VIDEO ENCRYPTION AND DECRYPTION USING SYMMETRIC KEY

IT5703 - CRYPTOGRAPHY AND SECURITY

A PROJECT REPORT

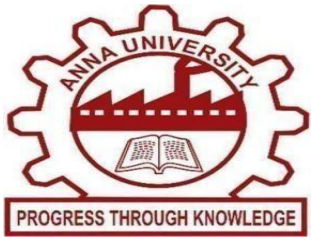
Submitted by

Shrish R

(2020506088)

Deepak Athipan A M B

(2020506021)



DEPARTMENT OF INFORMATION TECHNOLOGY

MADRAS INSTITUTE OF TECHNOLOGY CAMPUS

ANNA UNIVERSITY : CHENNAI - 600 044

AUGUST 2023 – NOVEMBER 2023

1. INTRODUCTION

The video encryption and decryption are processes that consist of safeguarding video data in order to prevent it from being viewed or accessed by unauthorized individuals. When it comes to protecting sensitive or secret video footage, such as proprietary information, personal movies, or classified material, these strategies are frequently utilized. The protection of video content is of crucial importance in a variety of businesses and situations such as Video Conferencing and Communication, Surveillance Systems, Government and Military Applications, Health care, IoT applications, etc. This report explains our work and the implementation of video encryption and decryption using symmetric key.

2. OUR WORK

The main idea behind this study is to use a single private key to both encrypt and decrypt the desired video. The length of the private key is not fixed here and it can be a combination of different data types. By raising the complexity of encryption and making it more resistant to certain sorts of assaults, a variable key length brings about an increase in the level of security that can be achieved through cryptography. For the purposes of encryption and decryption, the term "key length" refers to the number of bits or characters that are contained within the cryptographic key. We have developed a user-friendly Web App through which the sender can encrypt his/her sensitive video securely and the receiver can use the private key to decrypt the same.

3. VIDEO ENCRYPTION

Videos consist of a sequential set of frames. Each frame is of a defined height and width. The frames can be individually considered as an image, i.e., made up of millions of pixels. Video encryption in the proposed project is done by shuffling the individual pixels in a frame and overall shuffling over the set of frames. The shuffling order is done based on an encryption key, generated randomly within the range (0, width) and (0, height) of a frame. The password entered by the user is used to set the random state using the Numpy library's inbuilt function. The entered password is converted into its byte array representation and is passed as the parameter to set the random state. Once the random state is set, the properties of the video such as the number of frames, height and width of a frame is obtained. The video is read using the OpenCV library functions and the pixel values of the video frames are also obtained. Now two random shuffling orders are generated using the np.arange() function one to shuffle the rows of each frame and one to shuffle the columns of each frame. The height of the frame and the width of the frame are provided as the parameters of function respectively. A similar order is generated for the number of frames as well. After the shuffling orders are generated the frames are read one by one and each row and column of the frame is shuffled according to the respective shuffling orders. The frames are further shuffled according to the generated frame shuffling order. The frames are then written as a video file using OpenCV's VideoWriter function. The encryption process ensures that the encrypted videos are playable but does not provide any crucial information to the attacker.

4. VIDEO DECRYPTION

The decryption module is similar to the encryption module. First, the password

entered by the user is used to set the random state. Similar to the encryption process, encryption key generation takes place. Once the shuffling order for the rows and columns of a frame is generated, a reverse key is generated. The reverse key is generated by comparing the index values of the generated key and the value at that index. Once the reverse key is generated the encrypted rows and columns are reversed. The password entered by the user ensures that the generated key is always the same and the correct shuffling order used for encryption is generated. If a wrong password is generated, the byte array representation will vary and the random state set will not be the same as the one used during encryption. The user does not have any restrictions on the password to be entered. Since the password is converted into its byte array representation, the user may enter letters, numbers and characters in their preferred combination of the three. Once the pixel values are decrypted, the video is written into the ‘mp4’ format using OpenCV’s VideoWriter function. The user can then download the decrypted video and view the original contents.

5. FLASK SERVER

The proposed project has been set up using the Flask Web Framework. It is a web framework that allows developers to easily build lightweight web applications. The flask server obtains the video uploaded by the user and initializes the ‘VEnc’ class by passing the video file to the constructor. The password is set in the server to generate encryption and decryption shuffling orders as well. Then upon verifying the user’s selection either encryption or decryption is done. Later the the path to the output file is sent to the frontend so that users can download the file required,

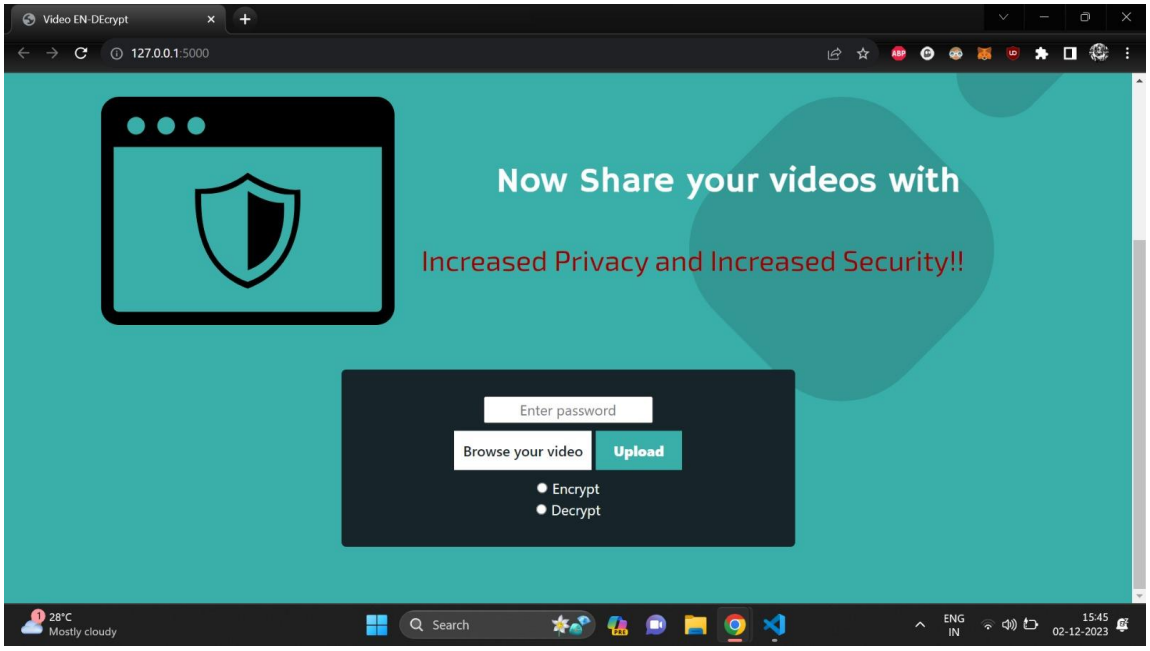
6. FRONT-END DEVELOPMENT

The user interface (UI) and user experience (UX) parts of an application are the

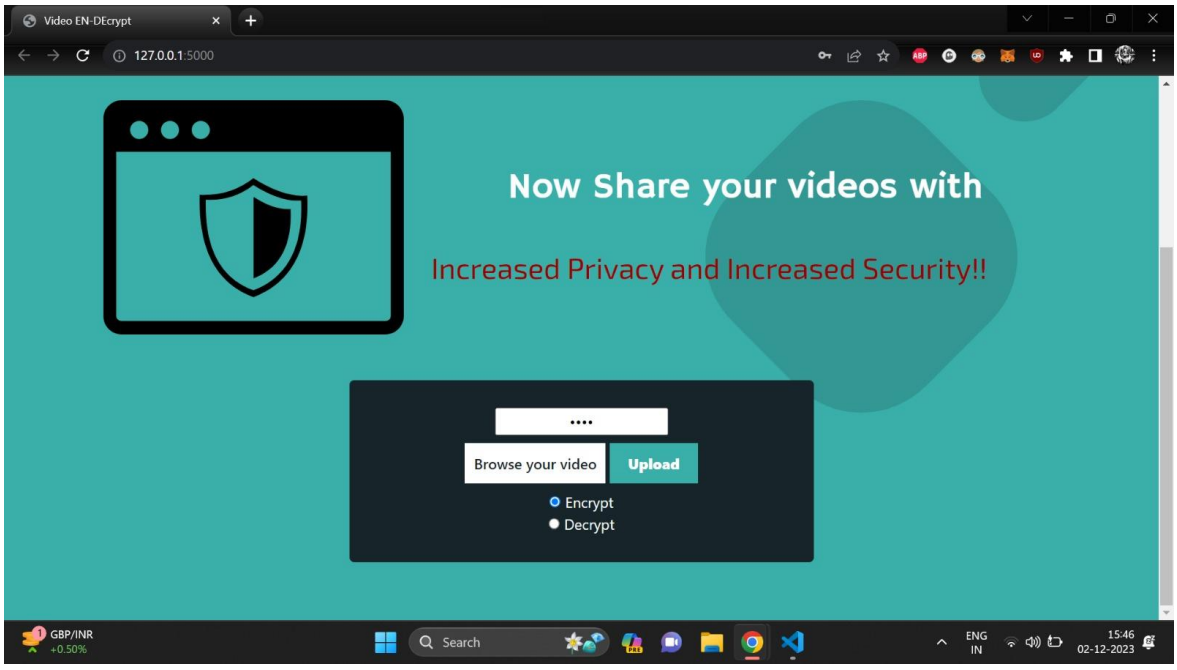
primary focuses of front-end development, which is an absolutely essential part of the process of developing software and websites. The front end is the part of the website that users engage with directly, and it includes the design, layout, and functioning of the visual elements that users see and interact with. We have used HTML and CSS for this need. Initially the sender needs to browse the required video from his/her system and enter the private key. After this process, we have presented two buttons namely, ENCRYPT and DECRYPT. The sender must click the encrypt button and the encrypted video will be generated. The sender can download now and send to the receiver through a secured channel. The receiver would have received the encrypted video and once he/she on entering the same private key, they can decrypt the video.

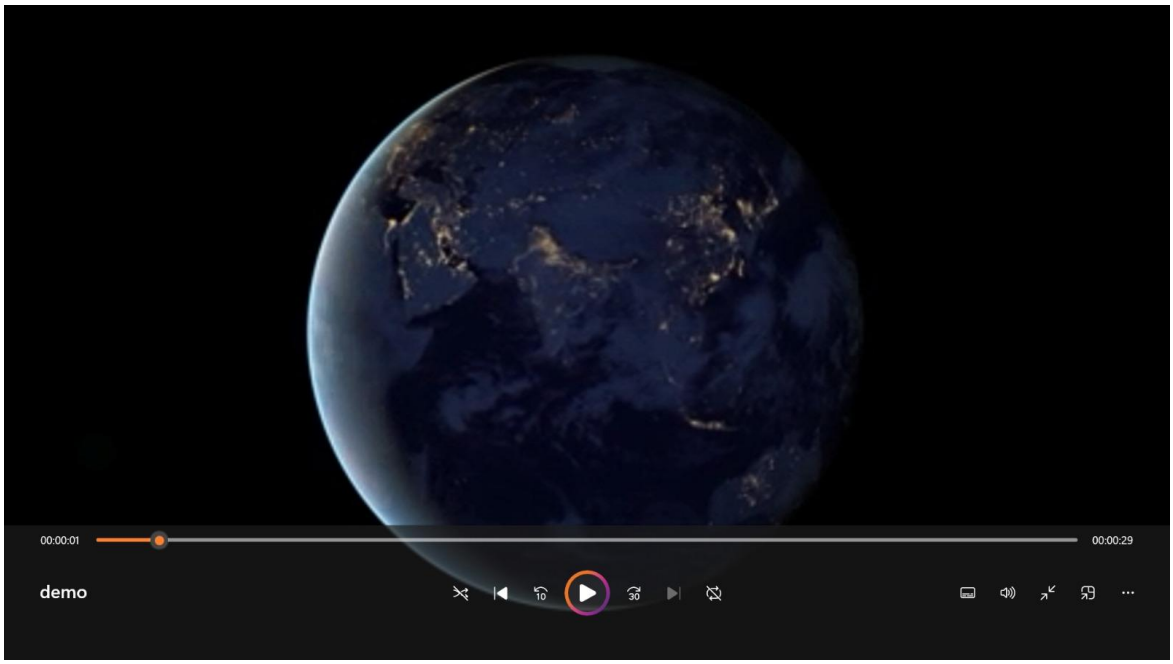
7. WEBAPP REAL TIME SCREENSHOT

- **HOME PAGE**

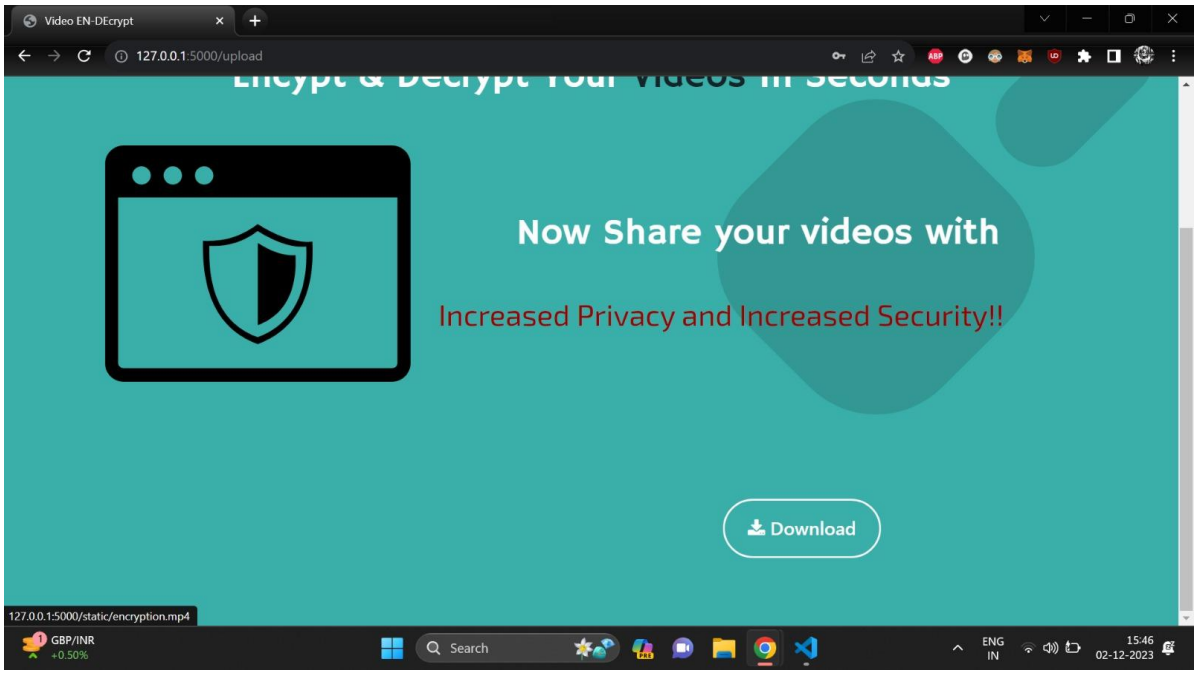


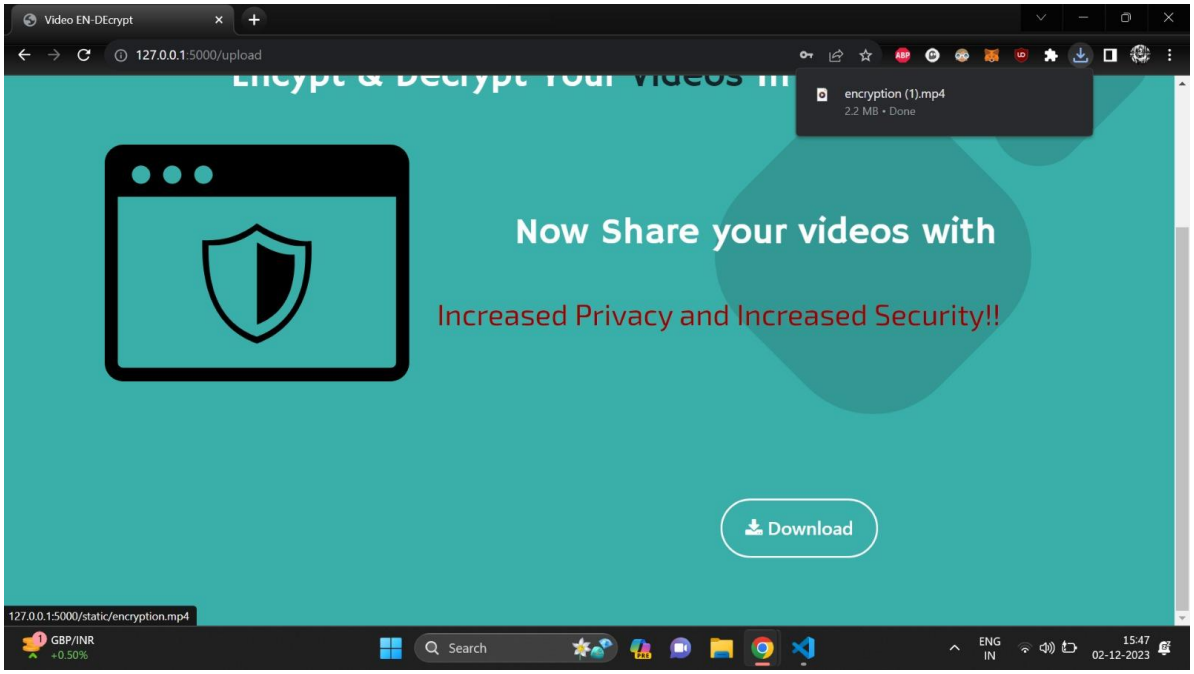
- **BROWSING THE ORIGINAL VIDEO, ENTERING PRIVATE KEY AND CLICKING THE ENCRYPT BUTTON**





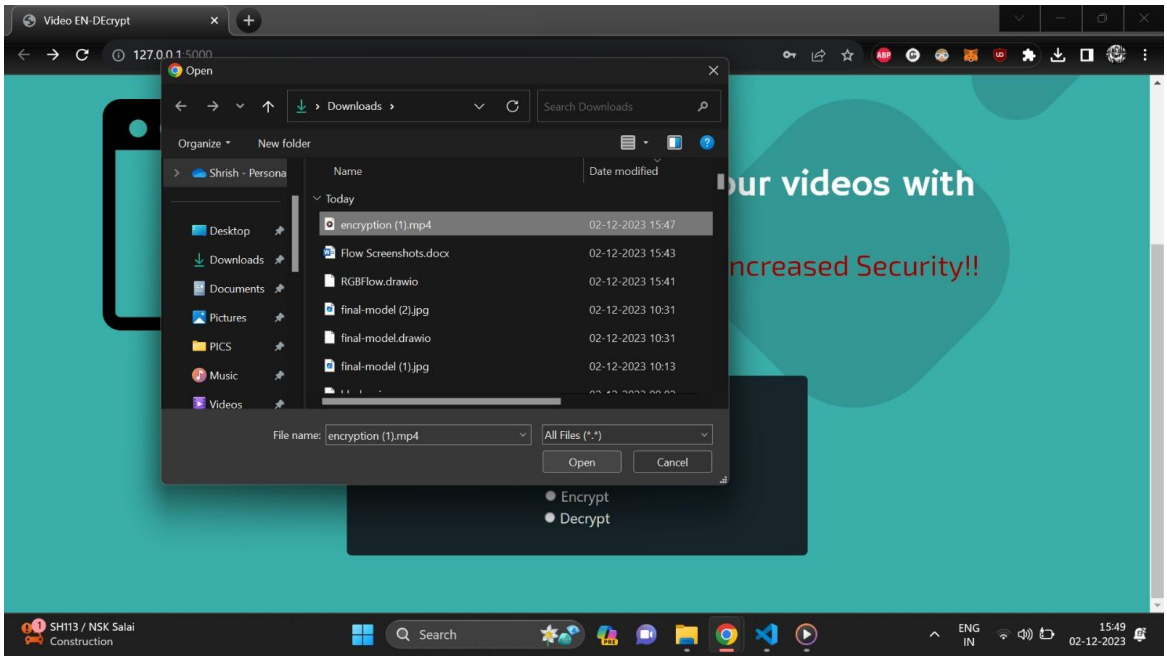
- **COMPLETION OF ENCRYPTION AND CLICKING DOWNLOAD BUTTON**



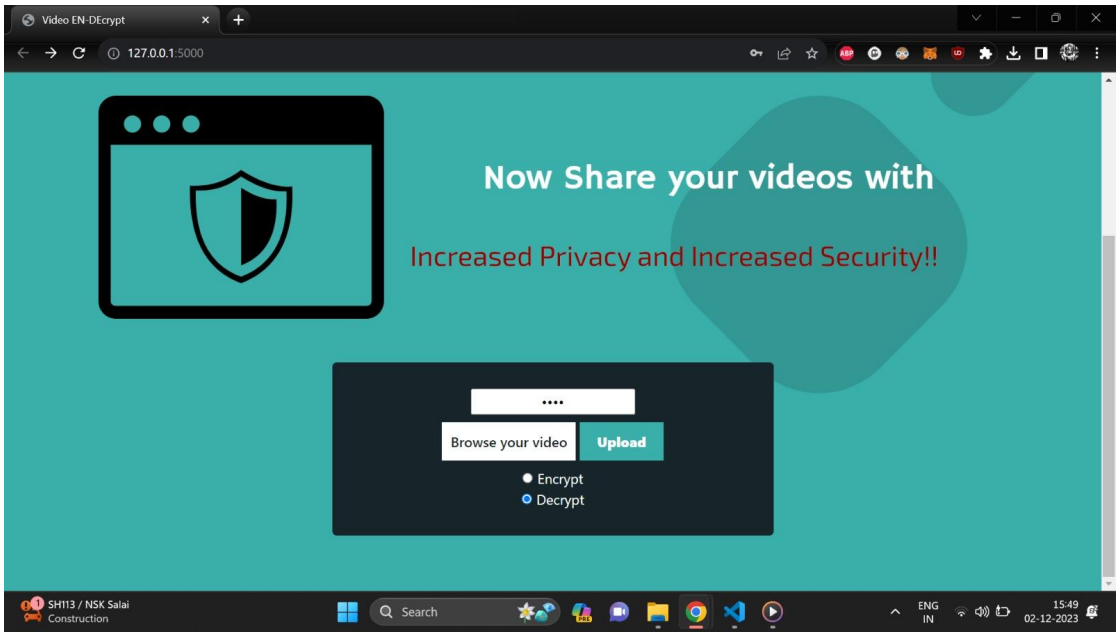


- **OPENING THE ENCRYPTED VIDEO**

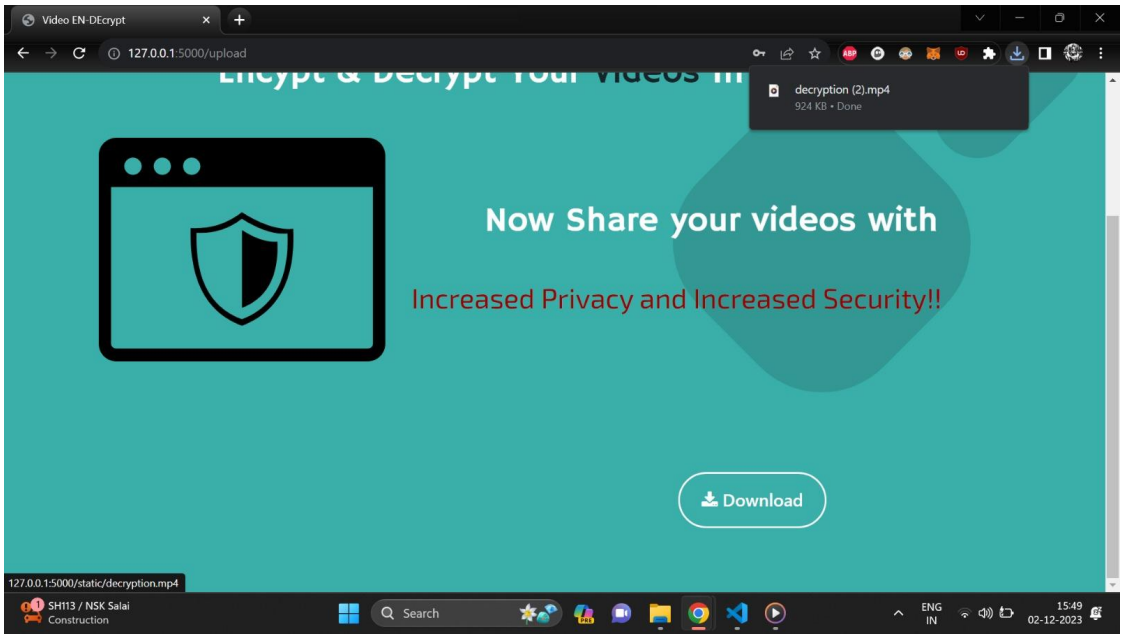
- **BROWSING THE ENCRYPTED VIDEO**



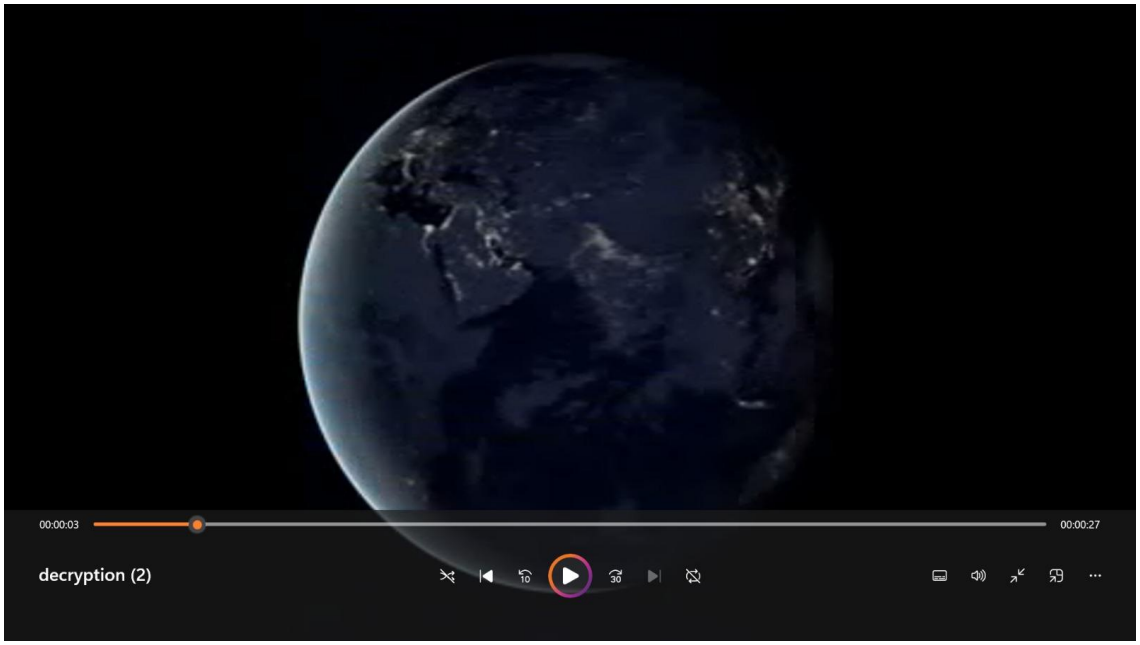
- **ENTERING PRIVATE KEY AND CLICKING THE DECRYPT BUTTON**



- **COMPLETION OF DECRYPTION AND CLICKING DOWNLOAD BUTTON**



- **OPENING THE DECRYPTED VIDEO**



8. FUTURE WORK

It is possible to enhance the functionality of the system that we have designed to include alternative forms of communication, such as audio, images, text, and so on. It is possible to use React.js for the construction of a web application, which would allow the application to be compatible with both Android and iOS. The usage of symmetric keys, which can also be upgraded to asymmetric keys and digital signatures that are more difficult, has been implemented in this work.

8. CONCLUSION

In conclusion, Secure multimedia communication, content distribution, and information protection in our linked and digital world are all dependent on video encryption and decryption. In essence, these two processes comprise the backbone of these processes. Not only are they important for maintaining secrecy, but they are also important for ensuring legal compliance, gaining the trust of users, and preventing security breaches and illegal access. As technology continues to progress, it is essential that video encryption techniques continue to be refined and adapted in order to ensure the continuous safety of video material across a wide range of applications and sectors.