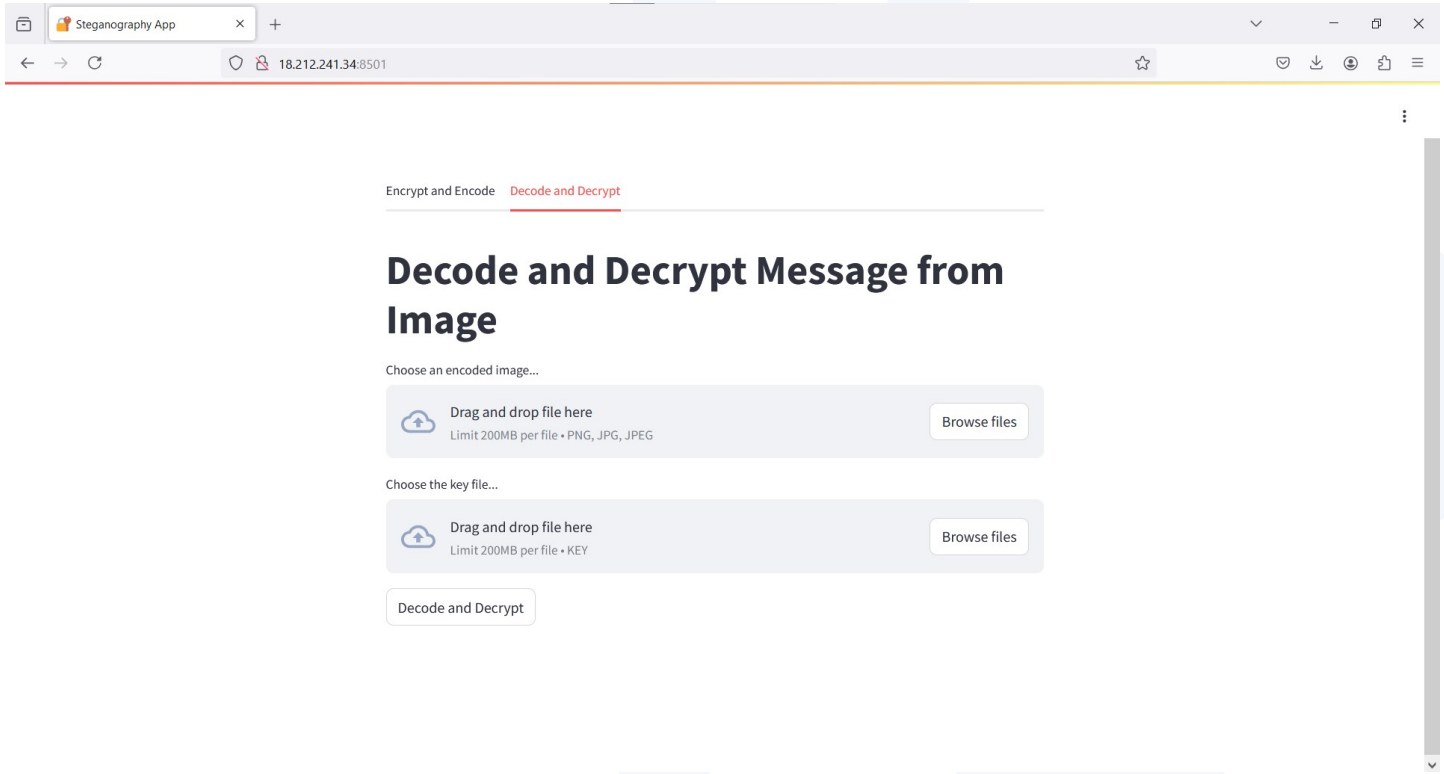
Web Link: http://18.212.241.34:8501/



1. Enter your secret message in the provided dialogue box and select an image by either dragging and dropping it or browsing files.
2. Click the "Encrypt and Encode" button to encrypt the message and encode it into the image.
3. Download the generated zip file containing the encrypted message, encoded image, and system-generated key.

Steganography App: Decode and Decrypt Panel

1. Upload the encoded image by dragging and dropping the file into the designated area or by browsing files.
2. Upload the key file by dragging and dropping it into the designated area or by browsing files.
3. Click the "Decode and Decrypt" button to extract and decrypt the hidden message from the image.



THANK YOU!