**Logo

Description automatically generated**

***Computer Network Lab***

***Project Report***

### *Course Name:* Computer Network Lab

### *Course Code:*CL - 3001

### *Instructor:* Sir. Ubaidullah

***Group Members:***

1. Muhammad Taha Majeed Khan (21K-3316)
2. Bilal Shakeel (21K-4874)
3. Muhammad Samamah (21K-3205)

***Table Of Content:***

**Table Of Content** **Page**

**Introduction 3**

**Implementation 4**

**Working 4**

**Conclusion 5**

***Project Report***

***Internet Radio Multicasting Multimedia Over IP***

# ***Introduction:***

The goal of this project is to simulate how a radio station would actually operate, with each station giving unique content to a networked client; however, instead of serving clients with audio data, we would serve them with video. It resembles a hybrid of video streaming and radio.

Our program uses the OpenCV library and Python sockets to implement a multi-station video streaming application.

The application consists of a connected client and server, from which a client can choose a video station and begin receiving video streams. The video stream data is sent to the multicast group by the server in response to an incoming request from a client, which is then received by the client and shown in a window.

Project Files:

* 1. “server.py”
  2. “client.py”
  3. “video.mp4” → Hosted on Station 1 on Server Side.
  4. “sample.mp4” → Hosted on Station 2 on Server Side.

# ***Implementation:***

The program is written in Python and makes use of the OpenCV package for processing and streaming video as well as socket programming to facilitate communication between the client and server. Separate files named "server.py" and "client.py" contain the server and client code, respectively.

# ***Working:***

## ***Server Side:***

* Establishes a UDP socket to accept client data.
* Configures a multicast group to send information to several clients at once.
* Extracts video attributes from two stations, such as resolution, aspect ratio, and frame rate, using the videoprops library.
* Awaits the client to send a "start" message before starting a conversation.
* Delivers the client the retrieved video properties.
* Holds out on sending the client the video stream of the station the client has chosen out of the two.
* Splits the video into JPEG frames, which are then sent at a 30 frames per second rate to the multicast group.
* Sends the video stream of the selected station and enables the client to switch stations at any moment.

## ***Client Side:***

* Establishes a UDP socket to receive data from the multicast group and transfer data to the server.
* Enters the multicast group that the server has created.
* Commences communication with the server by sending a "start" message.
* Obtains from the server the station's details.
* Shows the information about the station.
* Asks the user which of the two stations they would like to stream the video clip from.
* Sends the user's input to the server in order to select a station.
* Obtains the chosen video file's JPG frames from the multicast group and shows them in a GUI window at a 30 frames per second pace.
* The user can pause the video stream by pressing 'p' while it is being played in the GUI window.
* The user can choose from the following options while the video stream is paused:

i) Exit

ii) Change the Station

iii) Resume

Client termination and client-server connection closure occur with exit.

When a client wants to transfer stations, the server notifies the client and provides the client the video stream from the new station.

Resuming the paused video stream is called resume.

In general, the program enables a client to stream video over a multicast group from the server to a selected station.

# ***Conclusion:***

To sum up, this project successfully implemented an OpenCV library and Python sockets-based multi-station video streaming application. A client can use the program to choose a station and begin streaming video from it. Clients receive the video stream data and play it in a graphical user interface (GUI) window once the server delivers it to the multicast group. Applications for video streaming that need to stream from multiple stations at once can use this application.