Description:- We are solving the problem statement by Unacedemy
Platforms Used:- Python, Open CV, NumPy
Front end:- Bootstrap, Sass, HTML, CSS, Javascript Canvas

Scenario 1 -
Video background removal:- In this, we are identifying a uniform background like Curtain or wall of any uniform color and replace with some transparent pixels at the frontend.

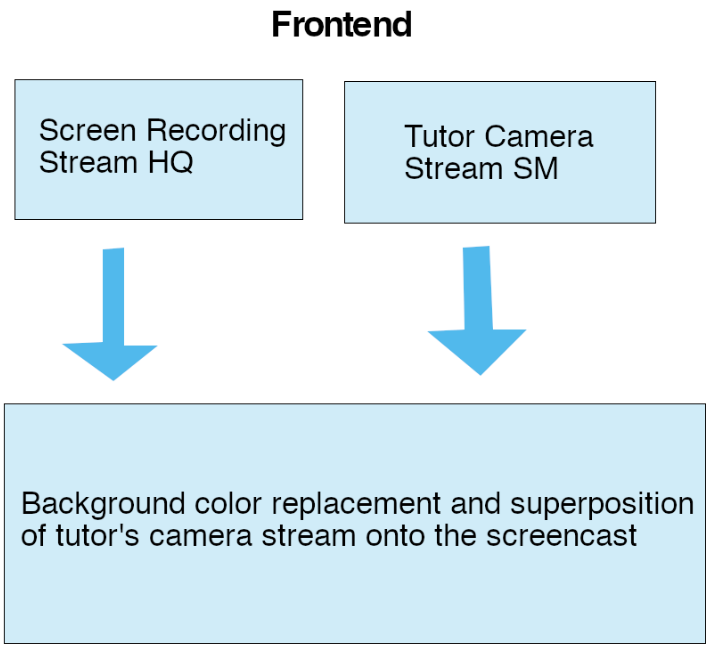
Scenario 2 -
Identifying the object in the video and filleting that objects by using python, Open CV, NumPy
The backend code ingests the live stream and replaces the background pixels with a uniform color pixel that is sent to the frontend.
The stream is also persisted based on the resource ID, so that it can be replayed if required in a stream friendly format like mp4.
The processed output is streamed to the frontend where the Javascript uses the logic used in scenario 1 to overlay the tutor onto the live screencast.

We are not superpositioning the tutor camera stream onto the screencast at the backend, since the screencast needs to be of higher quality but has very little entropy due to the almost static nature of screens (Usually only the cursor or mousepointer changes and almost all of the screen remains constant).
But, the tutor's camera feed can be of lower quality. And has a higher level of entropy.
Handling the two streams independently will pay back substantially in scaleability and flexibility of options in the future.

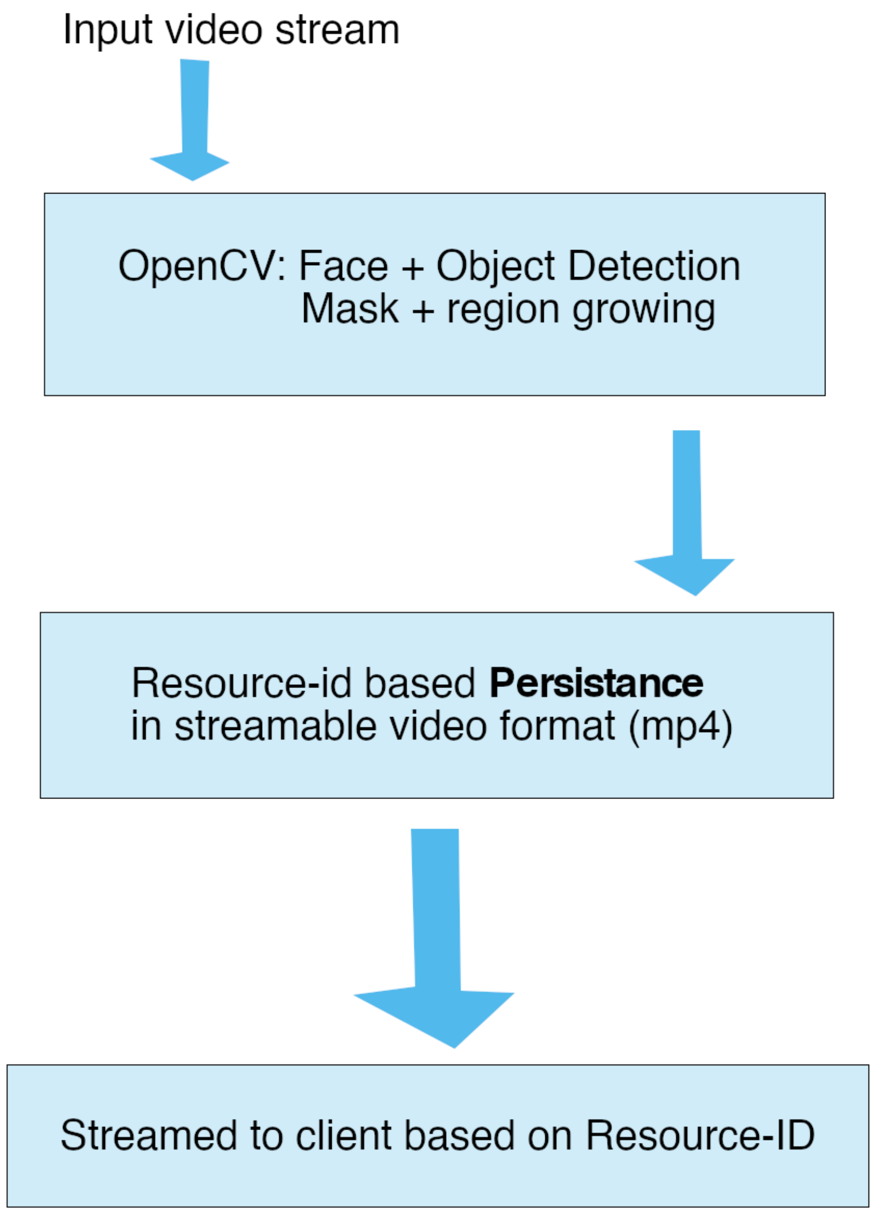
Conclusion - We are not only majorly focusing on the accuracy of background and object removal. We are also considering the scalability and flexibility of the architecture we are adopting.

Team Members:-

1-Rakesh P Gopal
2-Rakshitha S
3-Lokesh Kumar
4-Shivang Jain


Frontend
Screen Recording Stream HQ
Tutor Camera Stream SM
Background color replacement and superposition of tutor's camera stream onto the screencast

Backend


Input video stream
OpenCV: Face + Object Detection
Mask + region growing
Resource-id based **Persistance**
in streamable video format (mp4)
Streamed to client based on Resource-ID