PROJECT

Student Name: MD SAIF AHMAD **UID: 20BCS5822**
Branch: CSE **Section/Group: 4 - A**
Semester: 7th
Subject Name: CV LAB **Subject Code: 20CSP-422**

1. Aim/Overview of the practical:

Investigate deepfake technology, exploring technical aspects, ethical considerations, and societal impact through experimentation.

2. Software:

Any Python IDE (e.g. GoogleColab)

3. Code:

```
from typing import Optional
import cv2


from roop.typing import Frame


def get_video_frame(video_path: str, frame_number: int = 0) -> Optional[Frame]:
    capture = cv2.VideoCapture(video_path)
    frame_total = capture.get(cv2.CAP_PROP_FRAME_COUNT)
    capture.set(cv2.CAP_PROP_POS_FRAMES, min(frame_total, frame_number - 1))
    has_frame, frame = capture.read()
    capture.release()
    if has_frame:
        return frame
    return None



def get_video_frame_total(video_path: str) -> int:
    capture = cv2.VideoCapture(video_path)
```

```
    video_frame_total = int(capture.get(cv2.CAP_PROP_FRAME_COUNT))
    capture.release()
    return video_frame_total


from typing import Optional
import cv2

from roop.typing import Frame



def get_video_frame(video_path: str, frame_number: int = 0) -> Optional[Frame]:
    capture = cv2.VideoCapture(video_path)
    frame_total = capture.get(cv2.CAP_PROP_FRAME_COUNT)
    capture.set(cv2.CAP_PROP_POS_FRAMES, min(frame_total, frame_number - 1))
    has_frame, frame = capture.read()
    capture.release()
    if has_frame:
        return frame
    return None



def get_video_frame_total(video_path: str) -> int:
    capture = cv2.VideoCapture(video_path)
    video_frame_total = int(capture.get(cv2.CAP_PROP_FRAME_COUNT))
    capture.release()
    return video_frame_total

import threading
from typing import Any, Optional, List
import insightface
```

```
import numpy


import roop.globals

from roop.typing import Frame, Face


FACE_ANALYSER = None

THREAD_LOCK = threading.Lock()




def get_face_analyser() -> Any:

    global FACE_ANALYSER


    with THREAD_LOCK:

        if FACE_ANALYSER is None:

            FACE_ANALYSER = insightface.app.FaceAnalysis(name='buffalo_l',
providers=roop.globals.execution_providers)

            FACE_ANALYSER.prepare(ctx_id=0)

    return FACE_ANALYSER




def clear_face_analyser() -> Any:

    global FACE_ANALYSER


    FACE_ANALYSER = None




def get_one_face(frame: Frame, position: int = 0) -> Optional[Face]:

    many_faces = get_many_faces(frame)

    if many_faces:

        try:

            return many_faces[position]
```

```
        except IndexError:
            return many_faces[-1]
    return None


def get_many_faces(frame: Frame) -> Optional[List[Face]]:
    try:
        return get_face_analyser().get(frame)
    except ValueError:
        return None


def find_similar_face(frame: Frame, reference_face: Face) -> Optional[Face]:
    many_faces = get_many_faces(frame)
    if many_faces:
        for face in many_faces:
            if hasattr(face, 'normed_embedding') and hasattr(reference_face, 'normed_embedding'):
                distance = numpy.sum(numpy.square(face.normed_embedding - reference_face.normed_embedding))
                if distance < roop.globals.similar_face_distance:
                    return face
    return None
```

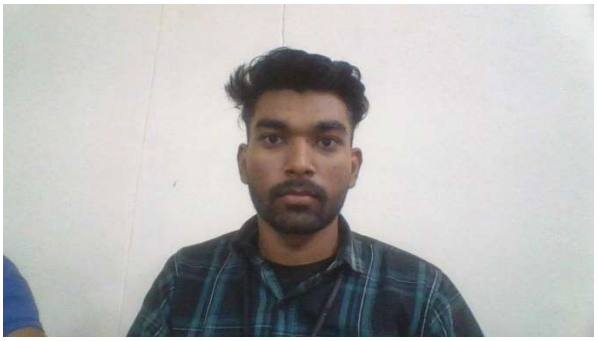
```
%cd "/content/roop"

!python run.py -s "face2.png" -t "brad org.mp4" -o "face_changed_video_v2.mp4" --keep-frames --keep-fps --temp-frame-quality 1 --output-video-quality 1 --execution-provider cuda
```

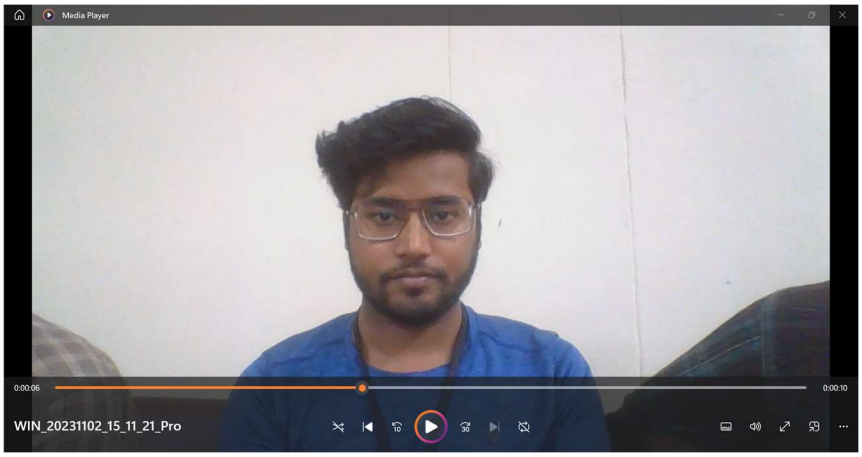
```
%cd "/content/roop"

!python run.py -s "/content/photo.jpg" -t "/content/video.mp4" -o "face_restored_video3.mp4" --keep-frames --keep-fps --temp-frame-quality 1 --output-video-quality 1 --execution-provider cuda --frame-processor face_swapper face_enhancer
```

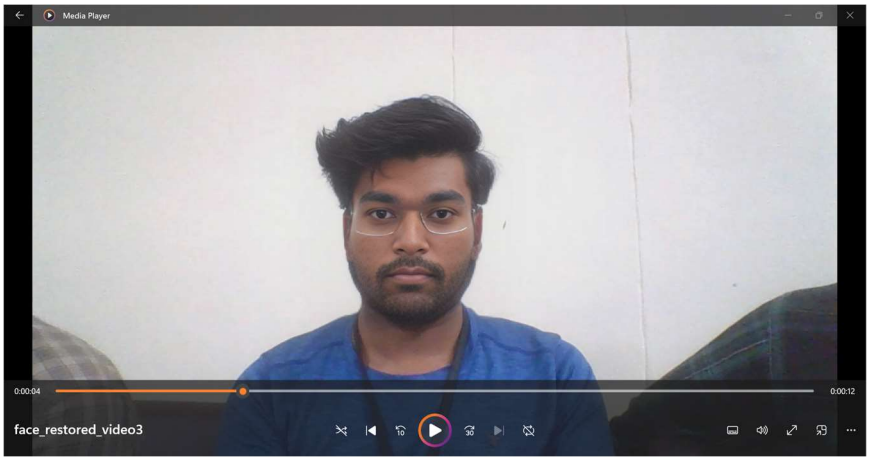
4. Output:



Swapping Face image



Swapping Face Video



Final Face Swapped Video

Project Video Link: https://drive.google.com/file/d/1p2UsUBaq1q5NNh-vraDOR3bJ6e6ttIUL/view?usp=sharing