REVIEW3 – CHAT APPLICATION FOR SENDING INFORMATION USING STEGANOGRAPHY TECHNIQUES FOR DIGITAL IMAGES

CSE3502 – INFORMATION SECURITY MANAGEMENT - EPJ

MR SIVA SHANMUGAM G

TEAM

1. ANIRUDH VADERA(20BCE2940)
2. AYUSH DWIVEDI(20BCE2939)

APRIL 14,2023

TITLE: CHAT APPLICATION FOR SENDING INFORMATION USING STEGANOGRAPHY TECHNIQUES FOR DIGITAL IMAGES

ABSTRACT:

MOTIVATION:

With the great popularity and expansion in digital world technologies and the rapid growth of the network, the Internet has turned out to be a commonly used channel for communication of all forms of data such as audio, video, image, and text, digitally. There are many cryptography and encryption algorithms available, but not all are accessible to everyone and there can be many vulnerabilities. With steganography, most of these problems are eliminated. Steganography is the technique of hiding text, images, audio or even video in other images or any other digital content, such that it is very difficult or near impossible to detect that some data has been hidden into an image. The use of steganography algorithms can be embedded with encryption as an additional step for hiding or protecting sensitive data.

INTRODUCTION:

OBJECTIVES:

● To understand various steganography and encryption techniques and how it is implemented.

● To actually implement steganography techniques to hide text and images into other images.

● To create a website which will enable users to use steganography techniques and hide their secret message or text into an image of their choice.

● To extract the data hidden into images with no error or loss of data.

● To create a Chat Application to send information encrypted using steganography techniques

COURSE:

To overcome the vulnerabilities of cryptography or general encryption, steganographic techniques are suggested to hide the data in such a manner that no one other than the sender and recipient even recognizes that there is some hidden data in it.

● Unlike the other forms of communication and encryption, the main purpose of steganography is defeated when the communication between 2 users is detected.

● Therefore, the prime motive of a steganographic structure is its undetectability.

● The basic outline of the project lies with a website which enables a user to access these steganography techniques.

● The core of the project lies with successful implementation of these techniques.

● We implemented **Least Significant Bit Substitution (text within image)** for our cause

● Next step was to create a working website with appropriate front end and back end.

● Finally we will be implementing a Chat application for our purpose of sending information encrypted using steganography techniques. Now this information will be sent into a group basically but will only be readable by persons who are correct owners to extract this information.

PROPOSED METHODOLOGY:

Sockets:

Define System Requirements: Define the requirements of the proposed system, including the desired functionalities, performance benchmarks, security measures, and user interface. This will guide the development process and ensure that the final product meets the expectations of stakeholders.

Select a Steganography Technique: Select an appropriate steganography technique, such as LSB steganography o and develop Python scripts to embed messages or data into images.

Implement Socket Programming: Implement the socket programming functionality in JavaScript to enable clients to send and receive steganographic images or messages securely. This includes implementing the different protocols and standards used in socket programming and how they can be integrated with the steganography functionality.

Integrate Steganography and Socket Programming: Integrate the steganography scripts with the JavaScript socket programming functionality to enable the clients to send and receive steganographic images or messages securely. This requires careful consideration of how the steganography functionality will be triggered and how the resulting images or messages will be transmitted through the socket connections.

Implement Security Measures: Implement additional security measures to ensure that the system is secure and protected against potential threats. This includes encryption of data transmitted over the socket connections and authentication of clients to prevent unauthorized access.*

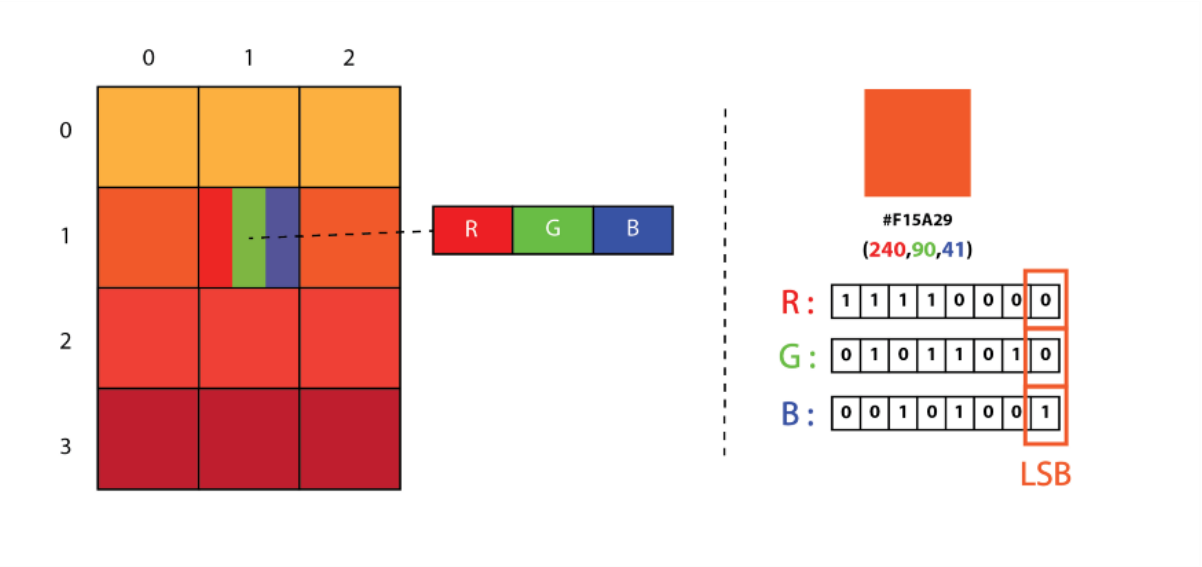
Make a proper chat app that supports all that functionality. Providing the user with a good UI and a convenient and safe way to communicate.

In socket programming:

- First, you would need to create a server-side script that listens for incoming connections from clients.
- Once a client connects to the server, the server-side script would create a socket connection with the client. This connection would be used to transmit data back and forth between the client and the server.
- To implement steganography, you would modify the client-side script to first encode a message or data into an image file using a steganography algorithm. Once the message is encoded, the client would then send the image file to the server over the socket connection.
- On the server-side, the server would receive the image file and decode the message or data from the image using the same steganography algorithm that was used to encode the message. The server could then perform any necessary processing on the message before sending a response back to the client over the same socket connection.
- To ensure security, you would need to implement encryption of the data transmitted over the socket connection, and authentication of clients to prevent unauthorized access to the server.

LSB Steganography:

LSB Steganography is an image steganography technique in which messages are hidden inside an image by replacing each pixel's least significant bit with the bits of the message to be hidden.
To understand better, let's consider a digital image to be a 2D array of pixels. Each pixel contains values depending on its type and depth. We will consider the most widely used modes — **RGB(3x8-bit pixels, true-color)** and **RGBA(4x8-bit pixels, true-color with transparency mask).** These values range from 0–255, (8-bit values).



We can convert the message into decimal values and then into binary, by using the ASCII Table. Then, we iterate over the pixel values one by one, after converting them to binary, we replace each least significant bit with that message bits in a sequence.
To decode an encoded image, we simply reverse the process. Collect and store the last bits of each pixel then split them into groups of 8 and convert it back to ASCII characters to get the hidden message.

Advantages of LSB Steganography:

The advantages of Least-Significant-Bit (LSB) steganographic data embedding are that it is simple to understand, easy to implement, and it results in stego-images that contain hidden data yet appear to be of high visual fidelity

- This method is very fast and easy to implement in comparison to other methods of image Steganography.
- The output image has very slight difference to the input image.
- Instead of embedding the message in only the LSB, we can embed the message in last two LSBs, thus embedding even large messages.
- This method forms the basics of many other complex algorithms
- Instead of embedding the message in only the LSB, we can embed the message in last two LSBs, thus embedding even large messages.

Disadvantages of LSB Steganography:

It can be shown that under certain conditions, LSB embedding is not secure at all. The fatal drawback of LSB embedding is the existence of detectable artifacts in the form of pairs of values (PoVs).

- This type of encoding the data is weak since it can be easily decoded by taking the LSBs of the image and getting the message in binary format.
- This is method is too old because it was used long ago when other encoding methods were not yet developed.
- When embedding the message in more than one LSB, the image quality may reduce depending on how many pixels are changed.

Some Better Steganography Techniques:

There are several steganography techniques that are more secure and robust than the LSB method. Here are some of them:

Spread Spectrum Steganography:

Spread Spectrum Steganography is a technique that spreads the message over multiple frequencies in a way that it becomes difficult to detect it by an attacker. It uses the properties of the frequency domain to hide the message by spreading it over a range of frequencies. This method is more secure than LSB since the message is distributed over multiple frequencies and not confined to a single location.

Statistical Steganography:

Statistical Steganography hides the message by modifying the statistical properties of the cover image. It analyzes the image's statistical properties and then embeds the message in such a way that it does not change the statistical properties significantly. The method is more secure since it is difficult to detect changes in the statistical properties of the cover image.

Transform Domain Steganography:

Transform Domain Steganography is a technique that hides the message in the transform domain of the cover image. It transforms the cover image into a different domain, such as the wavelet domain or Fourier domain, and embeds the message in the transform coefficients. This method is more secure than LSB since it is difficult to detect the changes in the transform domain.

Distortion Steganography:

Distortion Steganography embeds the message in the cover image by introducing controlled distortions. It changes the color or luminance of pixels or introduces noise in the image to hide the message. The method is more secure than LSB since the changes are introduced in a way that is difficult to detect.

Video Steganography:

Video Steganography is a technique that hides the message in the video stream. It can be done by modifying the frames of the video or by embedding the message in the audio track. The method is more secure than LSB since it is difficult to detect changes in the video stream.

RESEARCH GAP:

The research gap is the lack of consideration for potential attacks and vulnerabilities in the proposed system. While steganography can provide a high level of security and privacy, it is not entirely foolproof, and there is always a risk of an attack that can compromise the hidden data's confidentiality. Therefore, future research can focus on identifying and addressing potential attacks and vulnerabilities in steganography-based communication systems. Additionally, the proposed chat application's scalability and usability can also be considered in future research to ensure that it can accommodate a large number of users while maintaining a high level of security and user-friendliness.

LITERATURE SURVEY:

PAPER	SUMMARY	METHODOLOGY	RESULT	CONCLUSION
Image steganography using uncorrelated color space and its application for security of visual contents in online social networks	The paper presents a new approach for image steganography using uncorrelated color space and its application for security of visual contents in online social networks.	The proposed approach uses an uncorrelated color space to embed secret information into the cover image. The approach consists of three main steps: color decorrelation, embedding, and extraction. The embedding process involves modifying the least significant bits of the color space to embed secret information.	The proposed approach was evaluated using several standard image quality metrics, and the results show that the proposed approach achieves good results in terms of hiding capacity, imperceptibility, and robustness against various image processing attacks.	The proposed approach can be used for secure communication of visual contents in online social networks. The experimental results show that the proposed approach provides better performance compared to other existing approaches in terms of hiding capacity, imperceptibility, and robustness against various image processing attacks.
Steganography- A data hiding technique. International Journal of Computer Applications	The paper provides an overview of steganography as a data hiding technique.	The paper provides a literature review of various steganographic techniques, including image steganography, audio steganography, and text	The paper does not present experimental results, as it is a review article.	The paper concludes that steganography is a useful technique for secure communication and can be used in various applications, including digital watermarking, copyright

		steganography. The paper also discusses the challenges associated with steganography, including detection and prevention.		protection, and secure communication. The paper also highlights the need for further research in steganography to improve its effectiveness and security.
image steganography for authenticity of visual contents in social networks.	The paper proposes a new approach for image steganography for the authenticity of visual contents in social networks.	The proposed approach uses a two-stage encryption and embedding scheme to embed secret information into the cover image. The first stage involves encrypting the secret message using a symmetric key algorithm, while the second stage involves embedding the encrypted message into the cover image using a modified version of LSB steganography.	The proposed approach was evaluated using several standard image quality metrics, and the results show that the proposed approach achieves good results in terms of hiding capacity, imperceptibility, and robustness against various image processing attacks.	The proposed approach can be used to ensure the authenticity of visual contents in social networks by embedding a digital signature or watermark into the image. The experimental results show that the proposed approach provides better performance compared to other existing approaches in terms of hiding capacity, imperceptibility, and robustness against various image processing attacks.

Enhanced digital image and text data security using hybrid model of LSB steganography and AES cryptography technique	The paper proposes an enhanced approach for digital image and text data security using a hybrid model of LSB steganography and AES cryptography technique.	The proposed approach involves using LSB steganography to embed a secret message into the cover image, and AES cryptography to encrypt the message before embedding. The approach was evaluated using standard image quality metrics and cryptographic measures, such as PSNR and encryption time.	The experimental results show that the proposed approach achieves good results in terms of hiding capacity, imperceptibility, and security. The approach provides better performance compared to other existing approaches in terms of PSNR and encryption time.	The proposed approach can be used for secure communication of digital image and text data. The experimental results show that the proposed approach provides better performance compared to other existing approaches in terms of security and computational efficiency. The proposed approach can be used in various applications, including online communication and data storage.
Triple-A: Secure RGB image steganography based on randomization.	The paper proposes a new approach for secure RGB image steganography based on randomization, called Triple-A.	The proposed approach uses a three-stage encryption and embedding scheme to embed secret information into the cover image. The first stage involves	The experimental results show that the proposed approach provides better performance compared to other existing	The proposed approach can be used for secure communication of visual contents by embedding a secret message into the cover image. The experimental results show that the

		randomizing the image pixels using a pseudo-random number generator. The second stage involves embedding the secret message into the randomized image using a modified version of LSB steganography. The third stage involves re-randomizing the pixels to enhance security.	approaches in terms of hiding capacity, imperceptibility, and robustness against various image processing attacks.	proposed approach provides better performance compared to other existing approaches in terms of security and computational efficiency. The proposed approach can be used in various applications, including digital watermarking, copyright protection, and secure communication.
An efficient image cryptography using hash-LSB steganography with RC4 and pixel shuffling encryption algorithms.	The paper proposes a new efficient image cryptography approach using hash-LSB steganography with RC4 and pixel shuffling encryption algorithms.	The proposed approach uses hash-LSB steganography to embed a secret message into the cover image, RC4 encryption to encrypt the message, and pixel shuffling encryption to shuffle the pixel values of the image to enhance security. The	The experimental results show that the proposed approach provides good security and computational efficiency. The approach provides better performance compared to other existing approaches in	The proposed approach can be used for secure communication and storage of digital images. The experimental results show that the proposed approach provides better performance compared to other existing approaches in terms of security and computational efficiency. The

		approach was evaluated using standard cryptographic measures, such as encryption time and key space.	terms of encryption time and key space.	proposed approach can be used in various applications, including online communication and data storage.

CODE SCREENSHOTS:

Basic HTML:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>CHAT APP</title>
    <meta name="description" content="" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="./css/styles.css" />
  </head>
  <body>
    <div id="snackbar">Some text some message..</div>
    <div class="appcontainer flex-column flex-center">
      <header>
        <div class="container flex-column flex-center">
          <h1 class="title">Group Chat</h1>
          <p class="light-text">
            Please Provide Your Message to Encrypt or the encrypted image
            itself.
          </p>
        </div>
      </header>
      <div class="main-content container flex-grow-1 flex-column">
        <div class="login">
          <h2 class="text-center">Get Started</h2>
          <form action="" class="login-form flex">
            <input
              type="text"
              name="userName"
              class="userNameInput flex-grow-1"
```

```
            placeholder="Enter Your Username / (Anonymous)"
          />
          <button class="loginBtn">Chat!!</button>
        </form>
      </div>
      <div class="chat flex-grow-1 flex-column hidden">
        <div class="message-list flex-grow-1"></div>
        <form action="" class="messageForm flex">
          <input
            type="text"
            name="message"
            placeholder="Enter a message"
            class="messageInput flex-grow-1"
          />
        </form>
        <form action="" class="imageForm flex">
          <button class="imageInput flex-grow-1">Choose File</button>
          <input
            type="file"
            class="flex-grow-1"
            name="imageInput"
            style="display: none"
            id="getFile"
          />
          <button class="sendBtn2">Send</button>
        </form>
      </div>
    </div>
  </div>
  <div class="loginPopup">
    <div class="formPopup" id="popupForm">
      <h1>HI</h1>
    </div>
  </div>
  <script
    src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.2.0/socket.io.js
"
    integrity="sha512-
WL6WGKMPBiM9PnHRYIn5YEtq0Z8XP4fkVb4qy7PP4vhmYQErJ/dySyXuFIMDf1eEYCXCrQrMJfkNwK
c9gsjTjA=="
    crossorigin="anonymous"
    referrerpolicy="no-referrer"
  ></script>
  <script src="./app.js" async defer></script>
 </body>
</html>
```

Login:

```
// loginBtn
loginBtn.addEventListener("click",(event)=>{
    event.preventDefault();
    if(!usernameInput.value)
    {
        return console.log("must supply a username");
    }
    username=usernameInput.value;
    loginWindow.classList.add("hidden");
    chatWindow.classList.remove("hidden");
    sendMessage(
        {
            author: username,
            type: messageTypes.LOGIN
        }
    );
    console.log(username);
    document.querySelector(".title").innerHTML = "Group Chat " + username;

});
```

What happens when a image is clicked:

```
function sendMessage(message)
{
    socket.emit("message",message);
}

      You, last month • Final Project …
function imageClicked(event){
    event.preventDefault();
    imageUniqueKey = "encrypted_img" + event.target.dataset.name
    fetch("/" + imageUniqueKey).then((res)=>{
        return res.json()
    }).then((data)=>{
        var x = document.getElementById("snackbar");
        // Add the "show" class to DIV
        x.className = "show";
        x.innerHTML = data.data
        setTimeout(function(){ x.className = x.className.replace("show", ""); }, 5000);
    })
}
```

Steganography decode function:

```
def decode(image_name):
    print("[+] Decoding...")
    # read the image
    image = cv2.imread(image_name)
    binary_data = ""
    for row in image:
        for pixel in row:
            r, g, b = to_bin(pixel)
            binary_data += r[-1]
            binary_data += g[-1]
            binary_data += b[-1]
    # split by 8-bits
    all_bytes = [binary_data[i: i + 8] for i in range(0, len(binary_data), 8)]
    # convert from bits to characters
    decoded_data = ""
    for byte in all_bytes:
        decoded_data += chr(int(byte, 2))
        if decoded_data[-5:] == "=====":
            break
    return decoded_data[:-5]
```

Steganography encode function:

```
def encode(image_name, secret_data):
    # read the image
    image = cv2.imread(image_name)
    # maximum bytes to encode
    n_bytes = image.shape[0] * image.shape[1] * 3 // 8
    print("[*] Maximum bytes to encode:", n_bytes)
    if len(secret_data) > n_bytes:
        raise ValueError(
            "[!] Insufficient bytes, need bigger image or less data.")
    print("[*] Encoding data...")
    # add stopping criteria
    secret_data += "====="
    data_index = 0
    # convert data to binary
    binary_secret_data = to_bin(secret_data)
    # size of data to hide
    data_len = len(binary_secret_data)
    for row in image:
        for pixel in row:
            # convert RGB values to binary format
            r, g, b = to_bin(pixel)
            # modify the least significant bit only if there is still data to store
            if data_index < data_len:
                # least significant red pixel bit
                pixel[0] = int(r[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                # least significant green pixel bit
                pixel[1] = int(g[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
```

```
            # convert RGB values to binary format
            r, g, b = to_bin(pixel)
            # modify the least significant bit only if there is still data to store
            if data_index < data_len:
                # least significant red pixel bit
                pixel[0] = int(r[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                # least significant green pixel bit
                pixel[1] = int(g[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                # least significant blue pixel bit
                pixel[2] = int(b[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            # if data is encoded, just break out of the loop
            if data_index >= data_len:
                break
    return image


text = sys.argv[1]
image = sys.argv[2]
```

Live Share tabnine starter See Tabnine Insights Spaces: 4 UTF-8 CRLF Python 3.10.2 64-bit Go Live Stylelint+

Main socket Connections:

Client Side:

```
socket.on("message",(message)=>{
    if(message.type!==messageTypes.LOGIN)
    {
        if(message.author===username)
        {
            message.type=messageTypes.RIGHT;
        }
        else
        {
            message.type=messageTypes.LEFT;
        }
    }
    else{
        messages.push(message);
        displayMessages();
        chatWindow.scrollTop=chatWindow.scrollHeight;
    }

    if(message.toWhom == "To All" || message.toWhom == username){
        messages.push(message);
        displayMessages();
        chatWindow.scrollTop=chatWindow.scrollHeight;
    }
});
```

```
socket.on("base64 file",(fileInfo)=>{
    const image = fileInfo.data;
    if(fileInfo.author == username){
        fileInfo.type = messageTypes.RIGHT;
    }
    else{
        fileInfo.type = messageTypes.LEFT;
    }
    messages.push(fileInfo);
    displayMessages();
    chatWindow.scrollTop=chatWindow.scrollHeight;
});
```

Server Side:

```
// Checking if the users connect
io.on("connection", (socket) => {
  console.log("A user connected");
  // Checking if the users disconnect
  socket.on("disconnect", () => {
    console.log("A user disconnected");
  });

  socket.on("message", (message) => {
    console.log("message", message);
    // Broadcasting this message to all the users that are connected
    io.emit("message", message);
  });      dwivedi-ayush, last month • updated server.js …
```

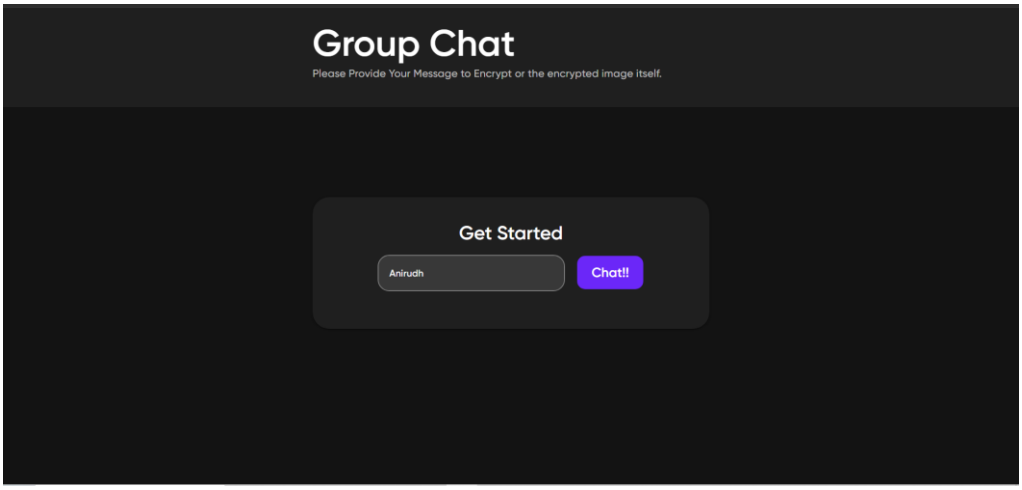
```
const hostname = "127.10.10.2"; // change this to your LAN IP address
const port = 5500;
http.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Setting Cross Origin:

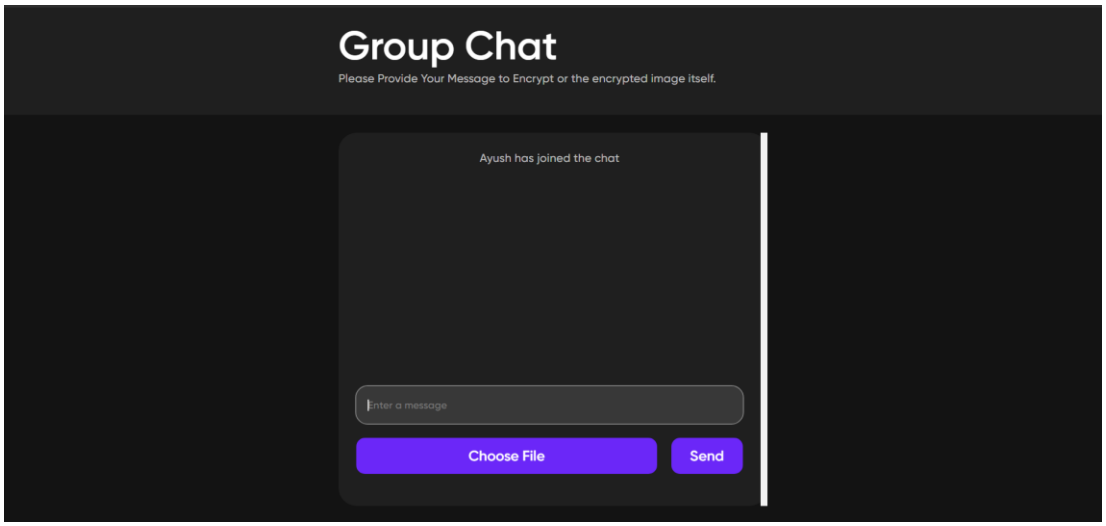
```
// Enable CORS
app.use((req, res, next) => {
  // Set CORS headers
  res.setHeader("Access-Control-Allow-Origin", "*");
  res.setHeader("Access-Control-Allow-Methods", "GET, POST, PUT, DELETE");
  res.setHeader("Access-Control-Allow-Headers", "Content-Type, Authorization");

  // Pass to next layer of middleware
  next();
});      dwivedi-ayush, last month • updated server.js …
```

IMPLEMENTATION SCREENSHOTS:

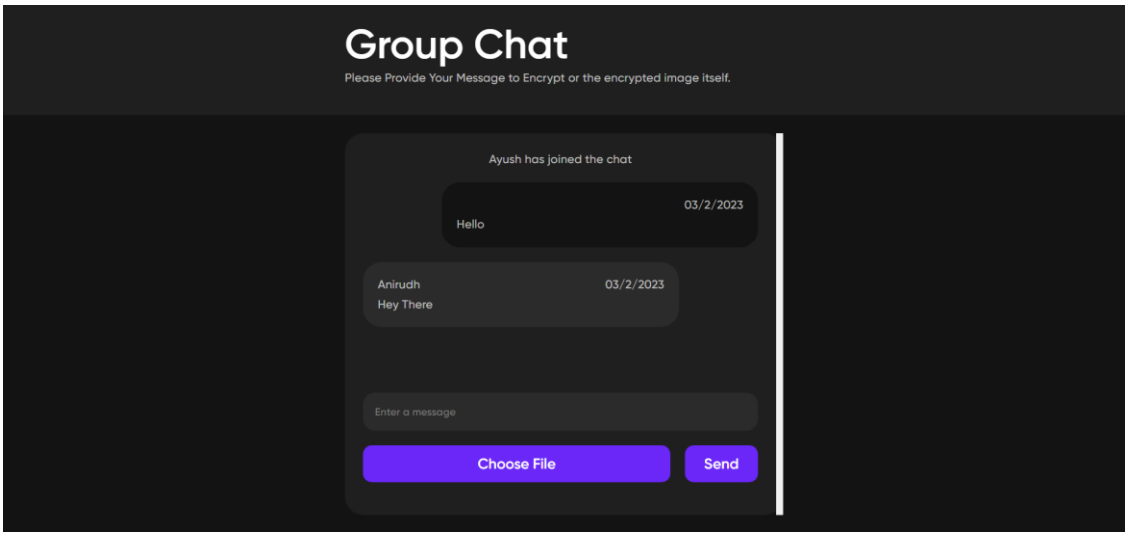
Home Page:

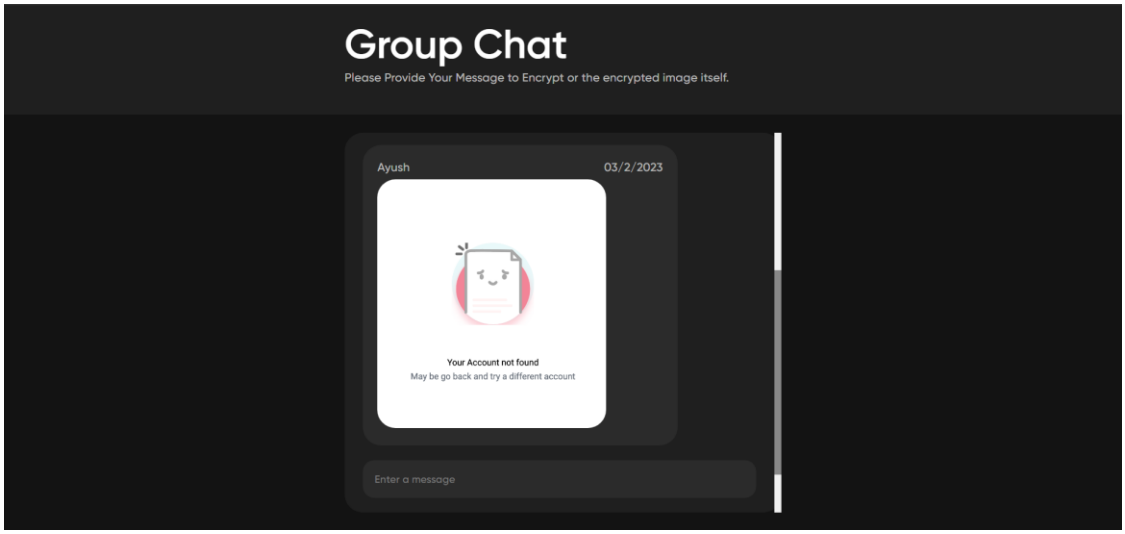


Another user joins:

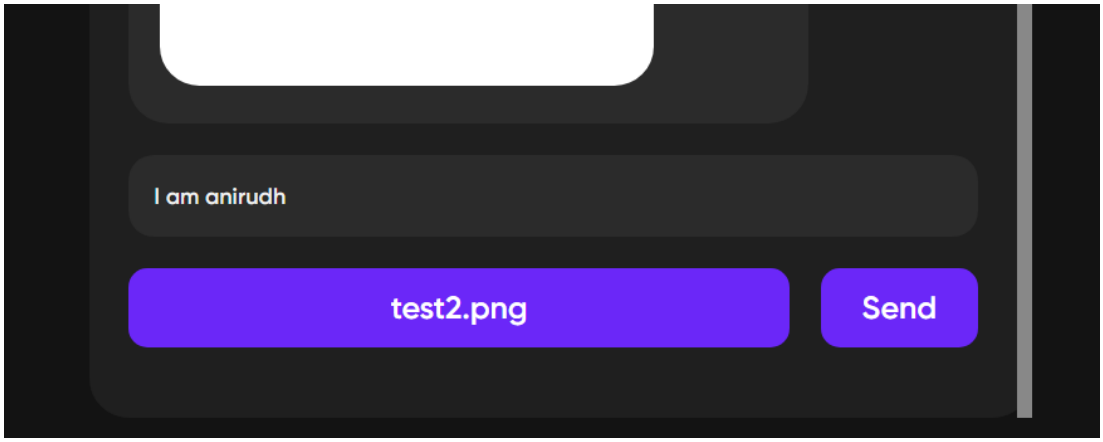


Sending Normal Messages in text and image format:



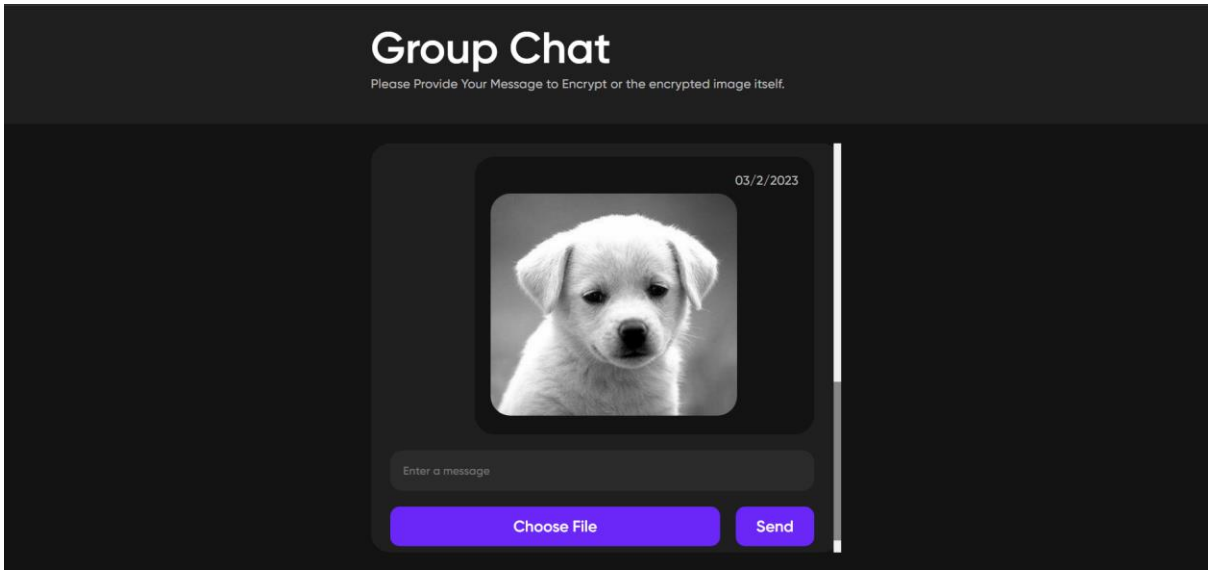


Sending text inscribed in an image(Steganography):

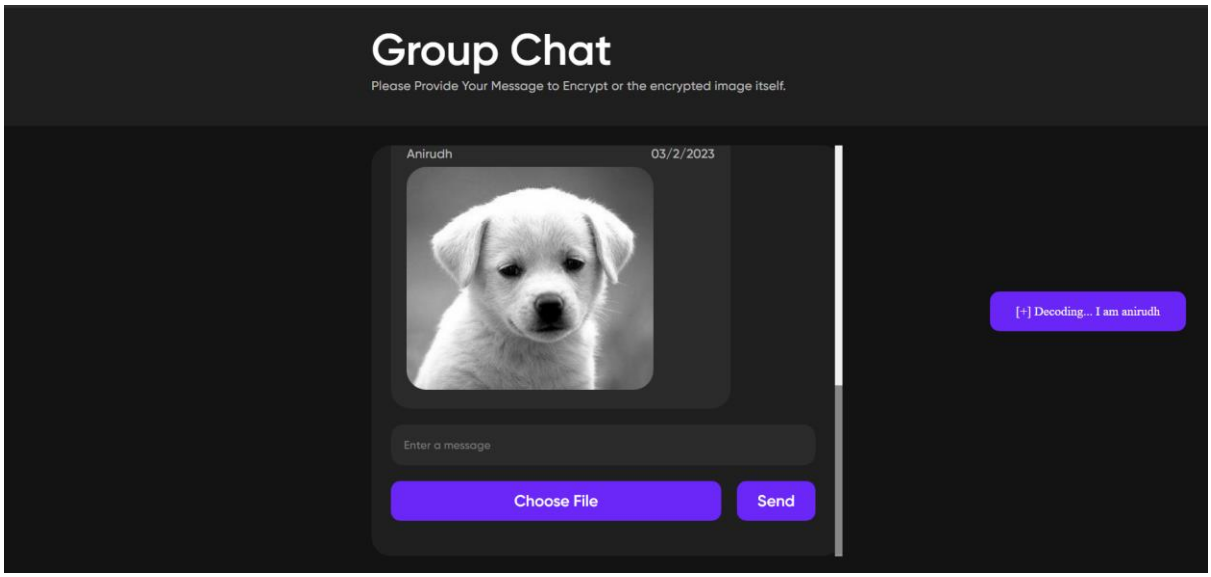


Secret Text: I am Anirudh

Image: test2.png

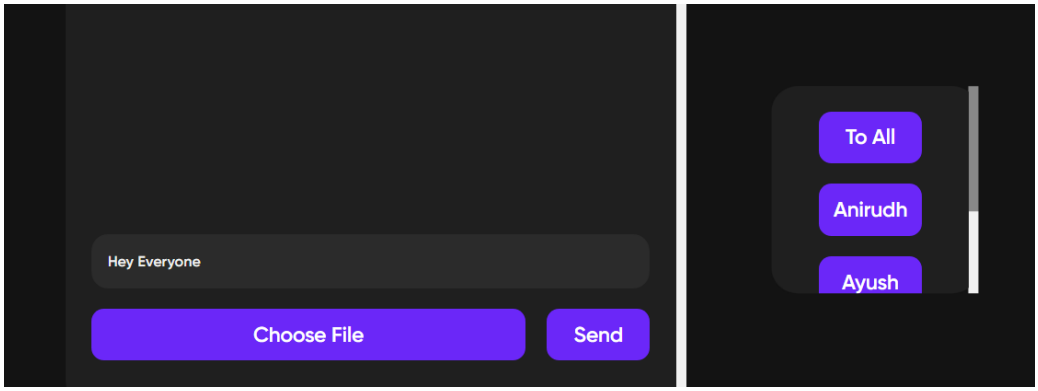


Client can decode the image by clicking on it:

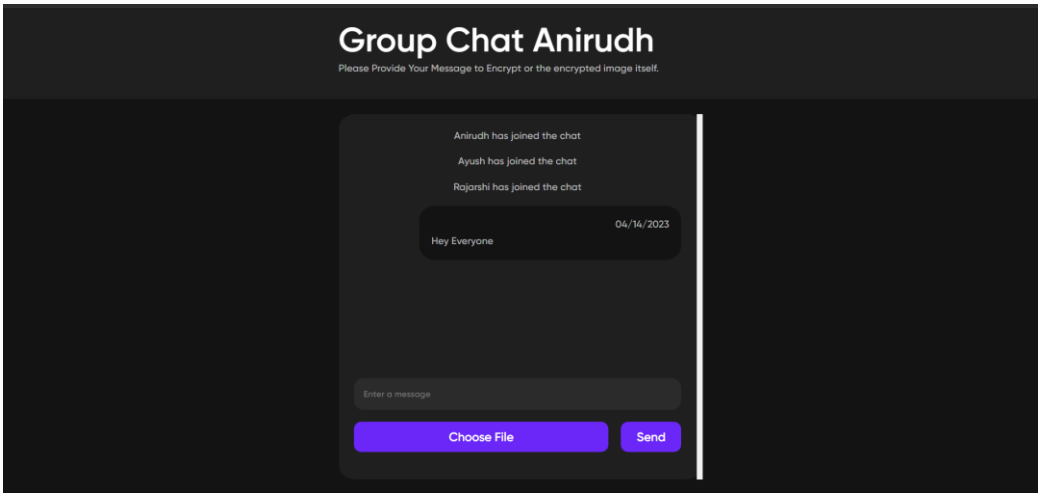


Many other users can choose to join:

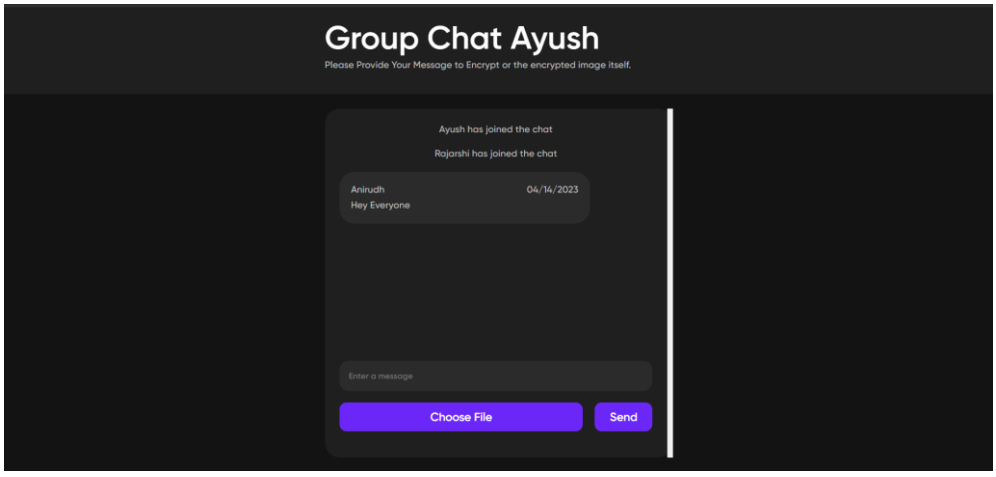
Sending a chat to everyone:



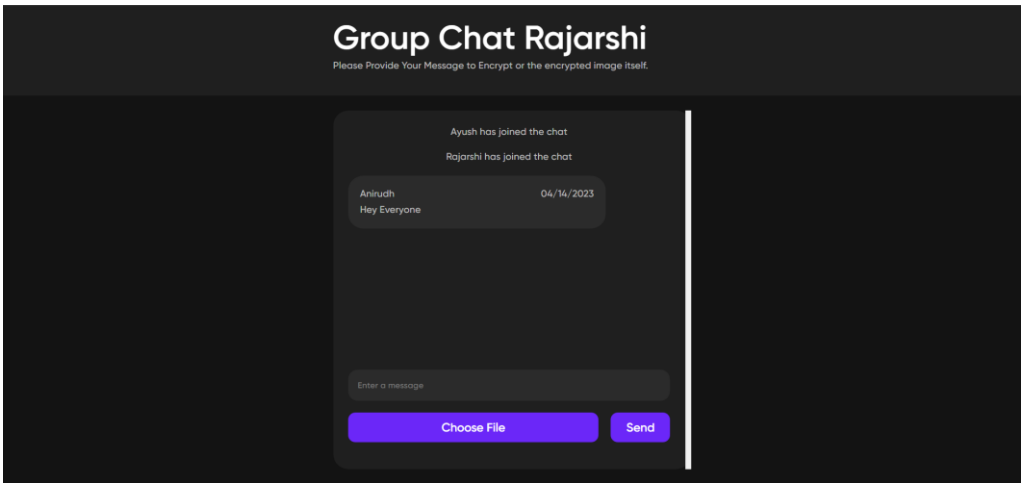
Anirudh Client Side



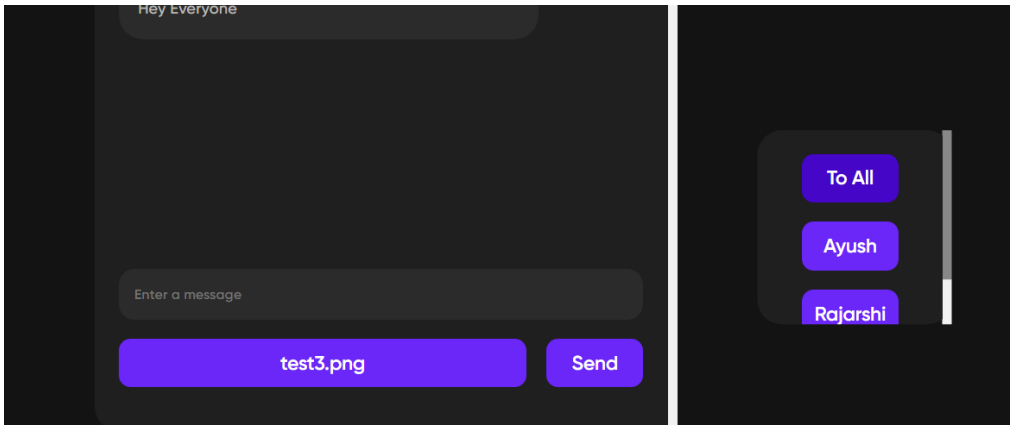
Ayush Client Side



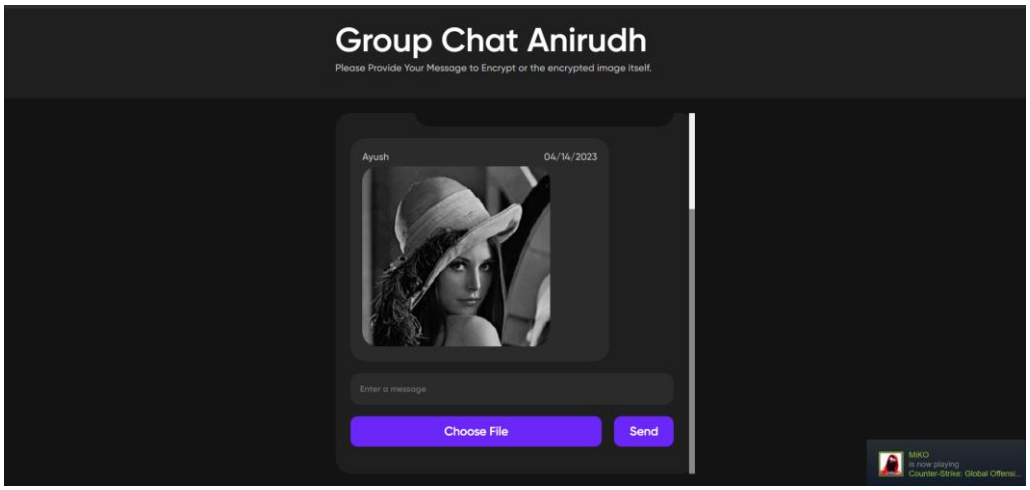
Rajarshi Client Side



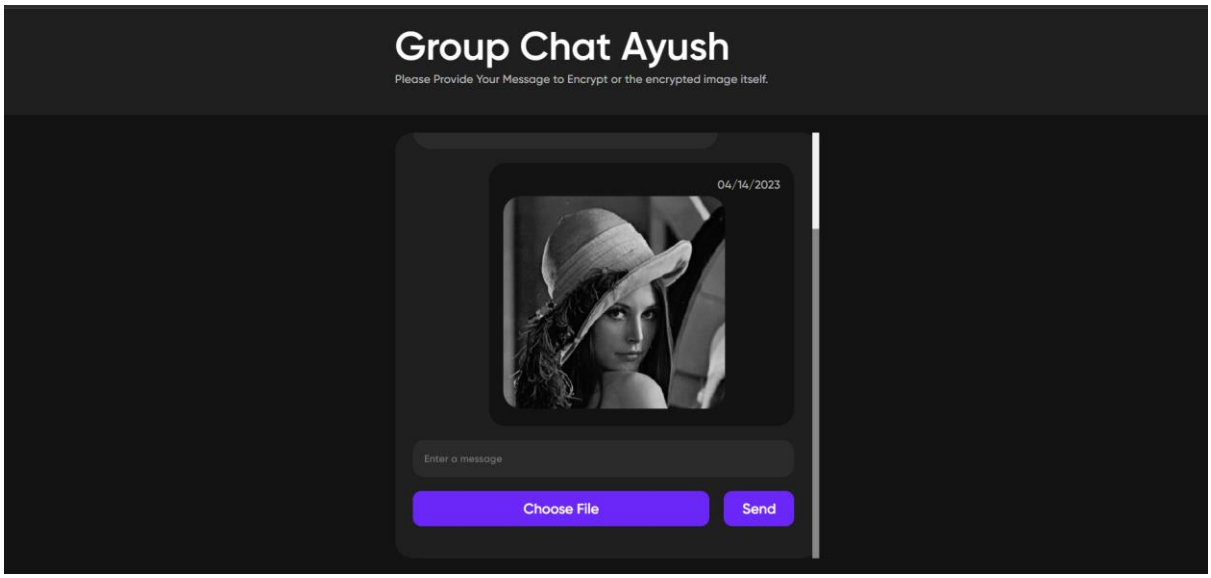
Sending a photo to everyone:



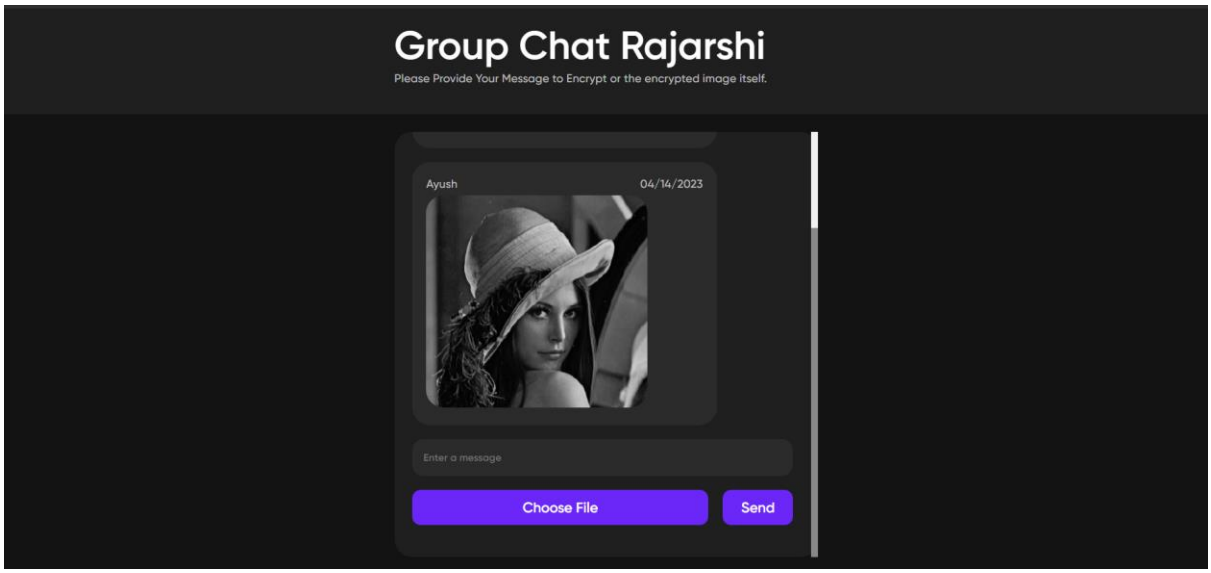
Anirudh Client Side



Ayush Client Side

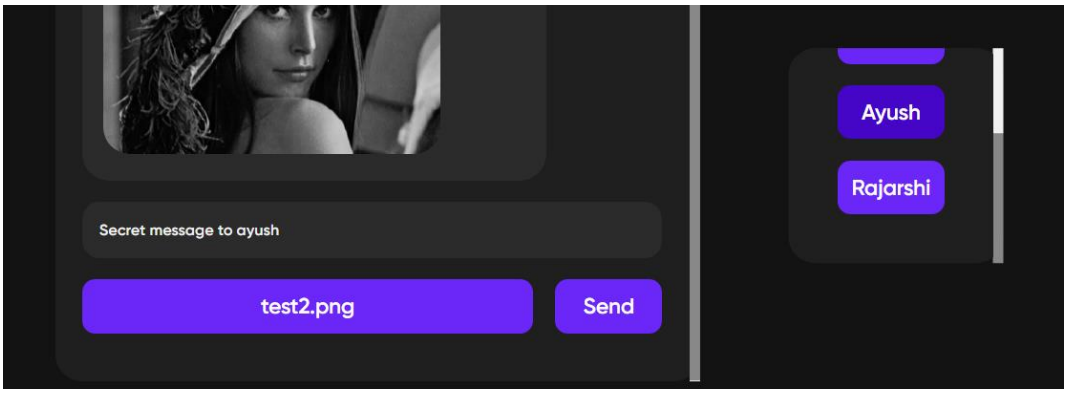


Rajarshi Client Side

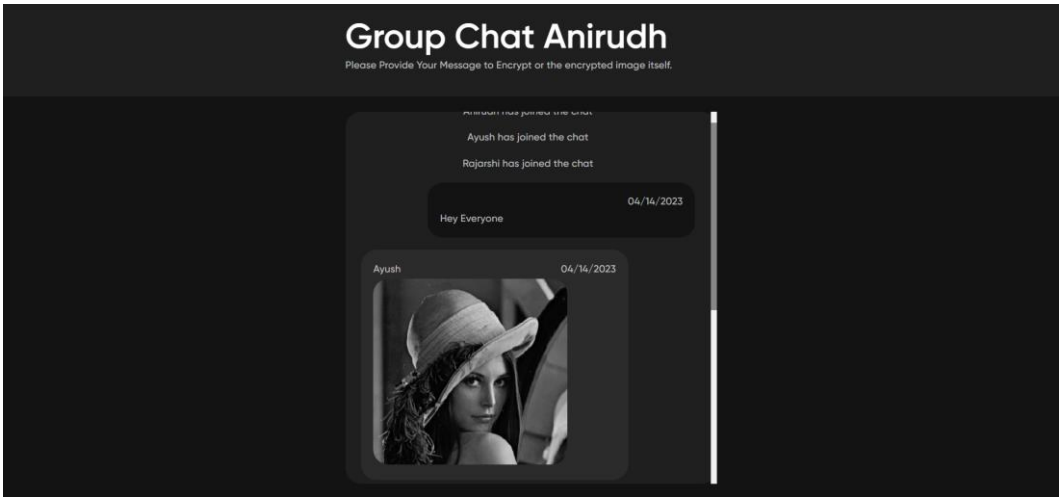


Sending a secret message to a person specified:

Let Anirudh send a secret message only to Ayush and Rajarshi shouldn't be able to see it:

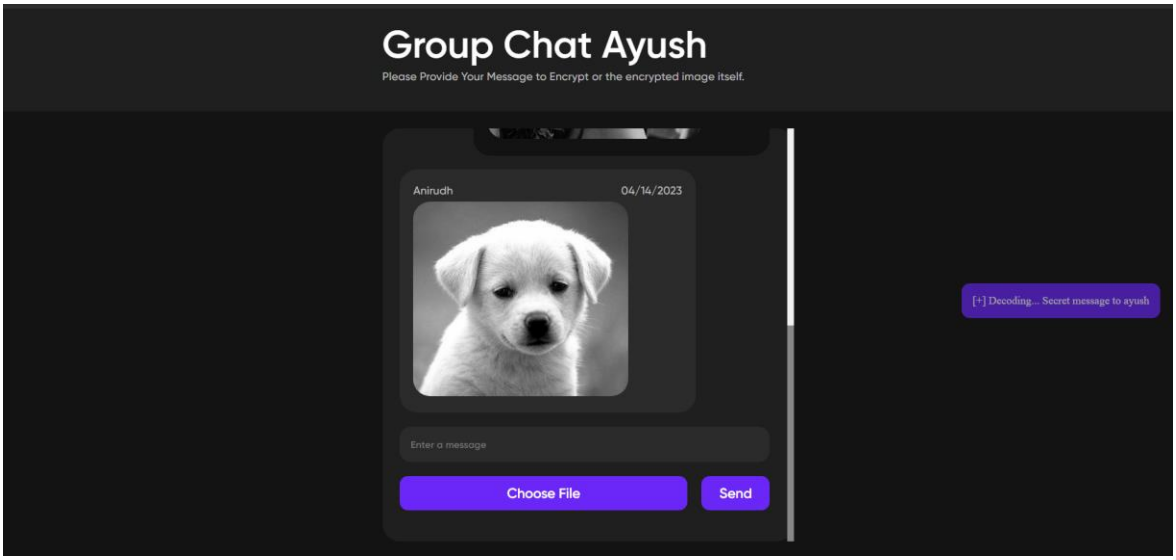


Anirudh Client Side

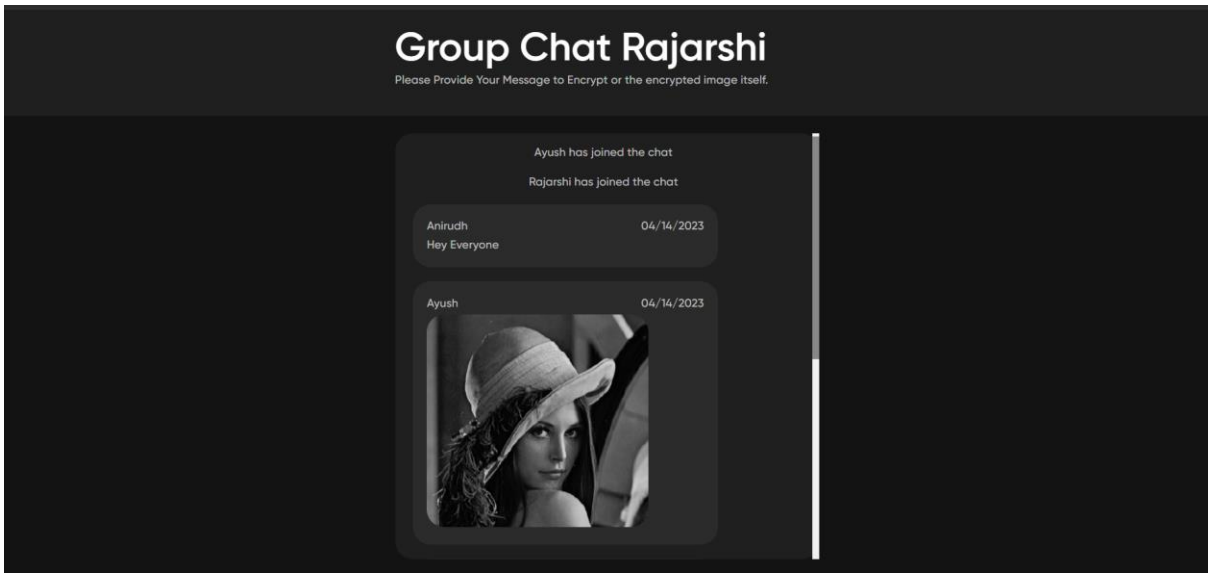


Ayush Client Side

Only he is able to see and decode the message:



Rajarshi Client Side



CONCLUSION:

In conclusion, this project focused on the implementation of steganography techniques to hide data within images and the development of a website that enables users to use these techniques to hide their secret messages in images. The project successfully implemented the modified Least Significant Bit (LSB) substitution technique, which is a simple and easy-to-implement method for hiding data in images.

While LSB steganography has some limitations and can be easily detected under certain conditions, it is still a useful technique for hiding small amounts of data within images. The project also highlighted other more secure and robust steganography techniques, such as Spread Spectrum Steganography and Statistical Steganography, which can be used to hide larger amounts of data and are less susceptible to detection.

Overall, the project achieved its objectives of understanding various steganography and encryption techniques, implementing steganography techniques to hide text and images into other images, creating a website for users to access steganography techniques, extracting data hidden into images with no error or loss of data, and creating a chat application for sending information encrypted using steganography techniques. The users can also send the image to the only person he specified. The project demonstrated the potential of steganography as a valuable tool for secure communication and data protection in the digital world.