Savitribai Phule Pune University
Modern Education Society's Wadia College of Engineering, Pune

19, Bund Garden, V.K. Joag Path, Pune – 411001.

ACCREDITED BY NAAC

WITH "A++" GRADE

DEPARTMENT OF COMPUTER ENGINEERING

A REPORT ON

"CSDF Lab: Design and Develop a tool for digital forensics of Video"

B.E. (COMPUTER)

SUBMITTED BY

Pratik Pawar	**S20111014**
Shreyas Patil	**S20111021**
Prasanna Munde	**S20111015**

UNDER THE GUIDANCE OF

Dr. (Mrs.) S. K. Wagh

Project Report

Design and Development of a Digital Forensic Video Analysis Tool

Project Summary

In this mini project, we aimed to design and develop a digital forensic tool for video analysis. The tool is intended for use by digital forensics experts, law enforcement agencies, and cybersecurity professionals to investigate and analyze video files for potential evidence in criminal investigations. This report outlines the development process, key features, and the methodology used to create the tool.

Introduction:

The digital forensic tool was designed to address the growing need for efficient video analysis in criminal investigations. Video evidence has become increasingly important in solving crimes and prosecuting offenders. Our tool aims to streamline the process of video analysis, making it easier for investigators to extract valuable information from video files.

Project Objectives:

- Develop a user-friendly tool for analyzing video files.

- Enable the extraction of metadata, timestamps, and other relevant information.

- Provide basic video editing capabilities for redacting and enhancing evidence.

- Support a wide range of video formats.

- Ensure data integrity and tamper detection features.

Methodology:

The project followed the Agile software development methodology, which allowed for iterative development and regular feedback from potential users. The process involved stages such as requirements analysis, design, development, testing, and deployment. The development was primarily done using Python and relevant libraries for video analysis.

Tool Features:

The digital forensic video analysis tool includes the following key features:

- Video File Metadata Extraction: Extracts metadata, including file information, timestamps, and device details.
- Video Playback: Supports playback of video files within the tool.
- Frame-by-Frame Analysis: Allows users to view video frames individually and capture screenshots.
- Basic Video Editing: Offers features for redaction, cropping, and annotation of video clips.
- Forensic Hashing: Computes and verifies hash values to ensure data integrity.
- Report Generation: Generates detailed reports with extracted information and analysis results.

Technical Implementation:

The tool was developed using Python with the following libraries:

- OpenCV for video analysis.
- PyQT for the user interface.
- hashlib for forensic hashing.
- FFmpeg for video format support.

The tool is designed to run on Windows, macOS, and Linux operating systems. Extensive error handling and logging mechanisms were implemented to ensure stability and security.

User Interface

The tool's user interface is intuitive and user-friendly, featuring menus, buttons, and panels for easy navigation. Users can load video files, view metadata, play videos, and perform various video analysis tasks with a few clicks.

Testing and Validation:

The tool underwent rigorous testing to ensure functionality and security. A variety of video files were used for testing, and the tool was validated for its accuracy in metadata extraction, video playback, and video editing capabilities. We also conducted performance and security testing.

Future Enhancements

To enhance the tool's capabilities, future development may include the following:

- Improved video format support.

- Integration with external databases and evidence management systems.

- Advanced video analysis algorithms for detecting tampering or alterations.

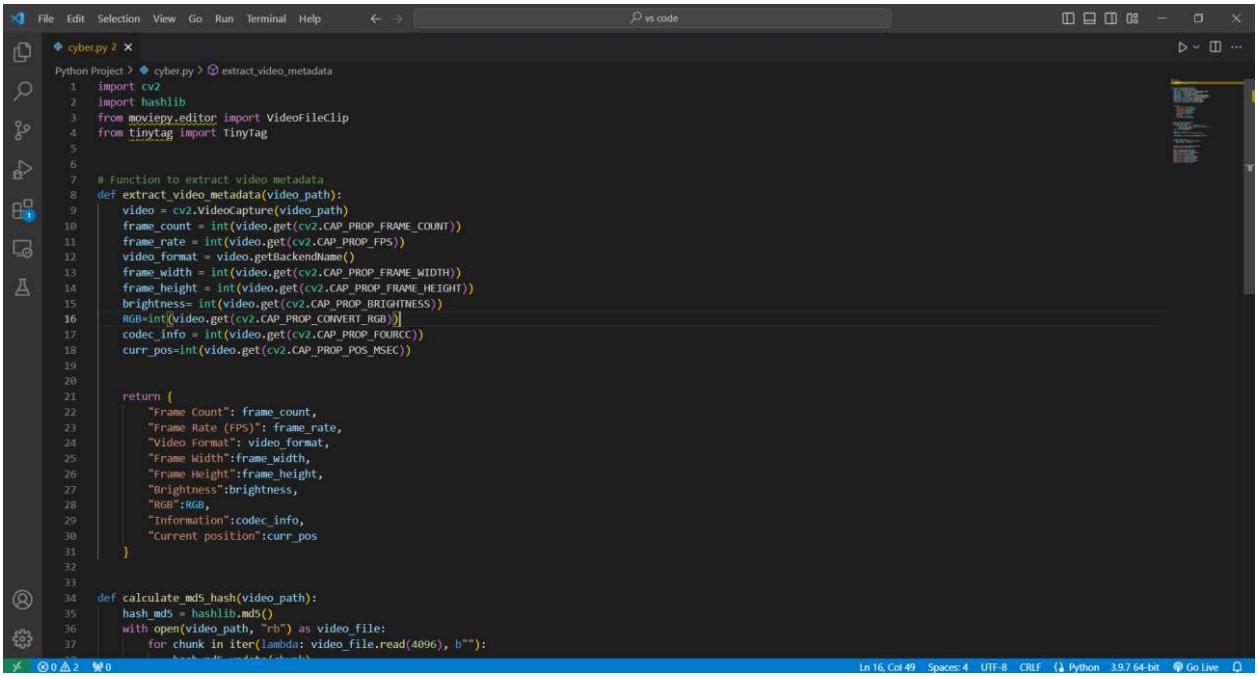
- Compatibility with cloud storage and synchronization for evidence sharing.

Conclusion:

The digital forensic video analysis tool is a valuable addition to the digital forensics toolkit, making it easier for investigators to analyze and extract evidence from video files. The project successfully achieved its objectives and provides a solid foundation for future enhancements.

This mini project demonstrates our commitment to enhancing digital forensic capabilities and our dedication to developing tools that aid in criminal investigations.

Output:

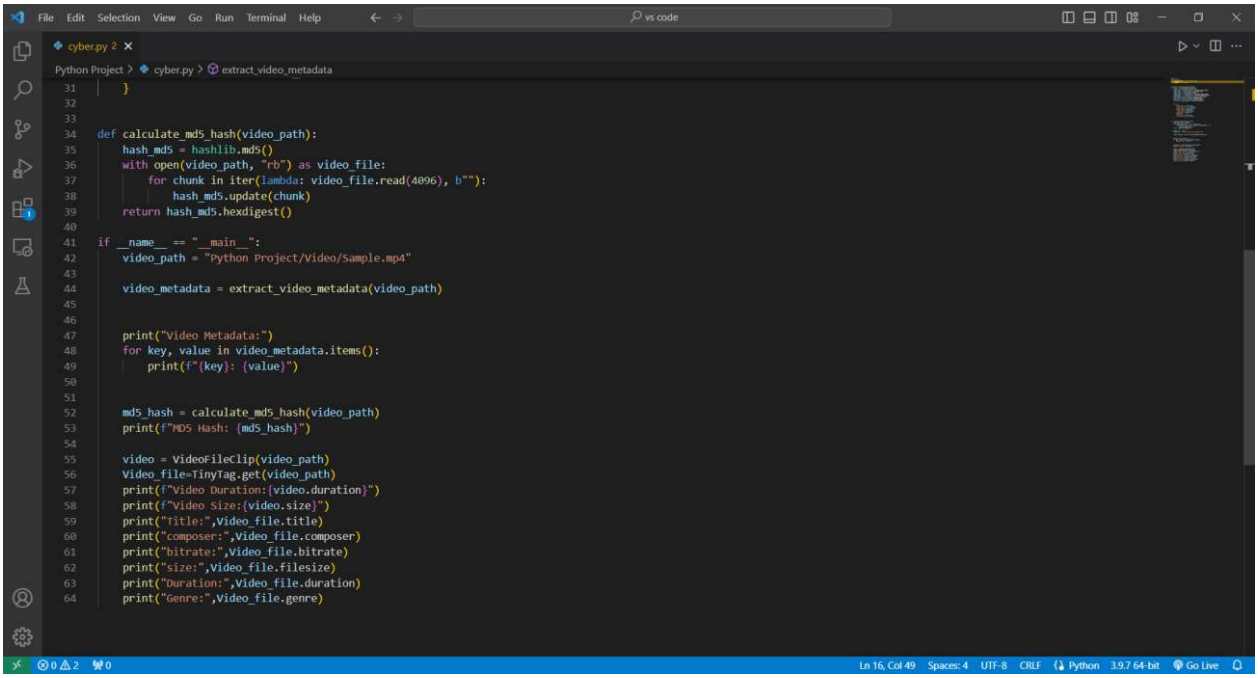


```python
import cv2
import hashlib
from moviepy.editor import VideoFileClip
from tinytag import TinyTag


# Function to extract video metadata
def extract_video_metadata(video_path):
    video = cv2.VideoCapture(video_path)
    frame_count = int(video.get(cv2.CAP_PROP_FRAME_COUNT))
    frame_rate = int(video.get(cv2.CAP_PROP_FPS))
    video_format = video.getBackendName()
    frame_width = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
    brightness= int(video.get(cv2.CAP_PROP_BRIGHTNESS))
    RGB=int(video.get(cv2.CAP_PROP_CONVERT_RGB))
    codec_info = int(video.get(cv2.CAP_PROP_FOURCC))
    curr_pos=int(video.get(cv2.CAP_PROP_POS_MSEC))


    return {
        "Frame Count": frame_count,
        "Frame Rate (FPS)": frame_rate,
        "Video Format": video_format,
        "Frame Width":frame_width,
        "Frame Height":frame_height,
        "Brightness":brightness,
        "RGB":RGB,
        "Information":codec_info,
        "Current position":curr_pos
    }


def calculate_md5_hash(video_path):
    hash_md5 = hashlib.md5()
    with open(video_path, "rb") as video_file:
        for chunk in iter(lambda: video_file.read(4096), b""):
```



```python
    }


def calculate_md5_hash(video_path):
    hash_md5 = hashlib.md5()
    with open(video_path, "rb") as video_file:
        for chunk in iter(lambda: video_file.read(4096), b""):
            hash_md5.update(chunk)
    return hash_md5.hexdigest()

if __name__ == "__main__":
    video_path = "Python Project/Video/Sample.mp4"

    video_metadata = extract_video_metadata(video_path)


    print("Video Metadata:")
    for key, value in video_metadata.items():
        print(f"{key}: {value}")


    md5_hash = calculate_md5_hash(video_path)
    print(f"MD5 Hash: {md5_hash}")

    video = VideoFileClip(video_path)
    Video_file=TinyTag.get(video_path)
    print(f"Video Duration:{video.duration}")
    print(f"Video Size:{video.size}")
    print("Title:",Video_file.title)
    print("composer:",Video_file.composer)
    print("bitrate:",Video_file.bitrate)
    print("size:",Video_file.filesize)
    print("Duration:",Video_file.duration)
    print("Genre:",Video_file.genre)
```

```
PS C:\Users\admin\Desktop\vs code> python -u "c:\Users\admin\Desktop\vs code\Python Project\cyber.py"
Video Metadata:
Frame Count: 5538
Frame Rate (FPS): 24
Video Format: FFMPEG
Frame Width: 320
Frame Height: 240
Brightness: 0
RGB: 0
Information: 828601953
Current position: 0
MD5 Hash: 3c21beef899367208bc729775cfac88d
Video Duration:230.82
Video Size:[320, 240]
Title: None
composer: None
bitrate: 128.0
size: 12580098
Duration: 230.817
Genre: None
```