**Multi-Image Steganography Using Advanced Encryption Standard**

Om H. Ingale1 , Anushka B. Kadam2 , Saily G. Patil3 , Sahil V. Shilimkar4 , Prof.

Tejal Panmand5

\*1, 2, 3, 4, 5 Computer Engineering, Zeal Polytechnic, Pune, Maharashtra, India.

***ABSTRACT: Steganography is the practice of hiding information within other non-secret data, making it difficult for unintended recipients to detect the presence of the hidden information. The primary goal of steganography is to ensure the confidentiality of the message by concealing its existence rather than encrypting it. This technique can be applied to various forms of audio, video and images.***

***KEYWORDS: Steganography, Hiding Information, Confidentiality, Concealment, Hidden Information, Detection, Encryption, Audio, Video and Images.***

**Introduction to Audio, Video, and**

**Image Steganography:**

Steganography is the practice of concealing information within another medium to ensure secure and undetectable communication. Unlike cryptography, which focuses on scrambling the content of a message to make it unreadable without a decryption key, steganography hides the existence of the message itself. This makes it a powerful tool for protecting sensitive information, especially in a digital era where data privacy and secure communication are critical.

**Image Steganography:**

Image steganography involves embedding secret data into digital images. This is typically achieved by altering the least significant bits (LSBs) of the pixel values so that changes remain visually imperceptible. Due to the high redundancy and large data capacity in images, they are one of the most commonly used carriers for steganography. The visual fidelity of the image remains almost unchanged, making it difficult for an observer to detect the presence of hidden information.

**Audio Steganography:**

In audio steganography, information is hidden within audio files. This can be done using various methods such as echo hiding, phase coding, or LSB manipulation. The human auditory system is less sensitive to slight variations in sound, which allows for embedding data without noticeably degrading audio quality. Audio steganography is often used in secure communications, watermarking, and copyright protection.

**Video Steganography:**

Video steganography involves embedding secret information in digital video files. Since video is a sequence of images (frames) with accompanying audio, it offers a large hiding capacity and greater robustness. Data can be embedded either in individual frames or across the video stream, using techniques similar to image and audio steganography. The dynamic nature of videos adds an extra layer of complexity for unauthorized detection and extraction.

Together, these forms of steganography enable secure, covert communication across various digital media, making them valuable in fields such as cyber security, digital rights management, and military intelligence.

**Literature Survey**:

1. **ImageSteganography:**  
   Chan and Cheng (2004) introduced LSB substitution for simplicity and high capacity, though with lower robustness. Baisa et al. (2013) improved imperceptibility using Pixel Value Differencing (PVD). Transform domain methods like DCT and DWT (Zhang et al., 2010) offered better resistance against compression and attacks.
2. **AudioSteganography:**  
   Johnson and Katzenbeisser (2000) reviewed LSB techniques, which are simple but less secure. Bender et al. (1996) proposed Echo Hiding to increase robustness. Phase Coding (Swanson et al., 1998) enhanced imperceptibility by modifying the audio's phase spectrum.
3. **VideoSteganography:**  
   Petitcolas et al. (1999) highlighted early challenges in video steganography. Noda et al. (2005) used DCT-based methods on MPEG I-frames. Kumar et al. (2017) advanced motion vector and transform domain techniques for greater security and capacity.

**Problem Statement:**

As digital communication becomes increasingly prevalent, ensuring the security and confidentiality of transmitted information is a growing concern. While traditional encryption techniques protect the content of the message, they do not hide the fact that communication is taking place. This can attract attention from malicious actors, making the transmission vulnerable to interception. Steganography offers a potential solution by concealing the existence of the message, but current techniques face challenges in balancing security, performance, and usability, particularly in resource-constrained environments like smartphones.

The challenge lies in designing a steganographic method that is secure against detection (steganalysis), performs well on mobile devices with limited resources, and remains easy to implement. While more advanced methods like DCT provide greater security, they often come at the cost of increased computational requirements, which can negatively affect performance on Android-based devices.

**1. Data Hiding**

Allows secret information (text, files, etc.) to be hidden inside audio, video, or image files without noticeable changes to the original media.

**2. Imperceptibility**

The changes made to the audio, video, or image are so minor that the human senses (eyes or ears) cannot detect any difference.

**3. High Capacity**

These methods can hide a good amount of data, especially in video files, which have large storage potential due to many frames and audio channels.

**4. Robustness**

Steganography can be made strong enough to resist compression, cropping, or noise—especially using advanced techniques like DCT or DWT.

**5. Security**

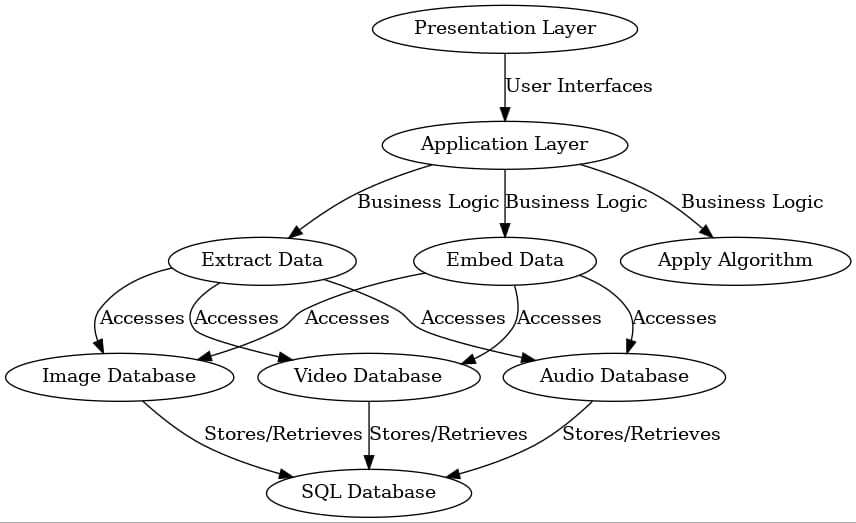
Only someone with the correct key or method can extract the hidden message, keeping the information safe from unauthorized access.

**Security and Reliability in Steganography:**

Steganography ensures secure and reliable communication by hiding data within media files in a way that prevents detection or tampering.

1. **Confidentiality:**  
   Steganography ensures that only the intended recipient with the correct key or decoding method can access the hidden message, preventing unauthorized access or interception.
2. **Undetectability:**  
   The hidden data does not noticeably alter the appearance or sound of the original media, making it extremely difficult for attackers or observers to detect that any information is embedded.
3. **Resistance to Attacks:**  
   Reliable steganographic methods are designed to withstand common operations like image compression, audio filtering, or video trimming, ensuring that the hidden data stays intact even after such modifications.
4. **DataIntegrity:**  
   Steganography maintains the accuracy and completeness of the hidden message, ensuring that it is not corrupted or altered during transmission or storage.
5. **System Architecture**

4.1 System Architecture The layered architecture of Multi-Media Steganography provides a robust framework for developing a secure, maintainable, and scalable steganography system. It promotes good software engineering practices and contributes to the overall success of the project.



1. **Presentation Layer**

The Presentation Layer is responsible for translating, encrypting, and formatting data so that it can be properly hidden and later extracted in steganographic processes. In audio, video, and image steganography, this layer plays a key role in:

1. **Data Encoding & Compression** – Converts secret data into a suitable format for embedding (e.g., binary format or compressed files).
2. **Media Formatting** – Prepares the carrier file (image, audio, or video) by applying transformations (like DCT or LSB techniques) to hide the data securely.
3. **Encryption (Optional)** – Enhances security by encrypting the secret data before embedding, ensuring confidentiality even if detected.
4. **Compatibility Handling** – Ensures data and media formats are compatible for embedding and extraction across different systems.

**User\_Interface\_(Screens):**

The user interface allows users to easily **select media files, enter secret messages, choose encryption keys,** and **embed or extract data** with simple buttons and clear instructions. A good UI ensures a smooth, user-friendly experience for secure steganographic operations.

**Application\_Layer:**

The application layer handles userinteraction and controls the embedding and extraction of hidden data. It processes inputs like media files, messages, and keys, and shows the final results, acting as a bridge between the user and the steganography system.

**Business Logic:**

The business logic controls how **data is hidden or extracted** using specific algorithms (like LSB or DCT), manages **encryption**, and ensures **valid input** and proper flow between the user interface and processing layers.

### Embedding Message Management:

### Audio Steganography

* Hides message inside audio files.
* Uses methods like:
  + **LSB**: Changes last bits of audio data.
  + **Echo hiding**: Adds small echoes to hide data.
* Must keep sound clear and undetectable.

**Video Steganography**

* Hides message inside video files.
* Uses methods like:
  + **Frame-based LSB**: Hides data in video frames.
  + **Motion vectors**: Alters motion data to store info.
* Should not affect video quality or playback.

**Image Steganography**

* Hides message inside image files.
* Uses methods like:
  + **LSB**: Changes last bits of pixel colors.
  + **DCT/DWT**: Hides data in image frequency parts.
* Image should look normal without visible changes.

**DATABASE:**

**Audio Steganography Database**

* **Stores**:
  + Original audio files
  + Stego-audio files (with hidden message)
  + Message data and keys used
* **Purpose**:
  + For comparing, extracting, or verifying hidden messages
  + For tracking different embedding techniques used

**Video Steganography Database**

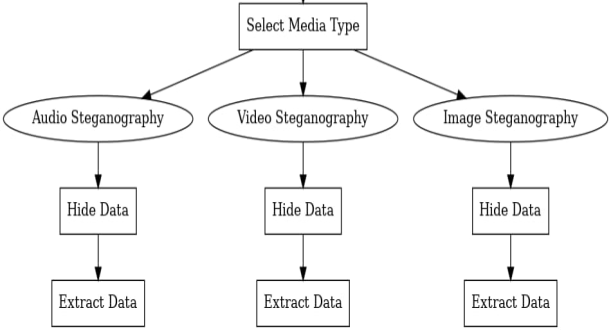
* **Stores**:
  + Original videos
  + Stego-videos with hidden messages
  + Frame-wise embedding details
  + Message, key, and timestamps
* **Purpose**:
  + To manage large video data and embedded content
  + For frame extraction and message recovery

**Image Steganography Database**

* **Stores**:
  + Hidden messages
  + Image format, size, and encryption key info
* **Purpose**:
  + To analyze image changes
  + For message retrieval and security check

**Flowcharts:**

1. **Steganography Flowchart**



**Media Options:**

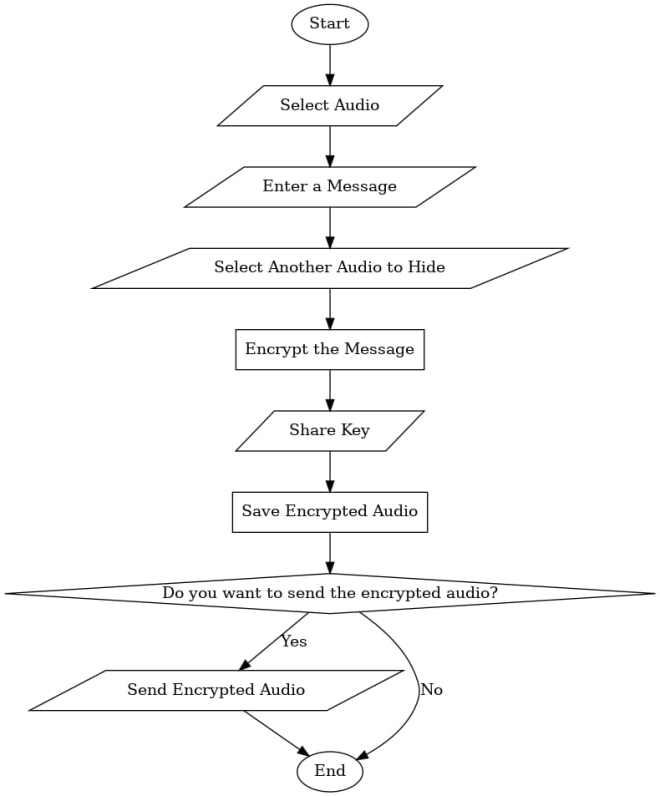
* Audio Steganography
* Video Steganography
* Image Steganography

**Processes for each media type:**

Hide Data: Embed secret data into the chosen media.

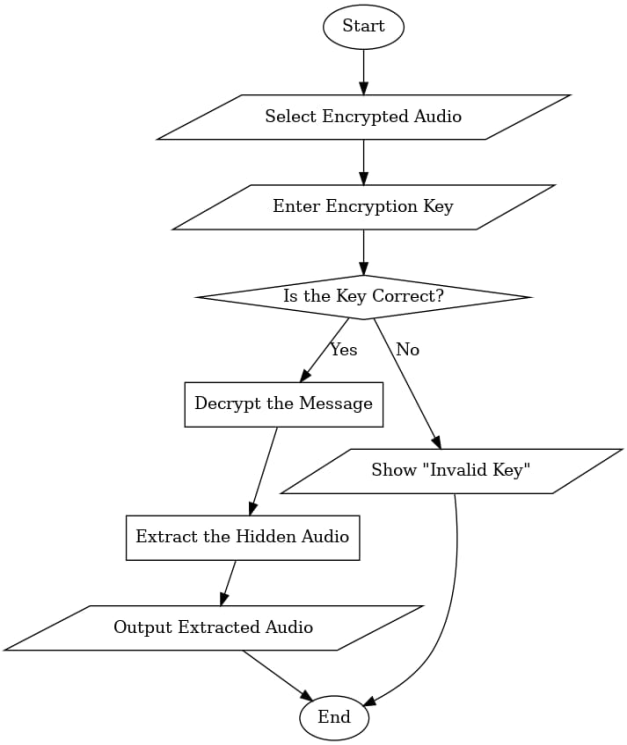
Extract Data: Retrieve hidden data from the media.

**2.Audio Encrypt Flowchart**

****

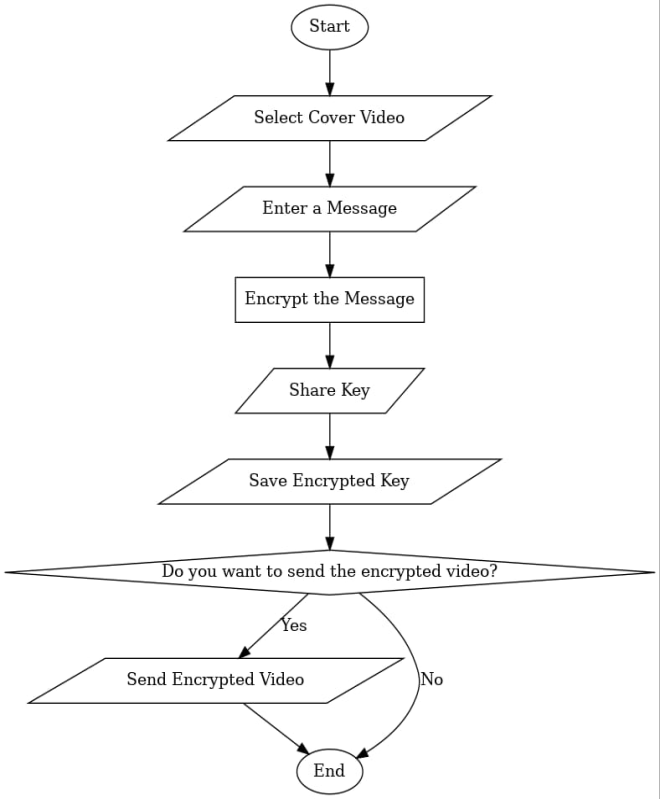
The flowchart illustrates the process of encrypting a message within an audio file using audio steganography. It begins with selecting an audio file, after which the user inputs a message to hide. Another audio file is then chosen to embed the message. The message is encrypted, and an encryption key is generated and shared. The encrypted audio is saved, and the user is prompted to decide whether to send the encrypted audio. If yes, the file is sent; otherwise, the process ends. This flow ensures secure communication by embedding encrypted messages within audio files.

**3.Audio Decrypt Flowchart**

****

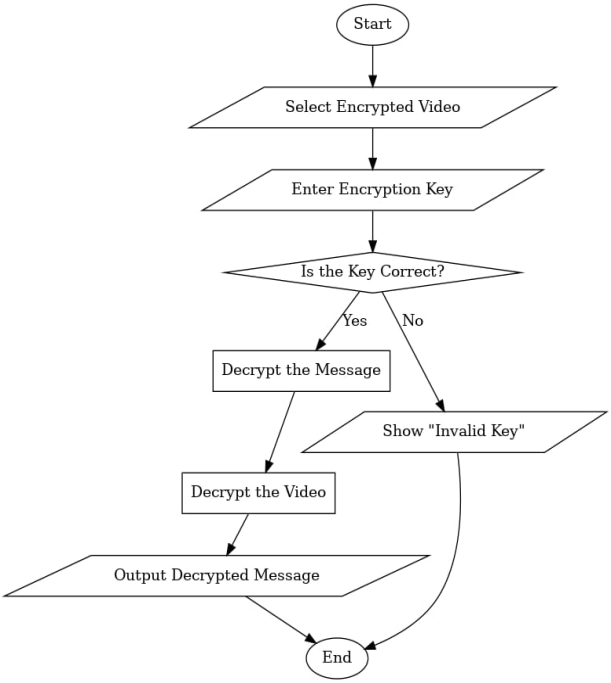
This flowchart represents the process of decrypting audio that contains a hidden message using audio steganography. The process begins with selecting the encrypted audio file, followed by entering the encryption key. The system then verifies the correctness of the key. If the key is correct, it proceeds to decrypt the hidden message and extract the hidden audio from it. The extracted audio is then outputted to the user, concluding the process. However, if the key is incorrect, an "Invalid Key" message is displayed, and the process ends without revealing the hidden content.

**4.Video Encrypt Flowchart**

****

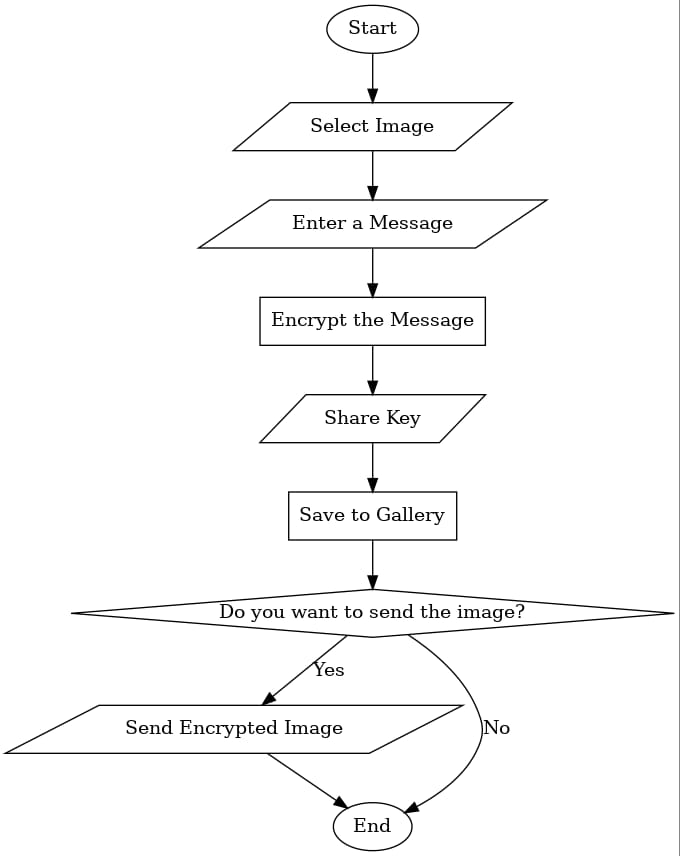
The image displays a flowchart titled "Video Encrypt Flowchart," which outlines the step-by-step process of embedding a hidden message into a video using video steganography. The process begins with the user selecting a cover video, which serves as the base medium. The user then enters a message that they wish to hide within the video. This message is encrypted to enhance security. After encryption, a key is generated and shared, followed by saving the encrypted key securely. At this point, the system asks the user whether they wish to send the encrypted video. If the user selects "Yes," the encrypted video is sent; if the user selects "No," the process ends without sending.

**5.Video Decrypt Flowchart**

****

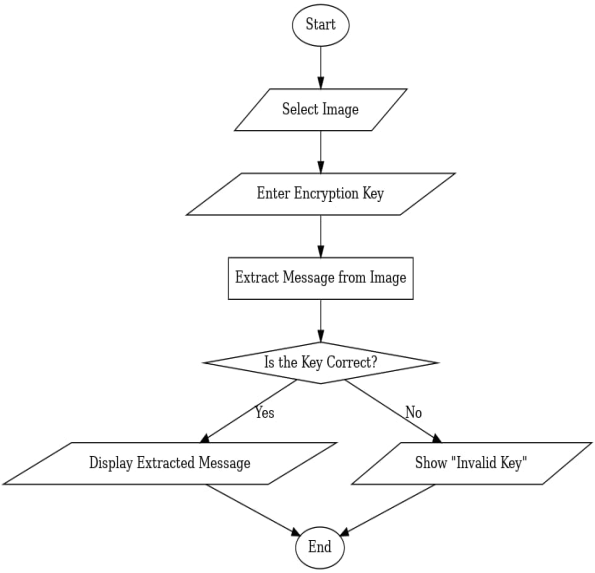
The image illustrates a flowchart titled "Video Decrypt Flowchart," which details the process of retrieving a hidden message from an encrypted video. The procedure starts by selecting the encrypted video file, followed by entering the corresponding encryption key. The system then checks if the entered key is correct. If the key is valid, the process continues with decrypting the hidden message and then decrypting the video itself to extract and display the concealed message. If the key is incorrect, an "Invalid Key" message is shown, and the process terminates. This flowchart ensures a secure and structured approach to video decryption in steganography applications.

**5.Image Encrypt Flowchart**

****

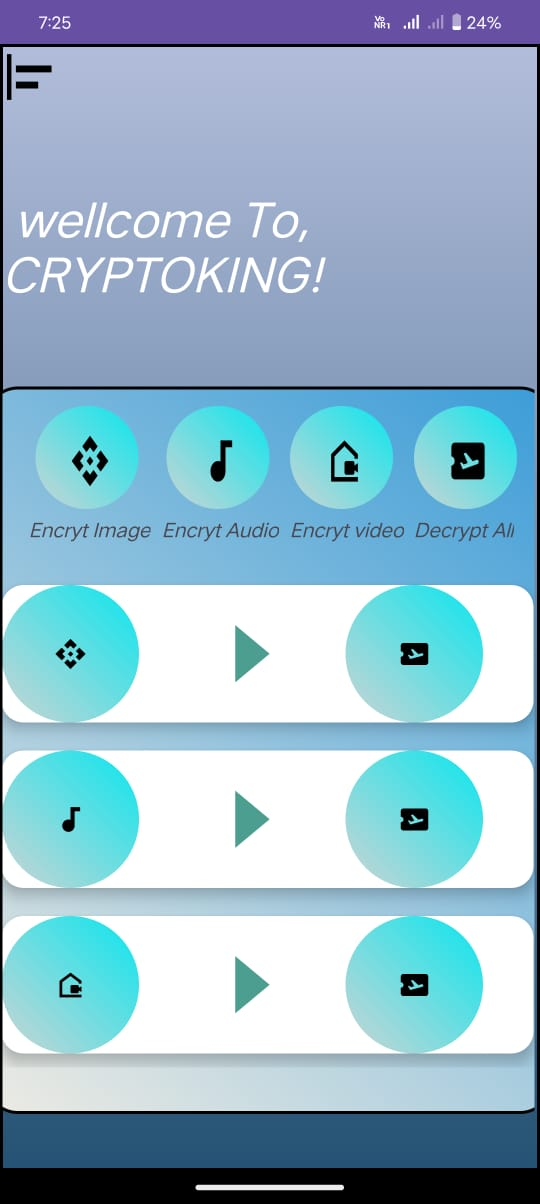
The image presents a flowchart titled "Image Encrypt Flowchart," which outlines the steps involved in encrypting a message into an image. The process begins with selecting an image, followed by entering the message that needs to be hidden. Once the message is provided, it is encrypted, and the encryption key is then shared. After encryption, the resulting image is saved to the gallery. The user is then prompted with an option to send the encrypted image. If the user chooses to send it, the image is transmitted; otherwise, the process concludes. This flowchart effectively captures the essential stages of image-based message encryption and optional transmission.

**6.Image Decrypt Flowchart**

****

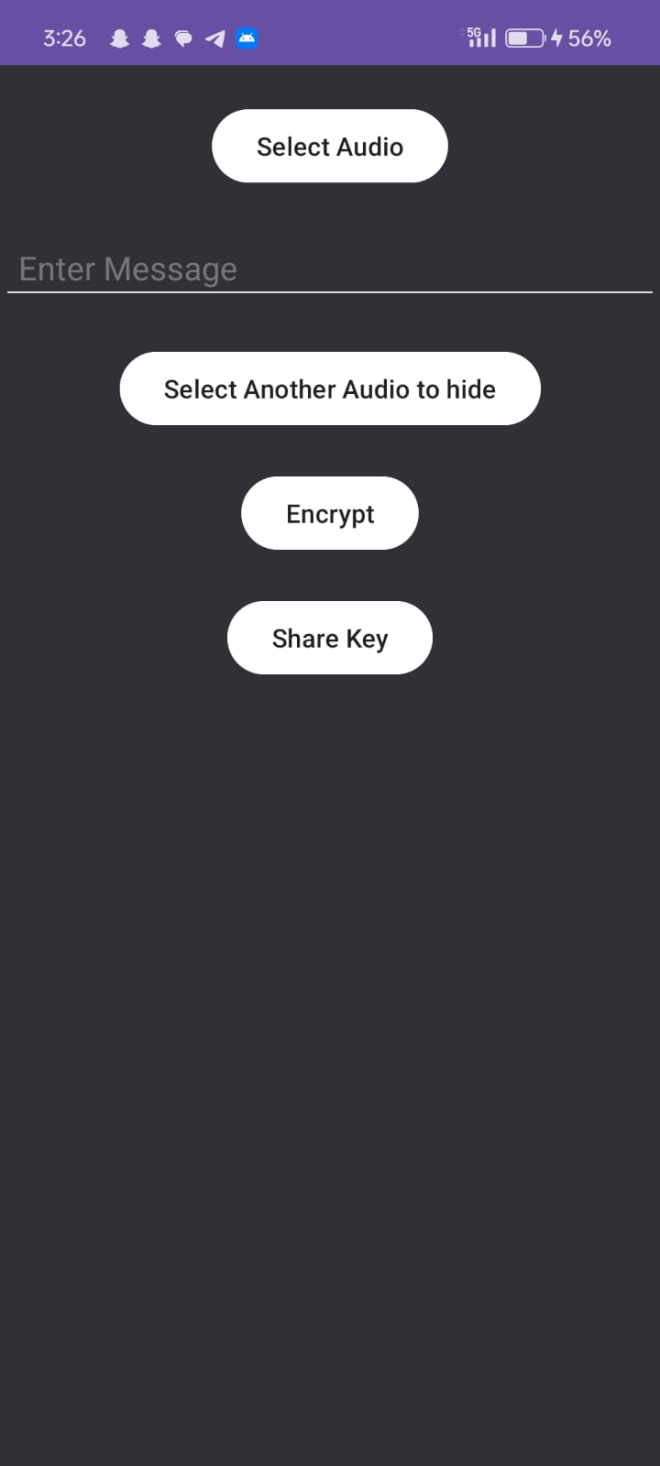
The image displays a flowchart illustrating the process of decrypting a message embedded within an image. The process begins with selecting an image and then entering an encryption key. The system then attempts to extract the message from the image using the provided key. A decision is made to check if the entered key is correct. If the key is correct ("Yes" branch), the extracted message is displayed. If the key is incorrect ("No" branch), an "Invalid Key" message is shown. The decryption process then ends in either case.

**RESULT**

****

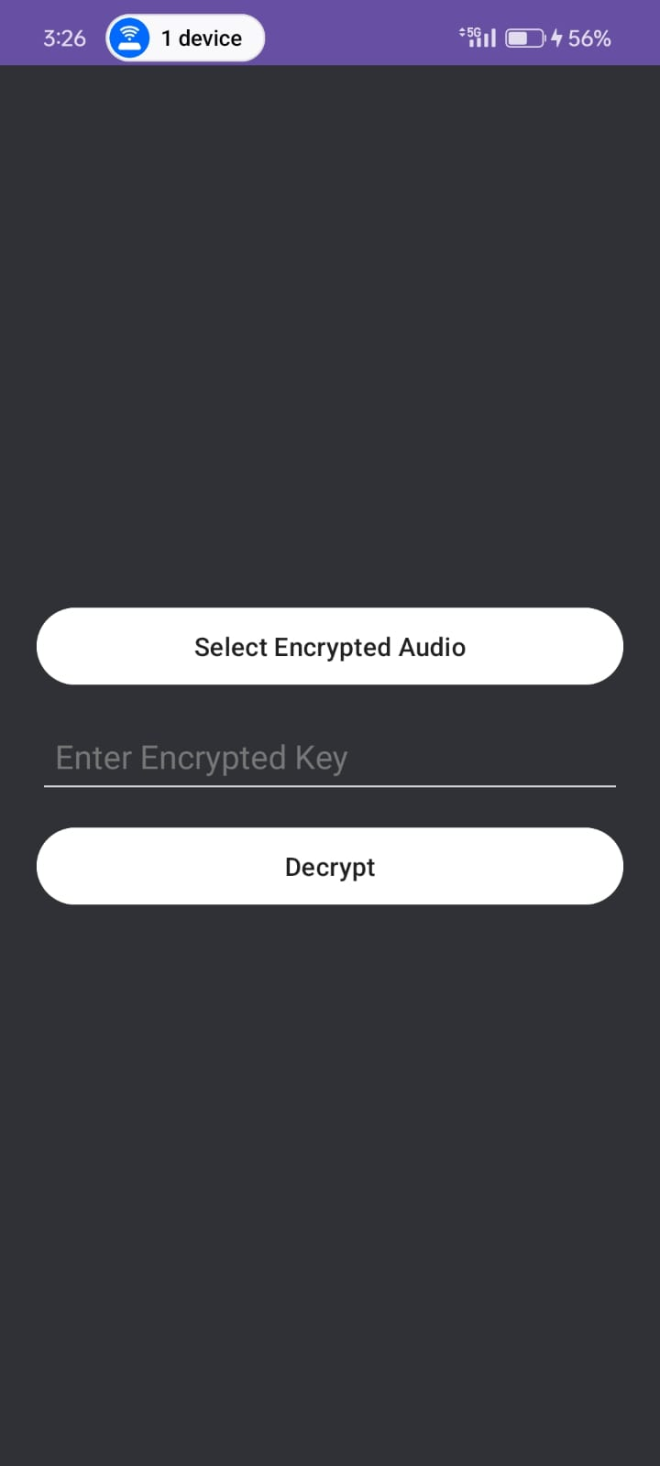
Steganography Homepage

1. This is the main interface of the "CRYPTOKING" app, designed for encrypting and decrypting images, audio, and video files. It features a welcome message, followed by four circular buttons for "Encrypt Image," "Encrypt Audio," "Encrypt Video," and "Decrypt All." Below that, each media type has its own row with icons representing the file type, a play button to initiate encryption, and a download icon likely for decryption. The design uses a clean, modern blue gradient with intuitive, user-friendly navigation.

****

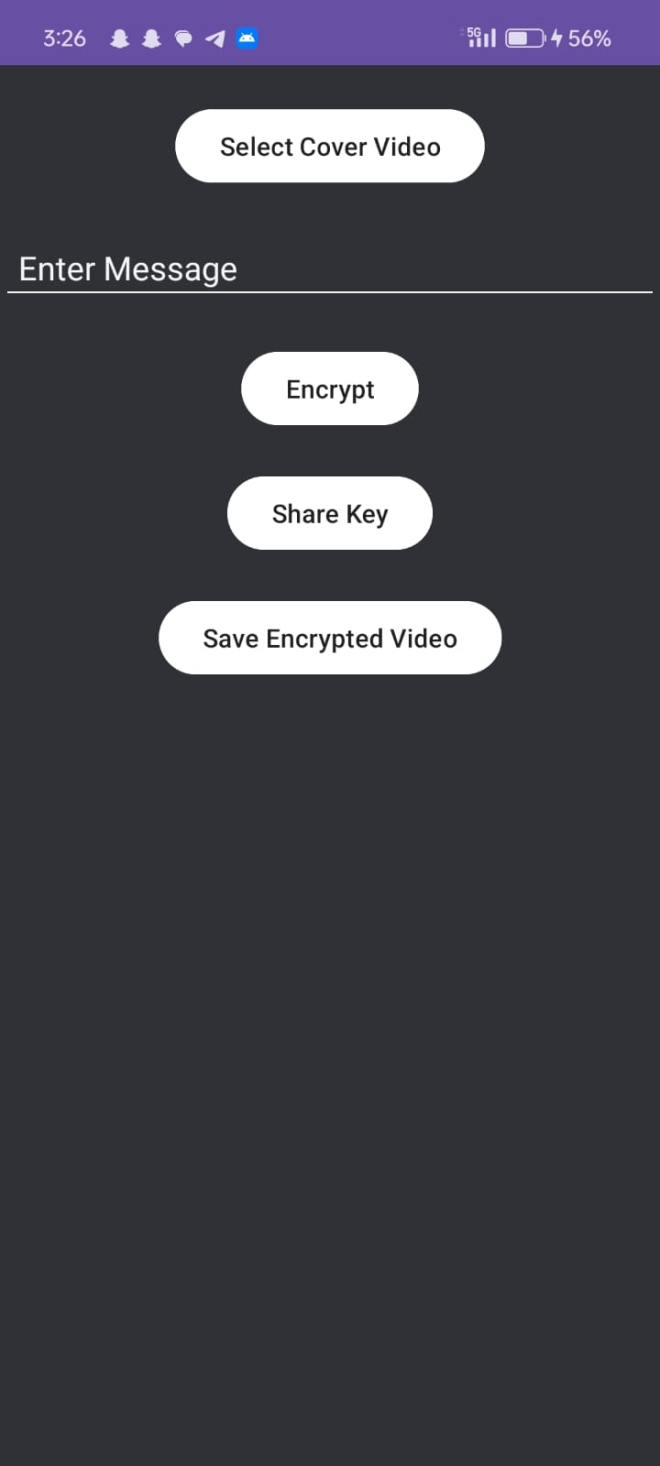
Audio Encrypt Interface

1. This interface is part of the audio encryption module in the CRYPTOKING app. It allows the user to select an audio file, enter a secret message, and then hide that message inside another audio file. The interface includes a "Select Audio" button to choose the main audio file, a text input field labeled "Enter Message" for the user to type the message to be hidden, and another button labeled "Select Another Audio to hide" which likely serves as the carrier audio file for steganography. After both files are selected and the message is entered, the "Encrypt" button performs the embedding process. Finally, there's a "Share Key" button that lets the user send the encryption key required for decryption, ensuring secure communication. The design is simple and focused, with a dark theme and clearly labeled white buttons for ease of use.



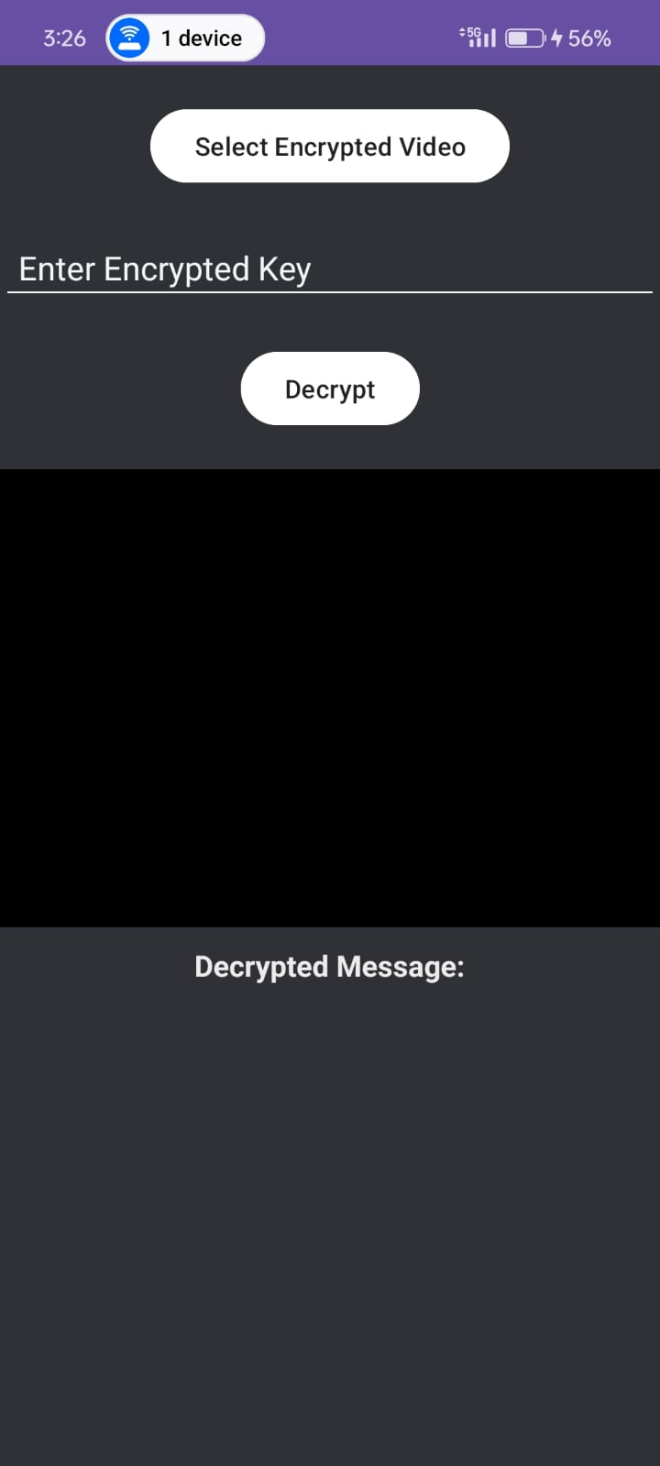
Audio Decrypt Interface

3. This is the decryption screen of the CRYPTOKING app. On this screen, the user first selects the encrypted audio file that contains the hidden message. After selecting the file, the user is required to enter the encrypted key that was originally used during the encryption process. Once both the encrypted audio and the key are provided, the user taps the "Decrypt" button to reveal and retrieve the hidden message from the audio file. This screen essentially allows the user to extract the secret message safely using the correct key.

****

Video Encrypt Interface

4. This is the encryption screen of the CRYPTOKING app for video steganography. The user begins by selecting a cover video where the secret message will be hidden. After selecting the video, the user enters the message in the provided text field. By tapping the "Encrypt" button, the app securely hides the message within the video using an encryption algorithm. Once the encryption is complete, the "Share Key" option allows the user to send the generated encryption key to the intended recipient. Finally, the user can tap "Save Encrypted Video" to store the video containing the hidden message for later use or sharing.

****

Video Decrypt Interface

5. This is the decryption screen of the CRYPTOKING app, which is used to extract hidden messages from encrypted videos. To begin the decryption process, the user must first select the encrypted video file that contains the hidden message. After selecting the video, the user is required to input the encrypted key that was previously shared during the encryption stage. By clicking on the "Decrypt" button, the app uses the provided key to decode and reveal the original hidden message. Once decrypted, the message is displayed on the screen under the label "Decrypted Message," making it visible to the authorized user.

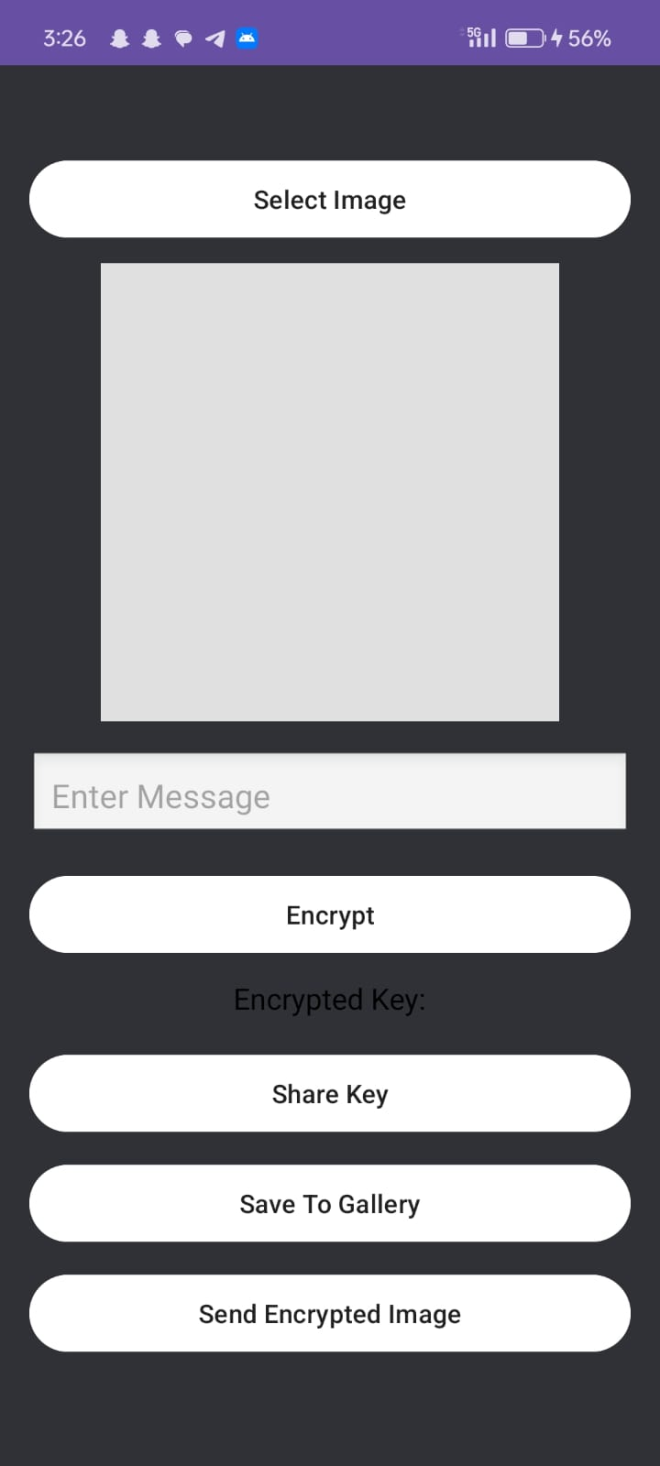
****

Image Encrypt Interface

6. This interface enables image encryption by allowing users to select an image and optionally embed a message before initiating the encryption process with the "Encrypt" button, which then generates a crucial "Encrypted Key" for decryption that can be shared via the "Share Key" option; the resulting encrypted image can be saved to the device's gallery using "Save To Gallery" or directly sent with the "Send Encrypted Image" button, all within a context that includes standard navigation and communication-related icons like "Reply" and an emoji button.

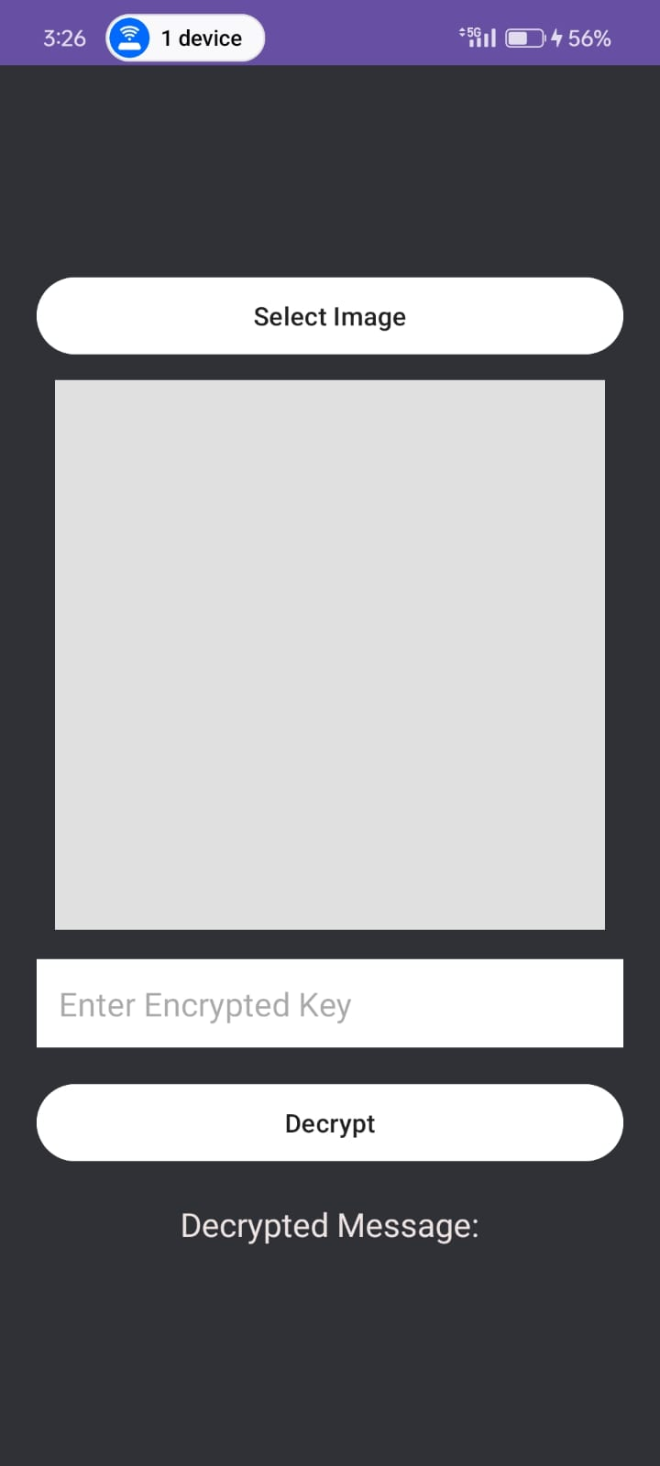
****

Image Decrypt Interface

7. This interface is designed for decrypting an encrypted image. It presents a straightforward process: the user is prompted to "Select Image", which allows them to upload the encrypted image file. Once the image is selected, the interface provides an input field labeled "Enter Encrypted Key". Here, the user must input the correct decryption key that was used to encrypt the image. After entering the key, the user needs to tap the "Decrypt" button to initiate the decryption process. If the key is correct, the decrypted image will likely be displayed in the area above the "Decrypted Message:", and any accompanying decrypted message might appear below that label. Similar to the encryption interface, this screen also includes standard navigation icons at the top and bottom for ease of use within the application.

**Conclusions:**

Audio, video, and image steganography are powerful techniques used to hide secret messages within digital media. Each method has its own advantages and challenges in terms of data capacity, security, and imperceptibility. Audio steganography ensures hidden communication through sound, image steganography uses visual files without noticeable changes, and video steganography allows embedding in moving visuals for large data. Together, these methods enhance secure communication, protect data privacy, and support digital forensics and copyright protection. Proper implementation and database management are essential for effective and reliable steganographic systems.

**References:**

<https://www.ijraset.com/research-paper/android-application-for-image-steganography-using-android-studio>

<https://www.ijraset.com/research-paper/android-application-for-image-steganography-using-android-studio>

<http://ijirrd.com/wp-content/uploads/050110.pdf>

<https://ijcrt.org/papers/IJCRT1812634.pdf>

<https://web.stanford.edu/class/ee368/Project_Autumn_1617/Reports/report_piens_staffa.pdf>