REAL-TIME E-LEARNING PLATFORM

Final Report for Lecture Live Streaming Module

Project ID: 19-079

A.M.H.B. Athapaththu

P.G.R.S.H. Gamlath

H.M.S.S. Herath

W.A. Geeth Sameera

Bachelor of Science (Hons) in Information Technology Specialized in Software Engineering

Department of Software Engineering

September 2019

REAL-TIME E-LEARNING PLATFORM

Project ID: 19-079

A.M.H.B. Athapaththu

P.G.R.S.H. Gamlath

H.M.S.S. Herath

W.A. Geeth Sameera

Supervisor's name – Dr. Malitha Wijesundara

Department of Software Engineering

September 2019

DECLARATION

We declare that this is our own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, we hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute our dissertation in whole or part in print, electronic or other medium. We retain the right to use this content in whole or part in future works (such as article or books).

Name	Student ID	Signature	Date
A.M.H.B.Athapaththu	IT16131002		
P.G.R.S.H.Gamlath	IT16124936		
H.M.S.S.Herath	IT16132306		
W.A.Geeth Sameera	IT16118096		

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the Supervisor: Date:

ABSTRACT

Real-time e-learning platform is a WebRTC based live teaching platform which facilitates the e-learning requirements of universities and other institutes. This solution includes lecture live streaming, lecture playback, a vector based interactive whiteboard, chatting and file sharing module and a real-time lecture movement tracking module using a PTZ camera. The system is capable of streaming two simultaneous streams of a 1080p camera and a 720p screen capture seamlessly using a network connection with 256KB/s bandwidth. Live streaming component is very less CPU intensive and it use around 14% of the CPU for streaming 10 simultaneous sessions with 10 listeners per each on a AWS t2.micro instance with 1 vCPU 2.5 GHz, Intel Xeon Family, 1 GiB memory. Because of that, this solution is a very cost effective product compared to existing competitors in the market.

ACKNOWLEDGEMENT

First, I would like to express my gratitude to supervisor Dr. Malitha Wijesundara and co-supervisor Mr. Pramadi Athapaththu for the guidance and mentoring given throughout this period to make this research a success. Also I would like to thank senior lecturer and lecture in-charge for Comprehensive Design and Analysis Projects module, Mr. Jayantha Amararachchi for the valuable assistance given to complete this project.

At last but not least I am also thankful to all my family members and colleagues who have been always helping and encouraging me throughout the research period. I have no valuable words to express my thanks, but my heart is still full of the favors received from every person.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF ABBREVIATIONS

WebRTC	Web Real-Time Communications
RTP	Real-time Transport Protocol
PTZ	Pan Tilt Zoom
CPU	Central Processing Unit
GPU	Graphical Processing Unit
Yolo3	You Look Only Once Version 3
SORT	Simple Online and Real-Time Tracking
FPS	Frames Per Second
Deep SORT	Simple Online and Real-Time Tracking with a Deep Association Metric
PyTorch	An open source machine learning library based on the Torch library

OpenCV	Open source computer vision. A library of programming functions mainly aimed at real-time computer vision
KB	Kilo Byte
MPEG	Moving Picture Experts Group
DASH	Dynamic Adaptive Streaming over HTTP
SQHW	Student Questioning Handling & Whiteboard

1. INTRODUCTION

Real-time e-Learning platform is a live lecture streaming web application which is developed focusing on Universities and other educational institutes. Lecturers can stream lectures from or out of the institute premises remotely while students can join relevant live streams from anywhere. This report will describe the live streaming component of the application.

Lectures streamed from institute premises will be equipped with a PTZ camera to get a third person perspective camera feed of the lecturer. Lecturers can use their web camera instead of PTZ camera if they are streaming remotely outside of the institute premises. Apart from the camera feed, Lecturer's Computer screen is captured and streamed. Lecturer can select between which screen, which application to share. Also audio feed from a mic attached to the lecturer's PC is broadcasted. The application is capable of handling multiple simultaneous lecture sessions without any interruptions to each other. This component is also responsible for recording live streams to facilitate the lecture playback where students are given the capability to rewind and playback already streamed lectures for future reference. Student Questions Handling & Whiteboard (SQHW) is one of the major component of the real time e-Learning system which enhances the interaction between lecturer and the student connected via internet by introducing real time questioning facility. There is a whiteboard facility which can be used by the students to ask questions more clearly by drawing or writing and this is a special feature with vector based drawing window.

Background Literature

Virtual education is an emerging concept around the world. Students are starting to adapt to learn more productively through internet than traditional classroom due to many reasons. One of the main reasons is that every student has a different pace of catching-up with the teaching where some of them can be considered as fast learners while others may be slow. In traditional classroom scenarios, there is no solution to this fact. Another reason is the schedule flexibility. Through virtual learning, students can access their courses at any time anywhere with internet

giving students the full control of their schedule of the day. Supporting the facts, universities provide students with the access to course materials via internet. Going beyond this, Real-time e-Learning Platform is a solution for the shortcomings in conventional education system mainly with universities. [2][3]

This proposed system is a web-based application where students can watch live lectures and interact with the lecturer and also playback previous lectures online. A tracking camera will be tracking the lecturer movements while recording and both lecturer's video feed and the lecturer's computer screen feed will be streamed. Also this system contains features to interact with the lecture. Student can ask questions from the lecturer remotely via his or her webcam in real-time and also can use a virtual whiteboard that is provided with the system to describe any misunderstandings.

Research Gap

When we consider about existing competitors of this field, the latency of the live streams is high where we cannot categorize their live streams as real-time. Main fact which has been affected to those is maintaining the quality of the live streams while delivering in real-time. Also supporting live streams for almost all the devices including mobile devices and different browsers is another focus of this research.

Compared to existing competitors of this field, most of them uses a fixed camera/cameras to take the feed of the lecturer moving in the lecture hall. If it's one camera, then lecturer may not be visible in some parts of the lecture hall based on the positioning of the camera. To address that issue, cameras with wide view is used to covers the entire lecture hall. Then the students can't see the lecturer properly. In a multi camera setups, some program is used to switch camera feeds according to the lecturer position. That will add extra cost without improving the student experience significantly. Both these approaches have major impact on the student experience on using an e-learning platform.

Systems / Feature	Eduscope	Proposed system
Real-time streaming	Yes with considerable latency	yes
Adaptive Bitrate enabled video player	No	Yes
Lecture tracking	Yes	Yes with capability of tracking using face and body
Vector based Whiteboard	No	Yes
Text-based lecturer student interactions	Yes	Yes
Video-based lecturer student interactions	No	Yes
Poll feature	No	Yes
Statistics and analysis of usage of content	No	Yes
Secure video player with content protection	No	Yes
Voice to text conversion of video content	No	Yes

Table 1.1: Comparison with Eduscope

Research problem

The most common teaching and learning practice adopted by many enterprises has always been a classroom with one or more instructors and learners meeting physically and in real-time. But in this teaching model, there are several drawbacks which will be addressed as problems within this research.

A classroom based learning experience means the class schedule is predetermined and not subject to change. Students must shape their personal schedules around school instead of the other way around. If plans unexpectedly change or an emergency comes up, the student cannot adjust the class schedule to turn in the work at a different time. This is one of the main problems that this research will find solutions for.

Content non-reusability is another problem that can be found in classroom learning. Memorizing or writing all the necessary content while listening to a lecture is difficult. Therefore, students might miss many important points that the lecturer is pointing out during a lecture.

Research objectives

Online Streaming

1. Minimize Latency

   Delivering a stream in real-time is a challenging task due to several factors. Main obstacle is the source of the stream should have a high bandwidth internet connection to broadcast the streams. Also the client should have a high bandwidth to view the stream without any delay. Main objective of this component would be finding solutions to address these issues.

2. Multiple Device / Browser Support

   Students should be able to access the application from any of their devices and watch the streams. Also lecturers should be able to stream from their devices remotely. Therefore, the application should support most of the common devices and browsers.

3. Record Live Streams

   Live streams have to be recorded and stored in order to facilitate students with lecture playback feature. This process should be very light weight and less CPU consuming task.

4. High Scalability

   System should be capable of handle all the load coming onto it. If not, there should be the flexibility of scaling the system to accommodate all the requests.

5. High Quality Streams

   System should have the capability of delivering very high quality streams in bandwidths capable of student's internet connection to handle.

6. Adaptive Bitrates

   Live streams should support adaptive bitrates so that receivers can receive an uninterrupted video stream.

Student Questioning with Whiteboard

The main objective of the SQHW component is to build up a way to handle the real time questioning facility with the ongoing lecture. Student can mainly ask questions using his webcam and microphone through video and audio medias. Rather student can use chat box to interact with lecturer with text and uploading files when necessary.

The developed system is a web based application and SQHW component is focused with two main functions.

1 Text, Voice based questioning

The main objective is to enhanced the interactive between online peers and the publisher. There were several methods implemented for this made success with chat box, with live audio and video feeds.

2 Video based questioning with whiteboard

Vector based drawing window that used when asking questions in live audio and video by the student. At the beginning only lecturer allows to control the whiteboard. Control over the drawing board can be handover to the specific student when he/ she asking questions.

There is special feature implemented for the polling which can make the students motivate to the lecture. Lecturers can create polls and send to all students who connected and get answers. Later answers will be analysis and will be show in a graphical way.

Offline Playback

1. Minimize buffering pauses in playback

   A main objective of the player is to have minimum buffering pauses if not nil. The player should automatically change the source to match the source bitrate with the current network bandwidth.

2. Keep the video and data feeds in-sync

   Video inputs for the player are independent feeds. The player should keep the feeds in-sync all the time to deliver the best user experience. Data feeds recorded from the whiteboard and chat window during the live lecture session should also played in sync with the videos.

3. Data collection and analysis

   User interactions with player should be captured relative to the position of the lecture playback. With these data, a detailed report has to be generated for each student, each lecture and each module.

Lecturer Tracking

1. Real-time lecturer tracking

   Lecturer must be detected and tracked accurately in real time to generate camera control commands. Algorithm should perform efficiently even in low light environments. False detections must be reduced to improve the accuracy of the algorithm

2. Handling multiple person detections (Person Re-Identification)

   If lecturer take a student to the stage or if someone passes across the lecture hall, multiple detections will be there. Since the camera only moves according to the lecturer, algorithm must be able to quickly differentiate between lecturer and the other detections and generate commands bases on that detections throughout the session

3. Camera movements control

   Based on the detections, a PTZ camera has to be moved smoothly and accurately keeping lecturer in the center of the video output. If the detection is failed at some point, camera control module should have a predefined way to find the lecturer again.

2. METHODOLOGY

Methodology

In order to develop this real-time application which is usable from multiple devices as laptops, tablets and smartphones we have chosen WebRTC technology [1]. WebRTC is an open source project developed by google to make web browsers support peer-to-peer communications. On top of that, to implement our solution, we used Janus as the media server. Janus is a general purpose WebRTC server which is capable of implementing WebRTC media communications with browsers [A]. Backend application server is developed with Node.JS while frontend application is implemented using React.JS.

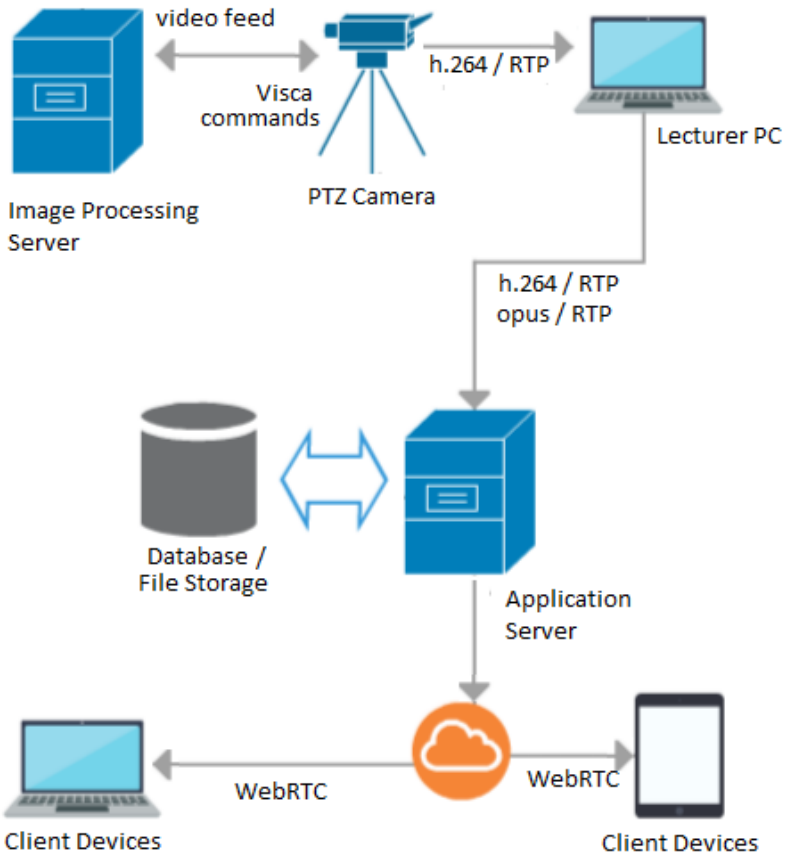

Figure 2.1 High Level Architectural Diagram

Live streaming

Main purpose of the live stream component is to transport video feeds from lecture's pc to student's pc with minimum latency and support multiple streaming sessions at once. Once the lecturer starts to stream, system creates a new streaming session and streams related to that session is broadcasted within the session. Once

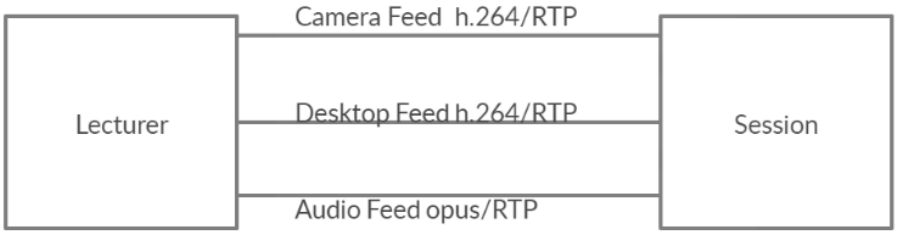

Figure 2.2 Feeds broadcasting within a session

a peer joins to a specific session, they will receive the related streams. Lecturer's computer screen and a third-person perspective camera view of the lecturer are the two video feeds that will be broadcasted within a session. A feed from the lecturer's mic is also streamed to the session. Those video feeds are encoded with h.264 codec and audio feed is encoded with opus codec and send to the media server using RTP protocol.

Once media server receives the streams, they are transcoded and broadcasted via WebRTC.

For the purpose of playing back live streams, all the feeds are recorded and stored in the server as a custom file format **mjr**. A separate file will be created for each audio and video stream which contains structured raw RTP packets exactly as they are received by the server. This operation allows the server to perform recording process as very less CPU intensive operation [13].

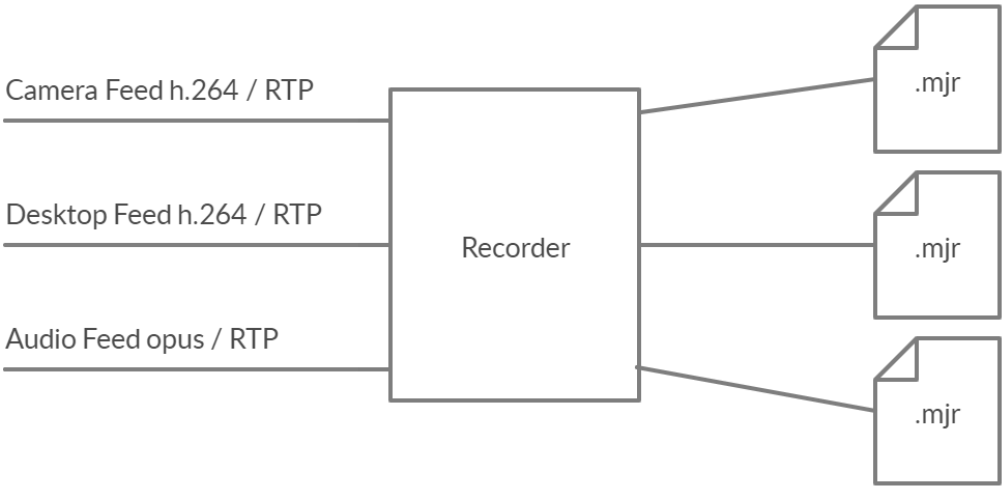

Figure 2.3 Feed recording process

After a steam is completed, media frames from the recorded RTP packets are extracted and saved to a media container in a way that media applications can consume them. Here video recordings are converted into .mp4 file format and audio feed as converted into opus file format.

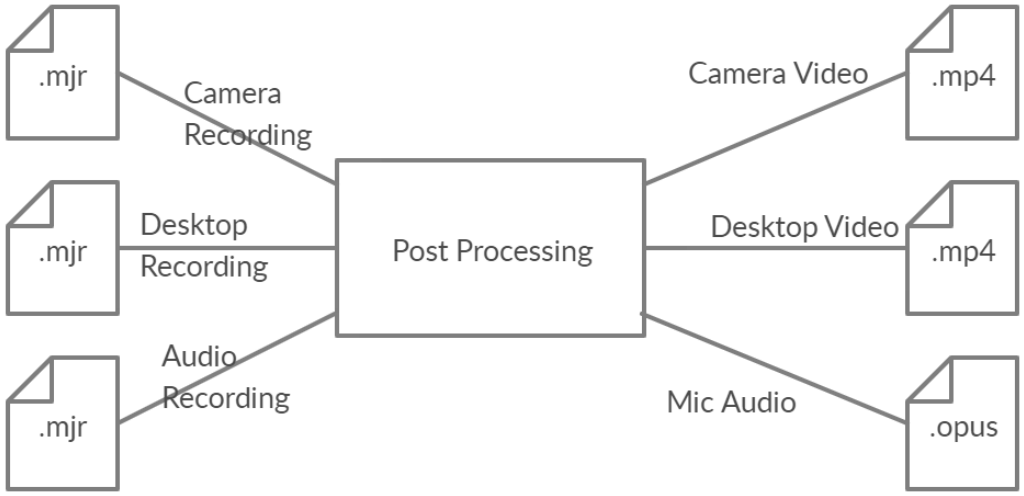

Figure 2.4 Post processing of recordings

Offline Playback

Main purpose of the lecture playback component is to provide a complete and uninterrupted playback experience of a past lecture session. Additionally,

students, lecturers and management should be able to view a comprehensive analysis of the usage data from the player.

In order to be played in a video player with ability to seek, input videos should possess all the necessary key frames. During the streaming process some key frames might not get created in the recorded stream. Therefore, the post processed video streams are passed through another process to recreate key frames at regular intervals [6].

The playback player is developed to support adaptive bitrate streaming. User of the playback player can explicitly select the video quality in which user wants the videos to be played. In order to cater this requirement, the key frame recreated video streams are automatically converted into 3 different video qualities with different resolutions and different bitrates. (1080p, 720p, 360p). After the conversion, each converted video and the original audio will be spitted into 2 second segments which are indexed in order. These segments are then fed into the player for the playback. If the user opted for automatic video quality, the player will select the suitable video stream according to the current network bandwidth of the user.

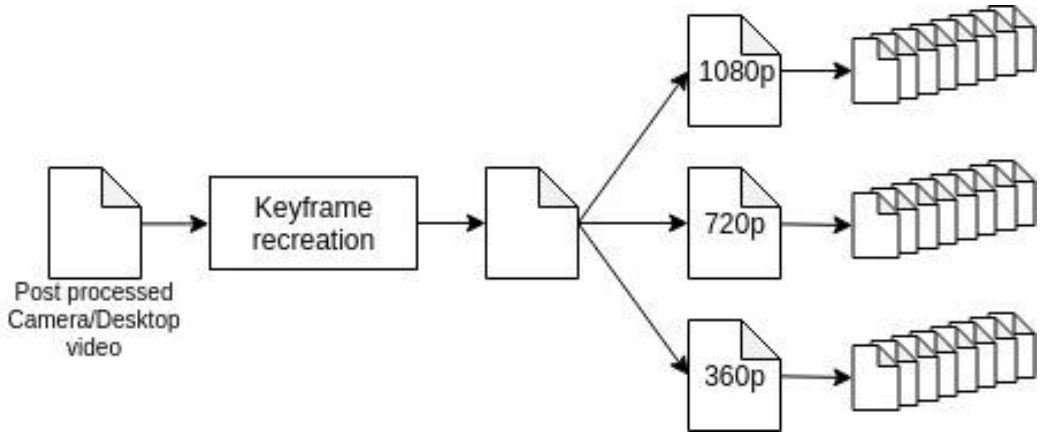

Figure 2.2: Video Conversion Process

In the player, two videos are played in sync along with the audio. Chat data and whiteboard data related to the playing lecture session will be retrieved from the

server and displayed at the relevant timestamp giving the original experience of the live play. User can annotate different sections of the video for future references.

Player will keep track of the sections which are played by the user and get sent to the server. These data combined with the user information and lecture session information, will be used generate statistics. With these statistics, lecturers can have a true feedback on each lecture session they conducted. Management can have an assessment on the lecture sessions based on the students' interactions with the videos. Students can have a self-evaluation based on own interactions with each lecture session.

Lecturer Tracking

Main objective of lecturer tracking component is to automate a special camera (PTZ) using an advance algorithm to do lecturer tracking process throughout the lecture. Using the tracker, set of commands will be generated to control a pan tilt zoom (PTZ) camera. Output video will then be streamed as the lecturer's video alongside with lecturer's pc video. Tracking algorithm consists of mainly three components

1. Person Detector and Tracker
2. Person Re Identification Module
3. Camera Controller

1 Person Detector and Tracker

Person detector is developed using GPU implementation of the Yolo version 3. Yolo is an extremely fast real time object detection model. It gives the best balance between the accuracy and the speed. Yolo stands for "You look only once". It applies one neural network to entire frame/image and come up with bounding boxes with the probabilities for each of them. Logistic regression is used to come up with the probabilities for detections and each bounding box is weighted by the associated probabilities. openCV is used to preprocess

every frame before passing in to the neural network. Deep neural network of openCV does not support nvidia GPU so the neural network module of Pytorch is used to implement yolo v3 in a nvidia GPU [14][15][16]. Yolo is a general purpose object detector with 80 classes but the tracking algorithm only requires to detect person class. So the other classes are filtered out and bounding boxes are only drawn for person detections.

After drawing bounding boxes in the frame it will be used to determine the movements of the detections. Before determining the movements, detections are filtered out again using re-identification module leaving only the lecturer's bounding box in the frame. Position of that bounding box is kept as a reference to following frames. By analyzing 10 such frames, movement of the bounding box is determined with the direction of the movement. 10 frames are used to minimize the noise occurred due to small movements of the bounding box. Those measures will be used by the camera control module to generate control commands to the camera.

2 Person Re-Identification Module

This module will handle the multiple person detections and keeps tracking only the lecturer. Deep sort, an extension of sort algorithm is used as a base to identify multiple detections and to calculate the similarity scores of them [18]. At the start of the lecture, only the lecturer must be there in the camera view. His object id will be 1. After starting the lecture, tracking algorithm will only consider the movements of bounding box with object id 1. If the detection is lost at some stage of the tracking process, then the camera will first zoom out to see if the lecturer is visible in the full zoom out view. If lecturer is unavailable in full zoom out view, then camera return to a predefined home position where the lecturer will re appear again. Then the same tracking process will continue from the beginning by issuing object id as 1 to the lecturer again.

3 Camera Controller

This module will control the PTZ camera based on the detections produced by above two components. Tracking module will store the centroid coordinates of bounding box in a numpy array. Based on the last 10 coordinates, movements and the direction of the movements are generated using the tracking module. Based on those data, camera control commands are generated and send to the camera. VISCA protocol [17] is used to send control signals to the camera over a socket connection. Camera is connected to the image processing server via an Ethernet cable for sending control signals. USB connection from the same camera is used to obtain the video feed of the camera.

If the bounding box is stable for a pre-defined time without any movement, camera zoom function is triggered and the bounding box is zoomed in. when he starts moving again camera will return to initial zoom and keep tracking the lecturer again. Camera will be moved in all x(pan), y(tilt) and z(zoom) axis. Speed of the camera controls will be dynamic according to lecturer movements.

Student Questions Handling & Whiteboard

Main purpose of this component is to create live interaction with the lecturer and the student. This component developed considering two main parts.

1. Student Questioning Handling with Whiteboard

Whiteboard was implemented using vectors. So only the line coordinates will be transferred among users and created a vector based whiteboard. When a line draws, the coordinates of that drawing goes to the server. Then through the server socket connection that line coordinates are streaming to the all users of the current session. As this whiteboard is a light weight board run with vectors, small process will be used not like video based whiteboards. And also lecturer and the student both can use and draw with this whiteboard as a real time whiteboard.

2. Live chat with poll

The main purpose of this module is to allow chat with each other who connected to the same session. Whether students can ask questions from the lecturer using chat or chat among other users is allowed. Users also can share images or pdf documents among others using the chat window.

Rather than this there is a specific feature that allows lecturer to create polls. So the poll will be displayed to students and they will be able to pick the preferred answer. After the vote is done results will be analysis in the server according to the poll id and lecturer will able to see the results in a graphical way. For the graph drawings plotlyJS is used as a library. Finally, the vote graph can be shared among students. So the students also able to see the results in graphical way.

Both whiteboard and the chat with live poll implemented using socket connection. For publishers and listeners one socket is created each and will be adding the necessary properties for the socket in server. Here the values of the properties of the socket given as pingInterval: 65000ms and pingTimeout: 60000ms on the front end. So it will give a more persistence connection that suitable for the sockets.

Drawings in the whiteboard streams with minimum or no delay. Mostly it depends with the internet connection, because socket has no or very low latency.

All these drawings and chats will be saved in a database for the usage of offline playback.

Commercialization aspects

Key stake holders of this system are:

- University / Educational Institutes / Schools
- Lecturers / Teachers
- Students

Our main goal of the product in the aspect of commercialization is reduce the cost for the usage of the product than the competitors in the current market.

Real-time e-learning platform is commercialized based on two business revenue models.

- Cloud based revenue model
- On premises revenue model

Cloud based revenue model

This revenue model is used if the client agrees to install their application setup in an amazon cloud environment. Amazon will charge for monthly resource usage. Clients are charged with an additional service charge which will be a multiple of the entire usage bill from cloud service.

Total usage bill = Amazon cloud bill x billing factor

On premises revenue model

If the client does not want to use a 3rd party service to host their application and wants to setup the system on premises, this revenue model will be used. Here client will be billed monthly based on the user accounts created in the system.

Testing and implementation

From the starting point of the project, we followed a waterfall model of development until the end of the project. Unit testing was done to each individual component developed while integration testing also carried out each time a feature integration was carried out to the system. We maintained a development environment which was our local devices which we use to develop as well as a AWS Linux server as the production environment. Each component was developed and tested in both of these environments for expected outcomes.

During the implementation period, we used GitHub for the version controlling of our application. Two Git repositories were maintained as for the frontend developments and backend developments. Clearly identified features and independent development

of them in different branches allowed the features to be integrated to the main system successfully without major merge conflicts.

After a successful development stage, following are the outcomes of above implementations in the UI point of view.

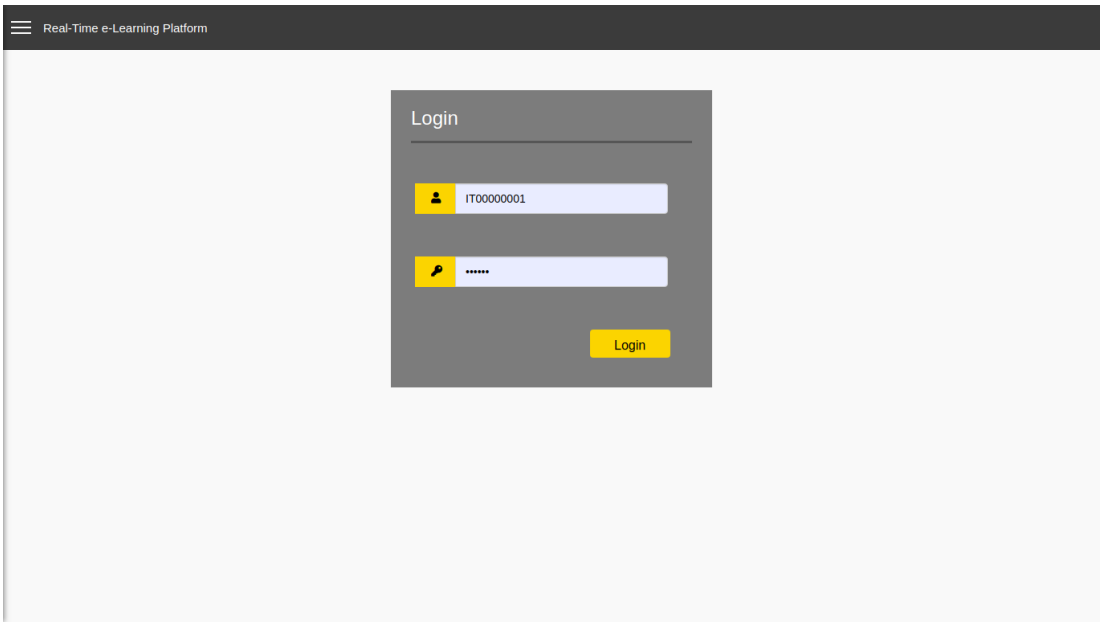

Figure 2.3.1. Login screen

All the registered users should log into the system through the login page shown in the figure 2.3.1. A default admin account is created on the installation and this account is used to setup other user accounts as required by the institute.

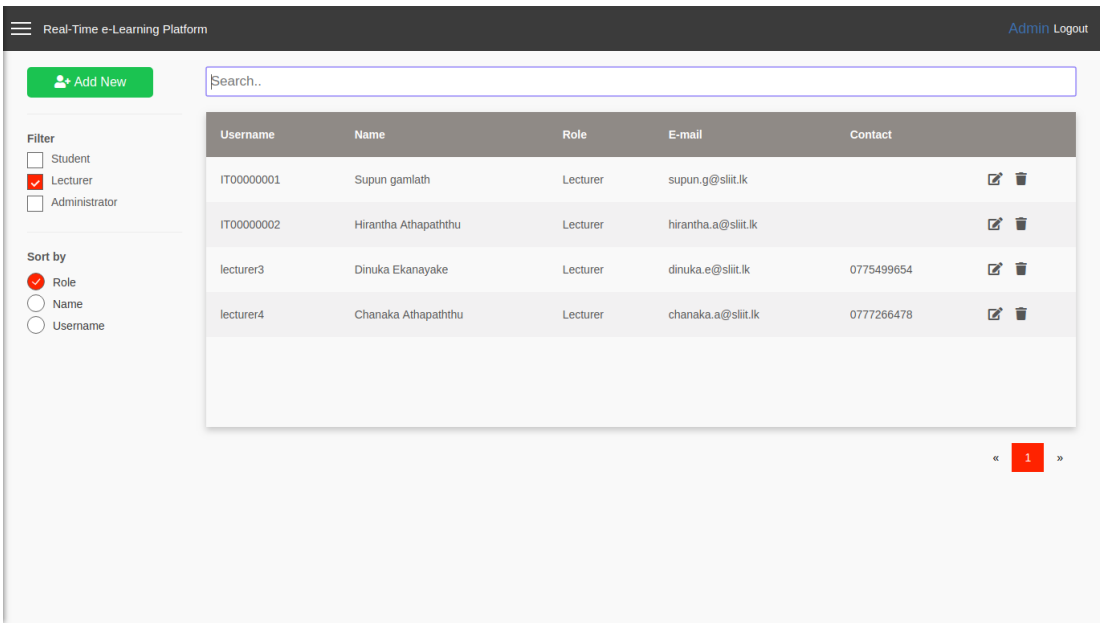

Figure 2.3.2. View users

Once logged in using admin account, new user accounts can be created with the Users dashboard UI shown in the figure 2.3.2. From this UI, admins can search for the current registered users with their username or name. Filtering using role and sorting also available.

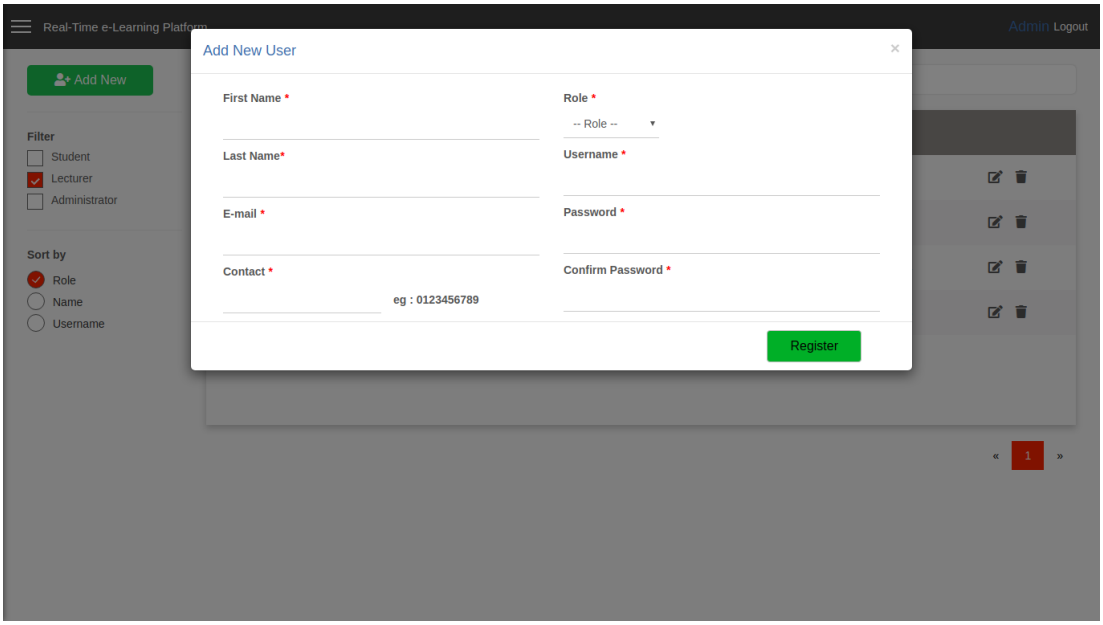

Figure 2.3.3. Add new user

New users can be added using the Add New User UI shown in figure 2.3.3.

Live Streaming

Once lecturer's logged into the system, they can navigate to the Live stream page which is shown in Figure 2.3.1.1 Live Streaming UI for lecturer which is a very simple interface to work with. In order to start a new stream, lecturer has to enter the Module name, title and the venue and click Start Streaming button.

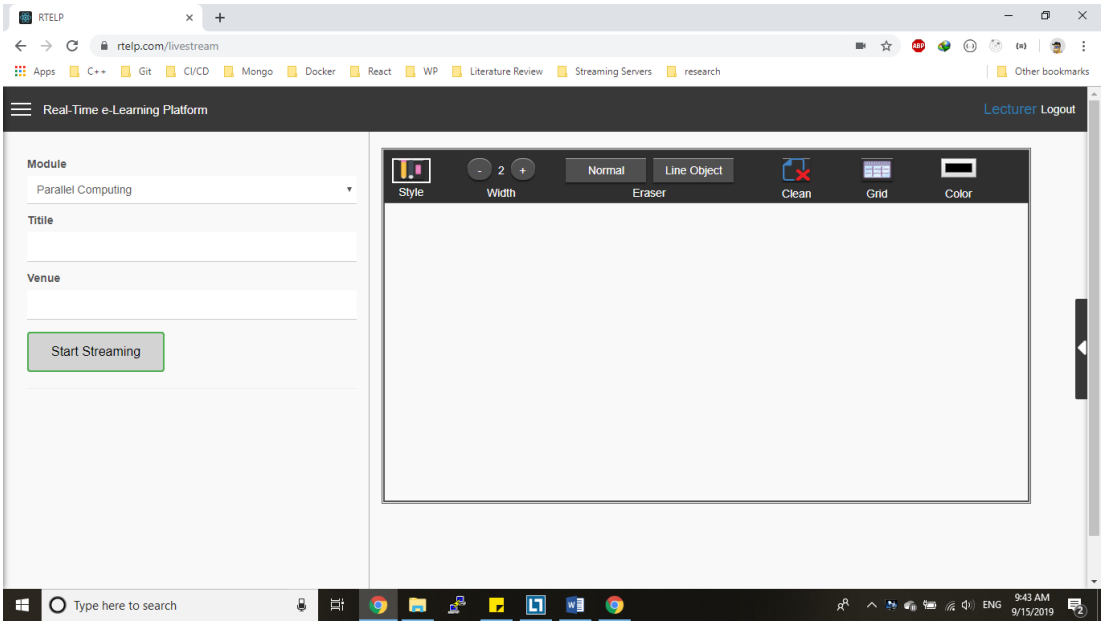

Figure 2.3.1.1 Live Streaming UI for lecturer

Once the start stream button is clicked, a menu will be shown to select which screen or application to be streamed. Figure 2.3.1.2 Screen Selection UI for Sharing screen shows the screens that lecturer can select during this process. If the lecturer PC is extended to multiple screens, those will be listed here. In this way lecturer can stream selected screen only while keeping rest of the screens for other uses.

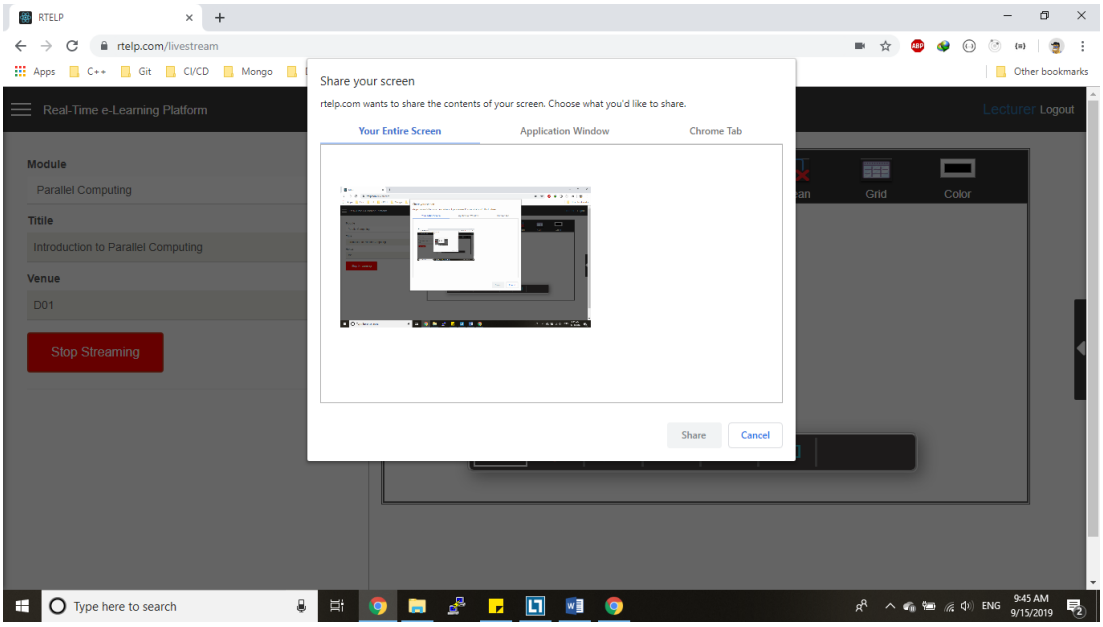

Figure 2.3.1.2 Screen Selection UI for Sharing screen

In the same window, it provides to select an application window to be capture and streamed (Figure 2.3.1.3 Application Selection UI for Screen Sharing). This option also maximizes the privacy of the lecturer during a live stream since only that particular application window will be captured.

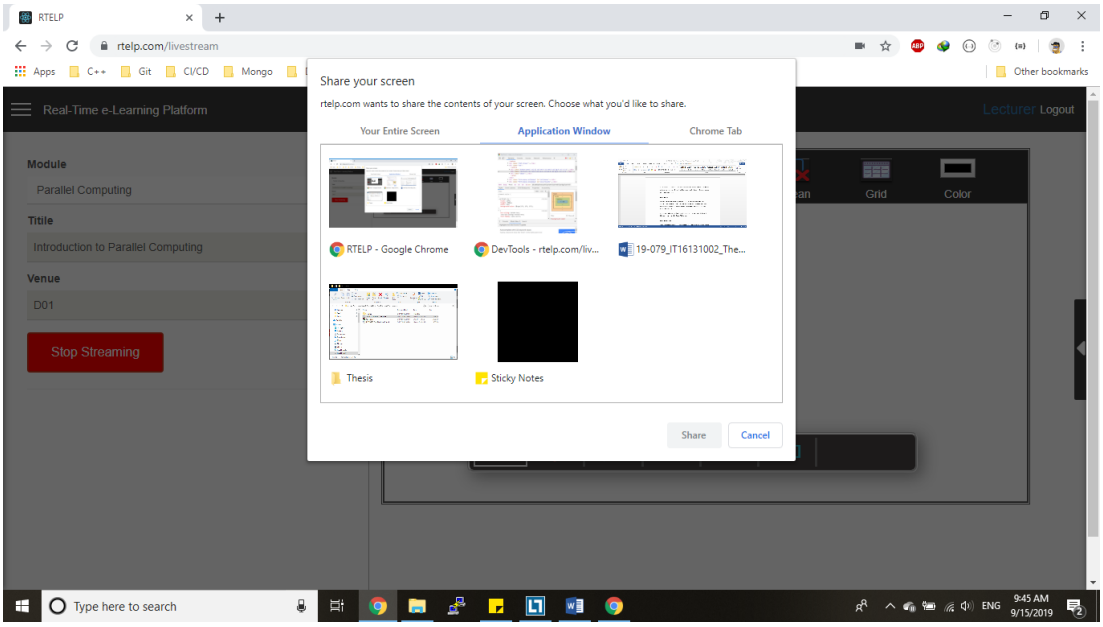

Figure 2.3.1.3 Application Selection UI for Screen Sharing

Once the screen to share is selected, system creates a new room for the session where students can join. Once streaming is ready, a preview of what lecturer currently share to students is displayed as shown in Figure 2.3.1.4 Preview of what lecturer shares on an ongoing stream4.

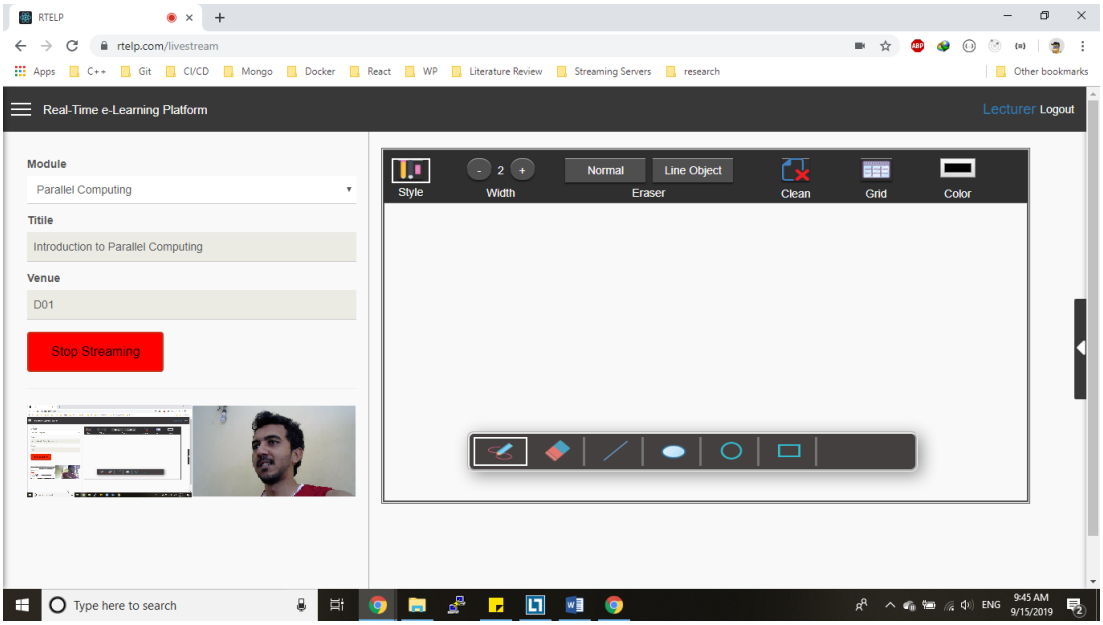

Figure 2.3.1.4 Preview of what lecturer shares on an ongoing stream

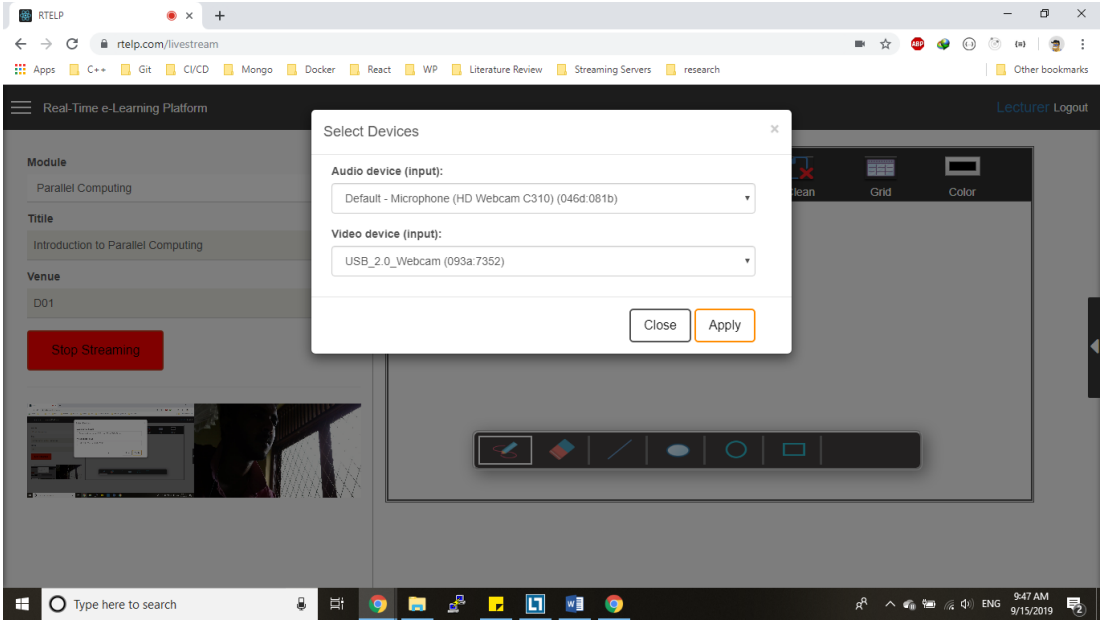

Figure 2.3.1.5 Input Device Selection Window

Lecturer can switch between streaming screens as well as cameras even during the stream which gives more flexibility and control to the lecturer about what they want to share.

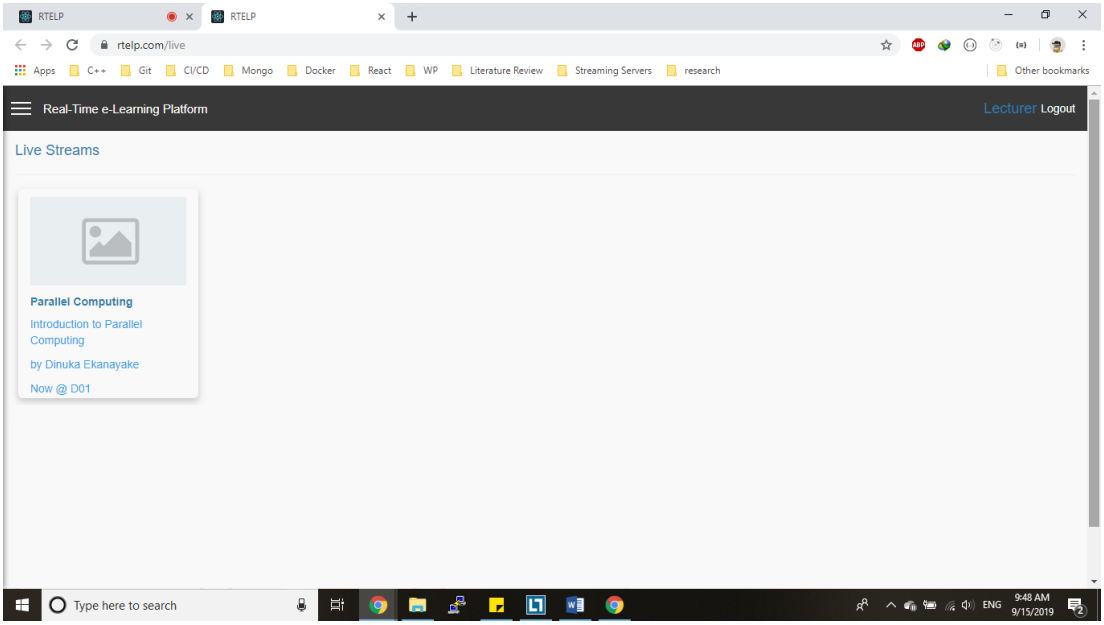

Figure 2.3.1.6 UI for Current Live Streams Available to Join

After a new live stream initiated, it will be shown in Live Now section as a thumbnail which contains the URL required to join the session. Once clicked, it will redirect student to the relevant session and streams will be loaded as shown in Figure 2.3.1.7 Student's view of a live stream.

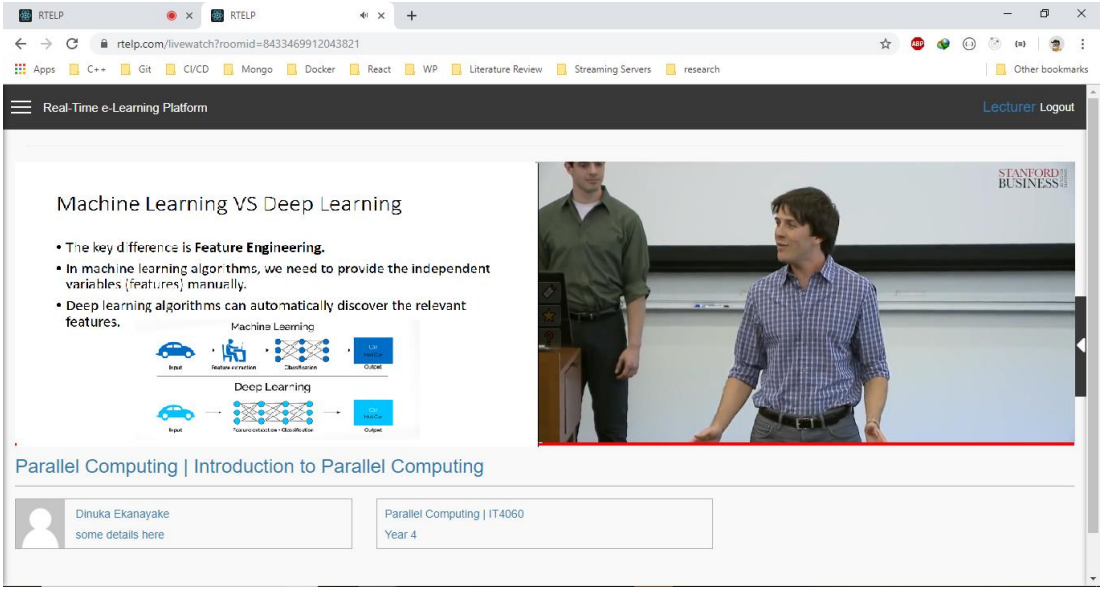

Figure 2.3.1.7 Student's view of a live stream

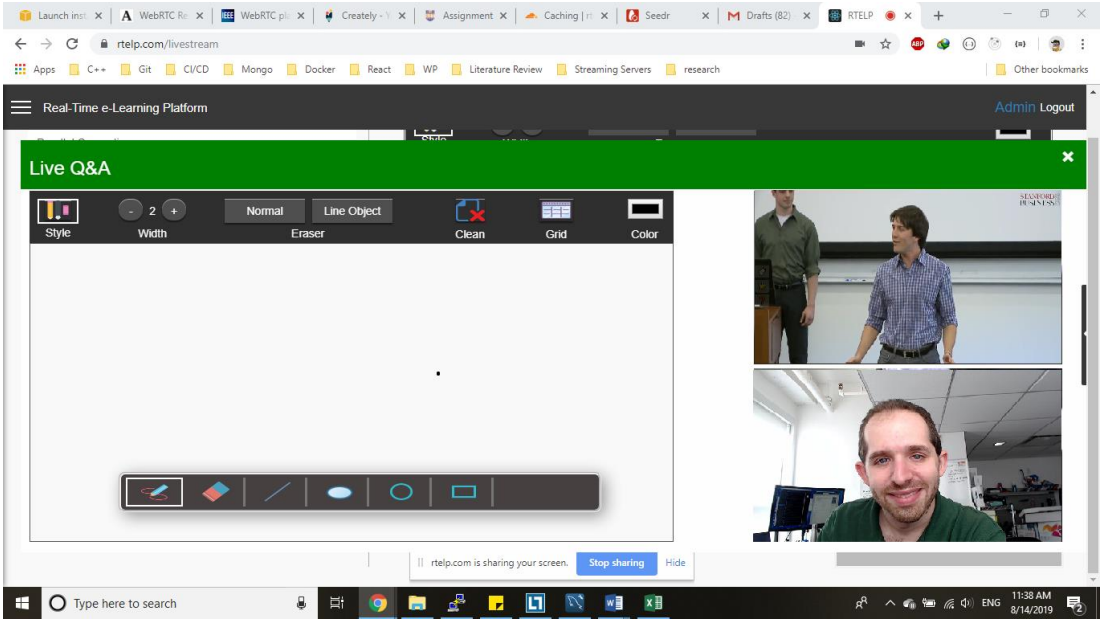

Figure 2.3.1.8 Live Q&A Sessions UI

Again when student is using live Q&A feature, WebRTC comes to play and start streaming students web camera to the lecturer. Whiteboard is placed alongside with the two camera previews of lecturer and the student. Both of the parties will be shown this window during a live Q&A. Student can use the whiteboard to explain and watch lecture explaining the solution to his or her problem in real-time during these live Q&A sessions.

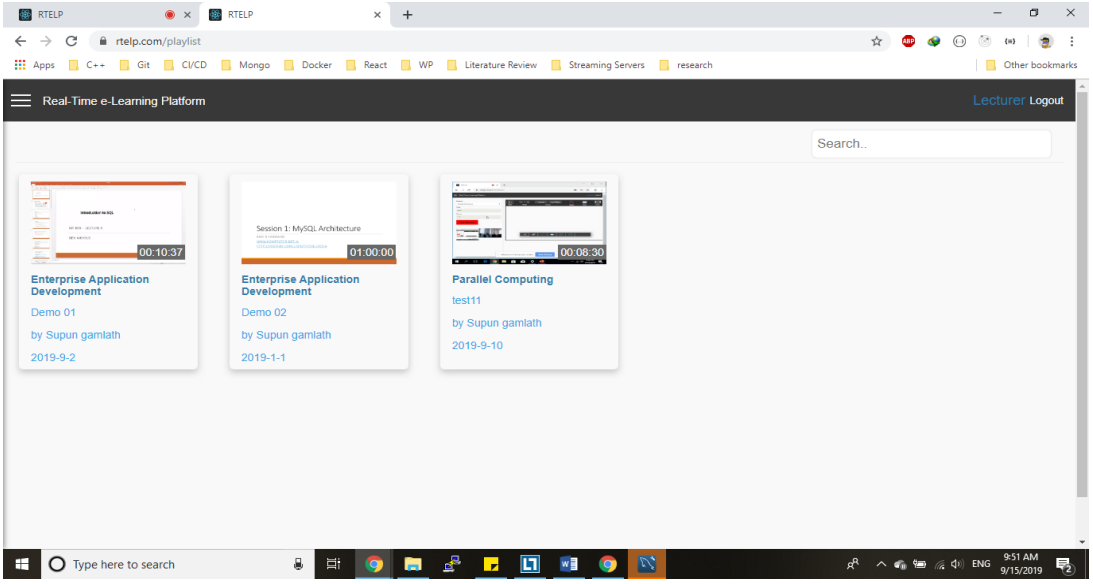

Figure 2.3.1.9 List of previously recorded lecture streams

Once the lecturer ends the session by clicking on Stop Streaming button, a background process will starts to process the recorded videos and will be published to the Recordings UI (Figure 2.3.1.9 List of previously recorded lecture streams9) once finished.

Those are the major UI implementations related to Live streaming component.

Student Questioning with Whiteboard

Current SQHW component UI's are as follows,

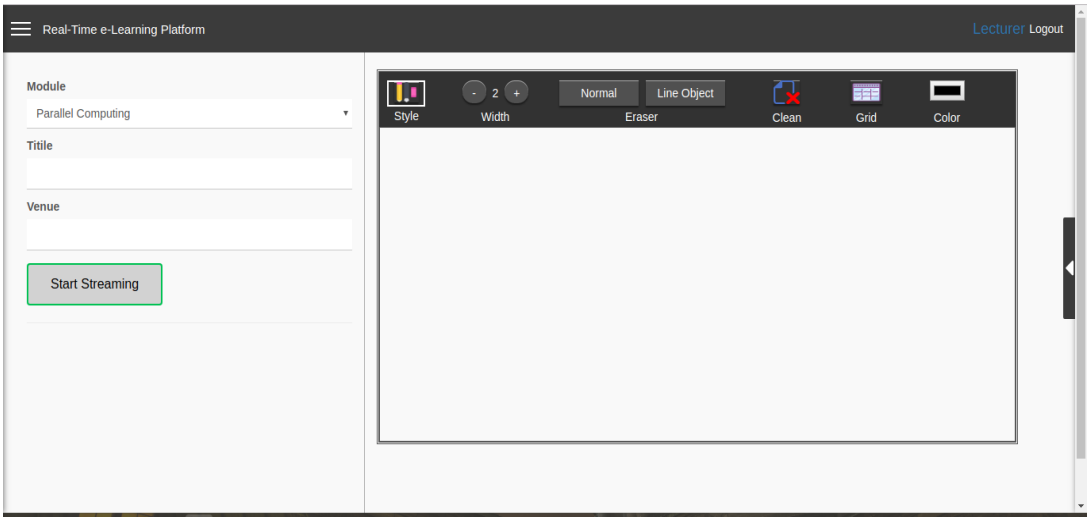

Figure 2.3.2.1: Lecturer Whiteboard UI

Above Figure 2.3.2.1 shows the current UI of the lecture's whiteboard. Here the whiteboard is fixed and display in a side. Any drawings by a student will be seen here.

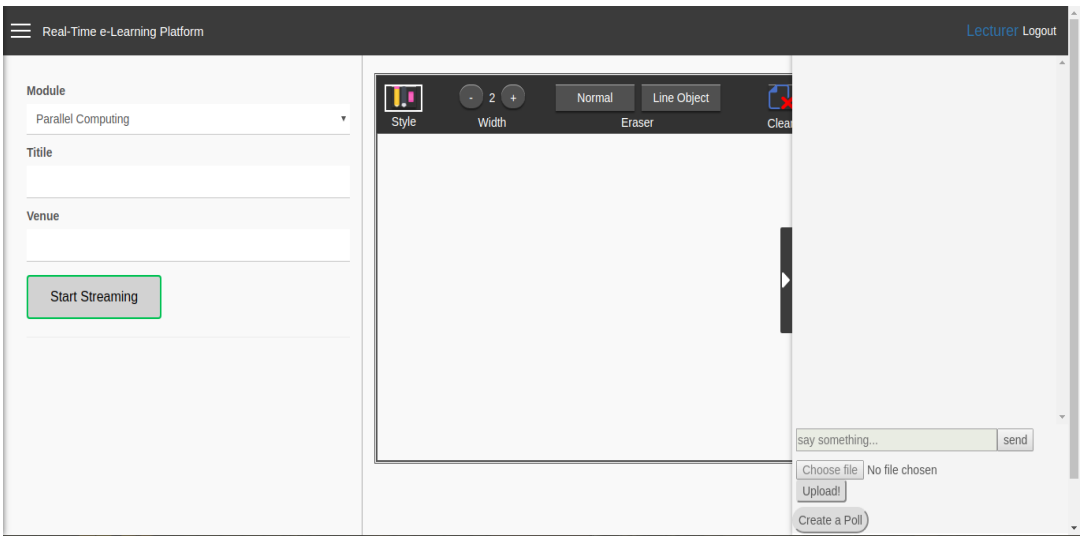

Figure 2.3.2.2: Lecturer Chat UI

In Figure 2.3.2.2 shows the chat box appearance. For the convenience of the lecturer it has been collapsed and whenever needed it can be expanding like in the figure. If a message comes there was a notification sound as well as blink in the expand button with a different color.

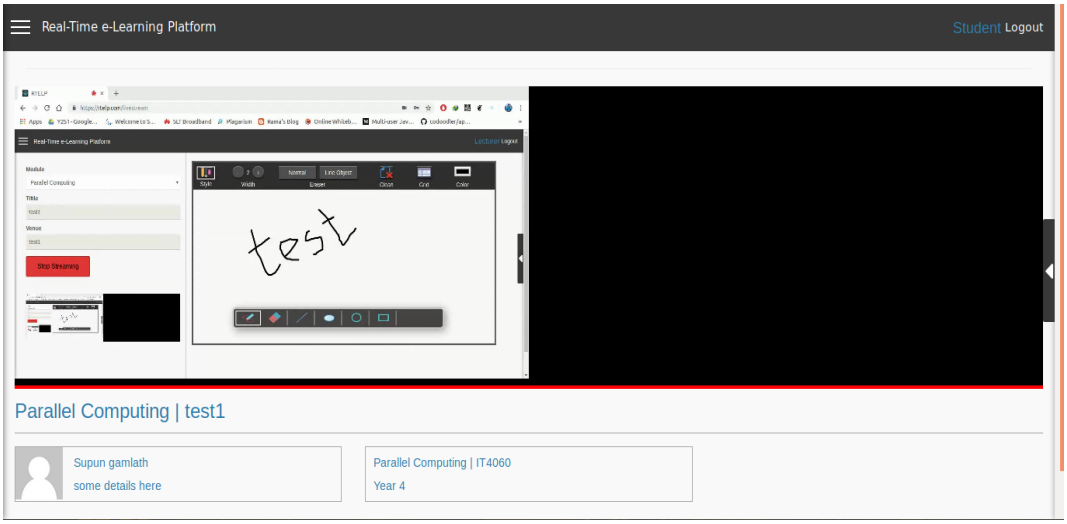

Figure 2.3.2.3: Student Live UI

In Figure 2.3.2.3 shows the screenshot of student side. At the beginning the whiteboard is not showing in the screen. If the student wants to see the whiteboard, he/she has to click the box sign at the beginning of the player controls. Lecturer screen video and the video feed from the camera appears on each sides.

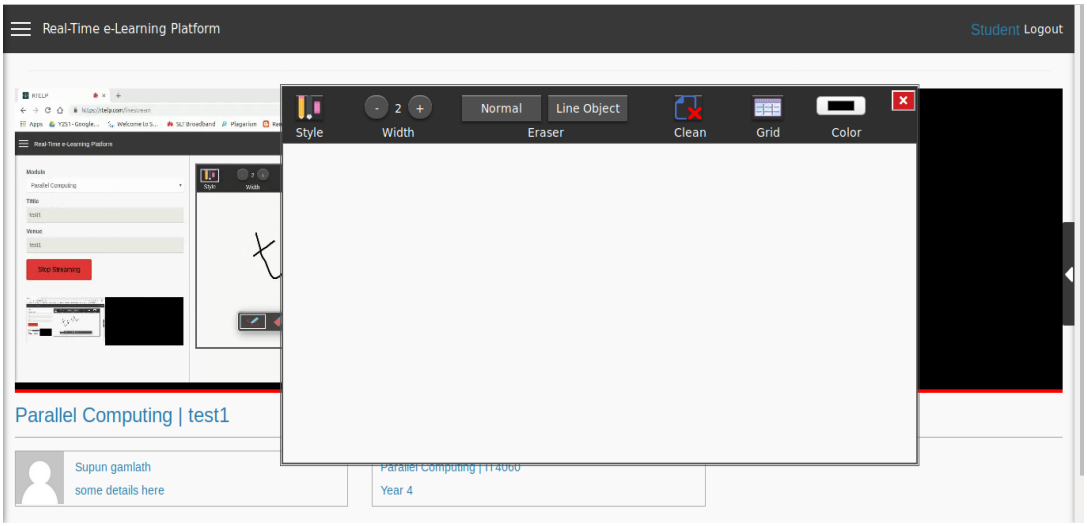

Figure 2.3.2.4: Student Whiteboard UI

In Figure 2.3.2.4 shows the whiteboard of the student side when it visible. Here in student whiteboard is drag gable. Student can move the whiteboard to anywhere as he/she wish. After using the whiteboard, student can hide it and continue watching the video feeds.

Student chat also same as the lecturer chat in Figure 2.3.2.2.

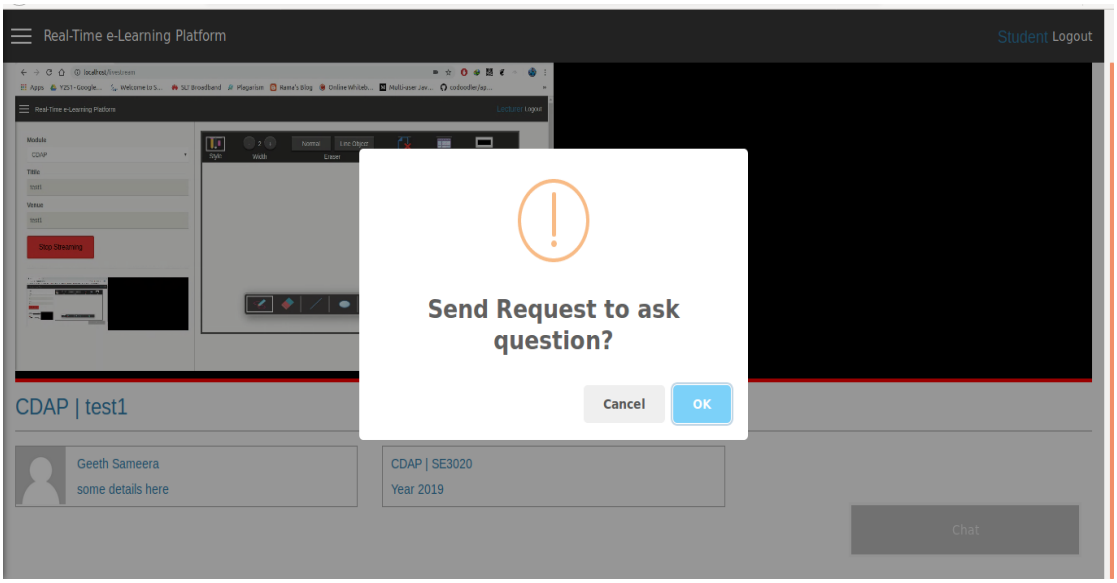

Figure 2.3.2.5: Raise hand send request

When student wants to ask a question, first he has to send the request using Raise Hand method. When student clicks the button to ask a question confirmation box is popup and with the confirmation of that request will be send to the corresponding lecturer. (Figure 2.3.2.5)

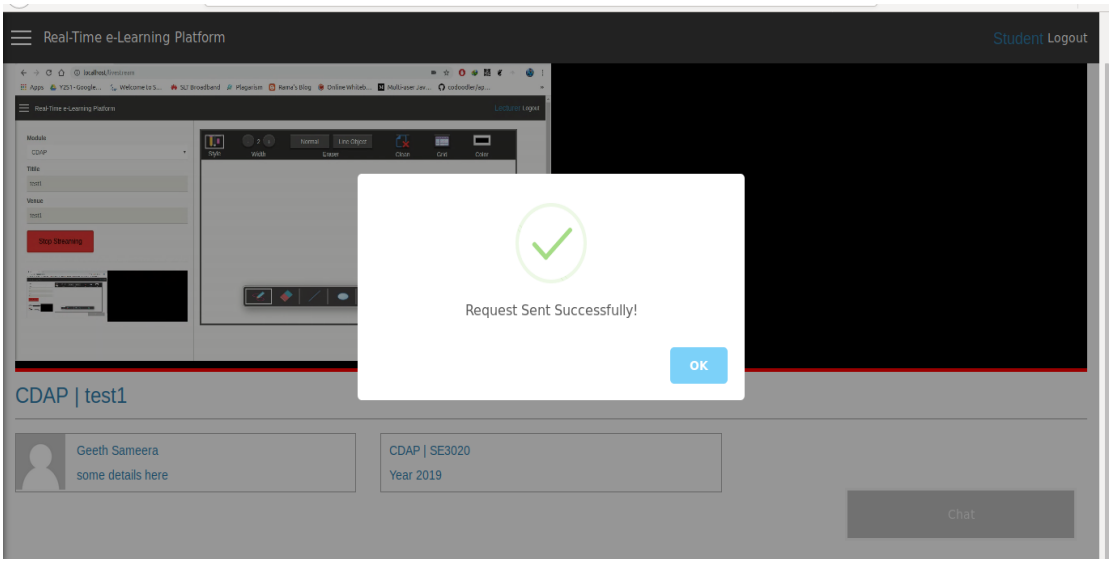

Figure 2.3.2.6: Raise hand successful

Figure 2.3.2.6 shows the successful request sent message. If there is something wrong or request doesn't send to the lecturer, sending failed message will be popup.

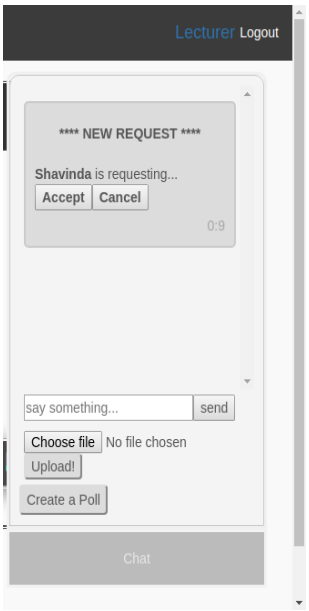

Figure 2.3.2.7: Raise hand lecturer view

The request from the student to ask a question is appearing on the lecturer side chat box (Figure 2.3.2.7). Lecturer can confirm of reject the request.

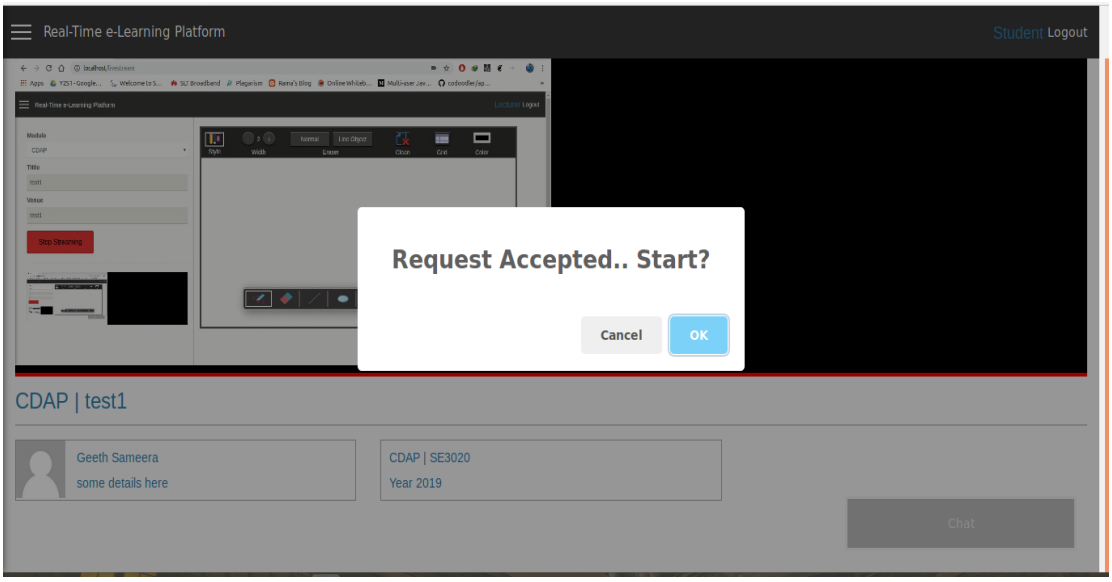

Figure 2.3.2.8: Raise hand student starts Q&A

After the confirmation of the lecturer popup box appears on the corresponding side and student can start the questioning when he clicks ‘ok’ button. (Figure 2.3.2.8)

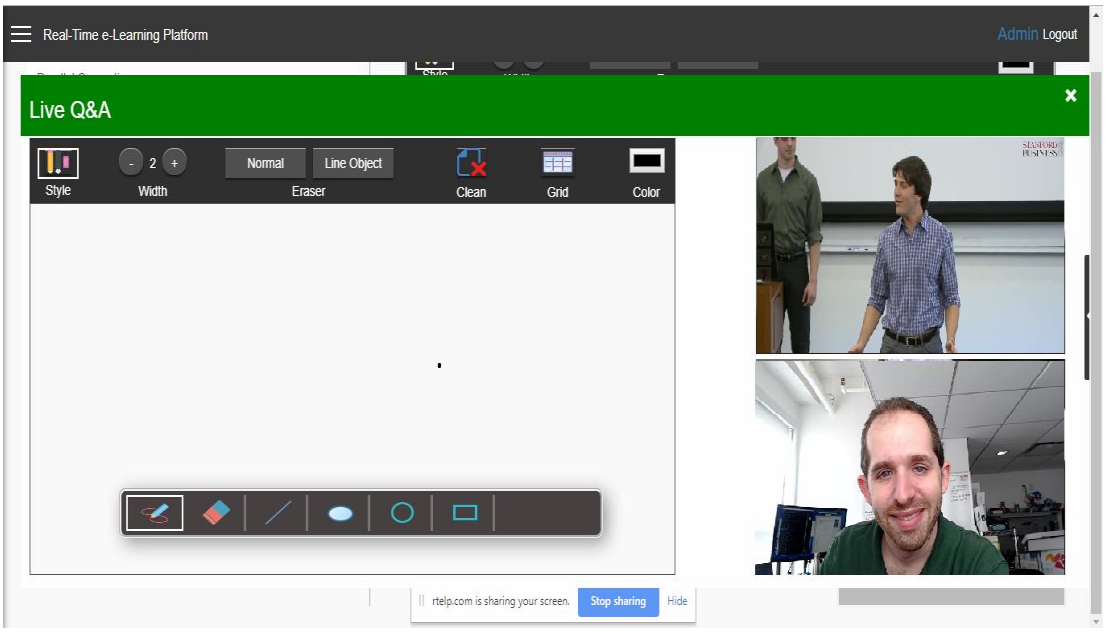

Figure 2.3.2.9: Live Questioning

After starting the questioning by the student lecturer and students both popup a window and through that live questioning can be done. (Figure 2.3.2.9)

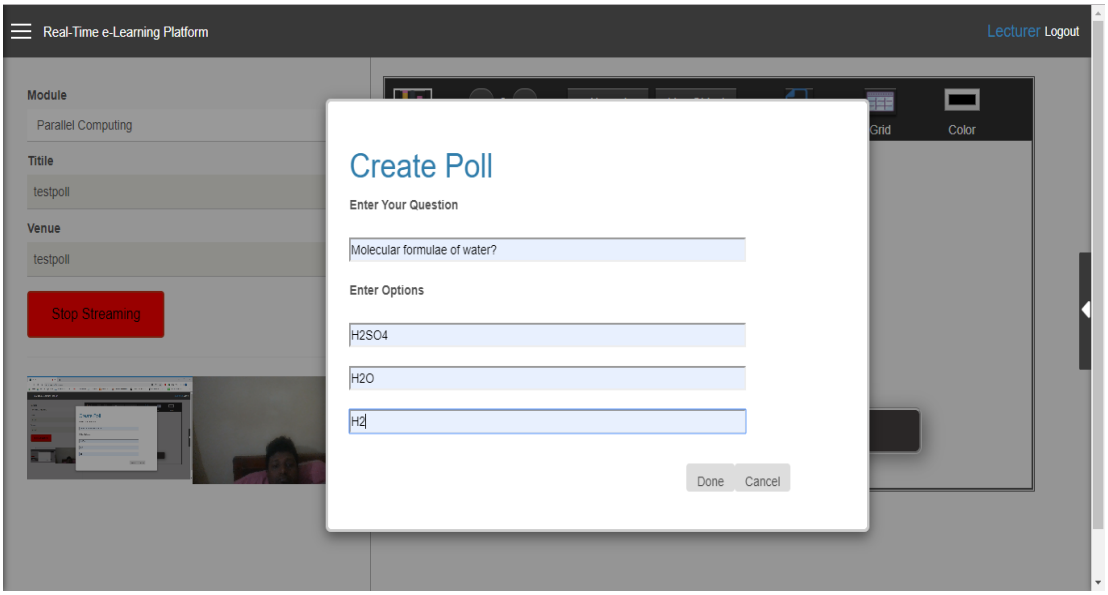

Figure 2.3.2.10: Create Poll

Figure 2.3.2.10 shows the creating of a poll by the lecturer. After creating the poll question and the options were transmitted to the students of the respective session.

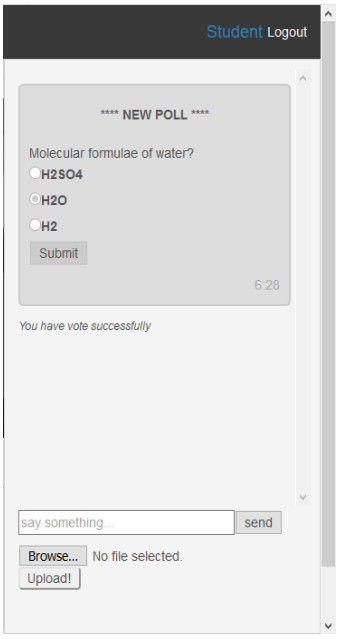

Figure 2.3.2.11: Student Poll

After lecturer start the poll, question and the options will appear on the chat box of the students (Figure 2.3.2.11). Students can vote with their preferred answer here.

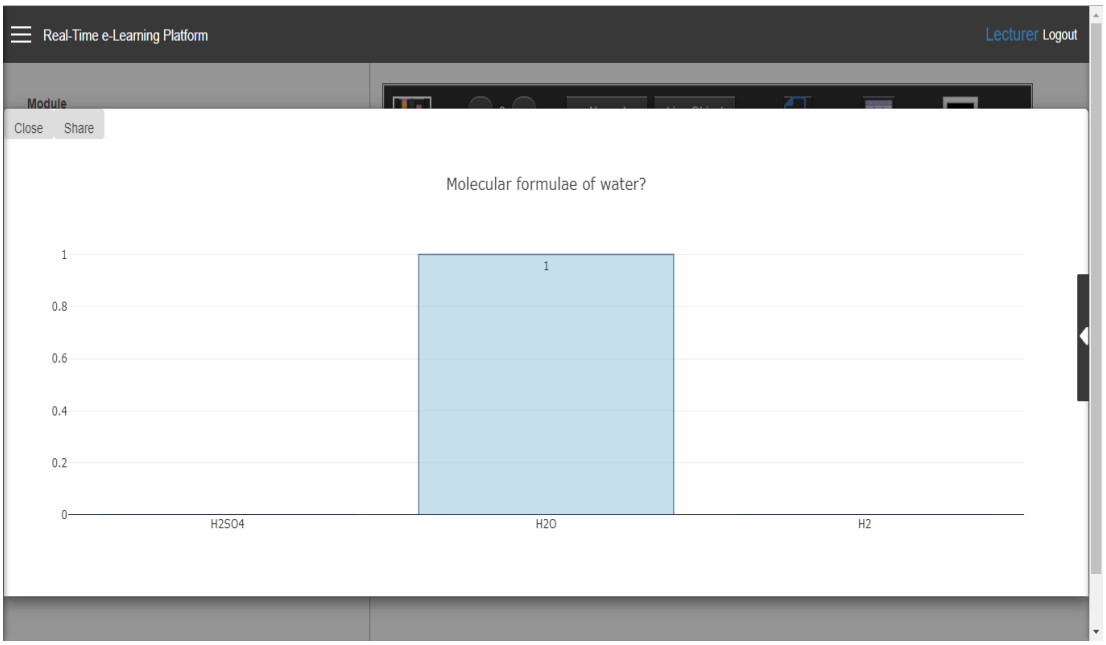

Figure 2.3.2.11: Poll Results

Lecturer can watch the poll results while students voting. Results are appearing in a graphical way. After students finished voting, Lecturer can share the results and then students also able to see the results. (Figure 2.3.2.11)

Offline Playback

Once the encoding process is completed for a recorded lecture, it will appear in the *Recordings* page accessed using side menu.

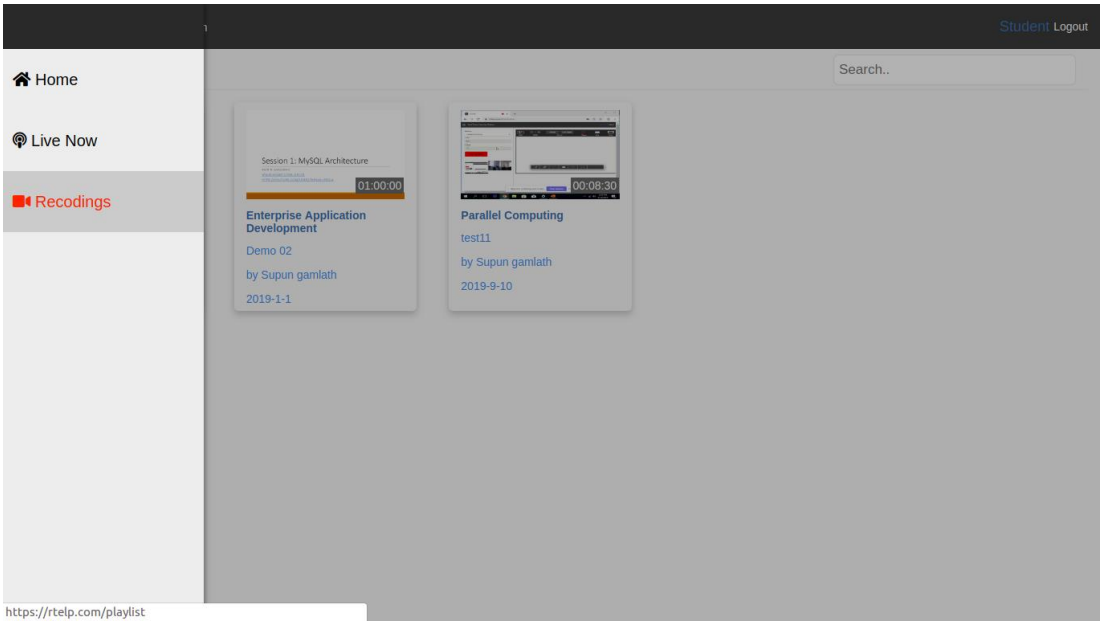

Figure 2.3.3.1 Recordings Menu

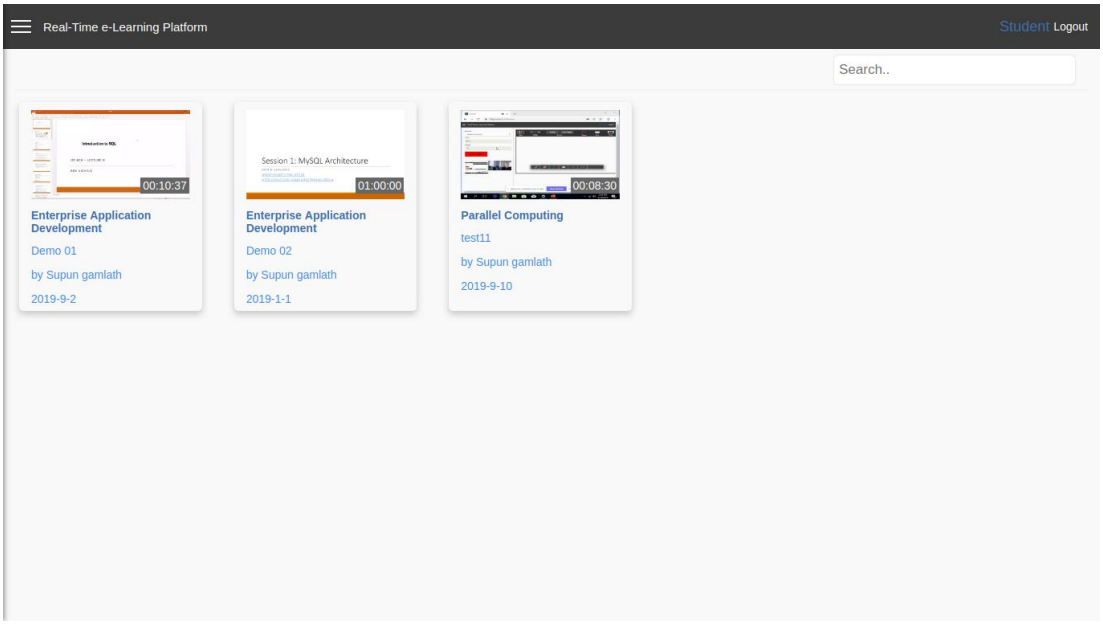

Figure 2.3.3.2 Recorded Lectures

Student can search for lectures using module name or lecture title. Once a lecture is selected, student will be redirected to the playback player.

Player will automatically start to play the lecture in the default 50:50 layout as shown in the figure 2.3.3.3.

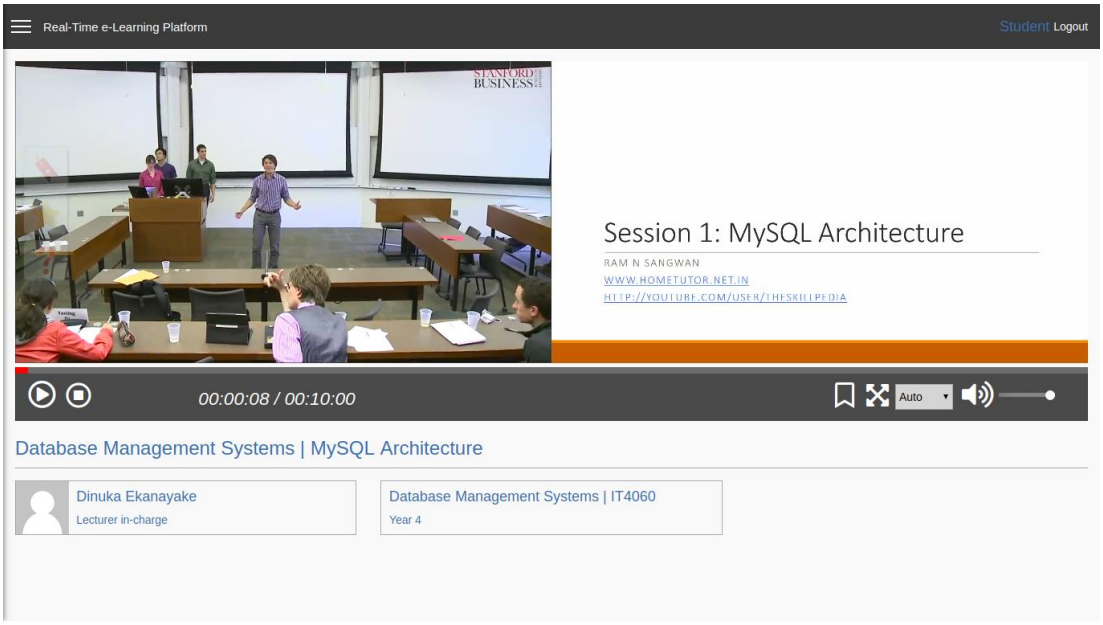

Figure 2.3.3.3 Playback player in default 50:50 layout

Student do have all the basic functionalities of a video player including *play, pause, stop, seek, volume up, volume down* and *mute.* Apart from the default 50:50 layout, there is a full screen option that can be enabled by double clicking on a video of choice or by clicking the *full screen* button in the controls bar.

Figure 2.3.3.4 and Figure 2.3.3.5 shows the full screen layout with 2 videos interchanged.

While one video is in full screen mode, other video is in a floating position. This floating video can be resized and moved around the screen.

To exit the full screen mode, student can either double click on the full screen video or by clicking the *full screen* button in the controls bar.

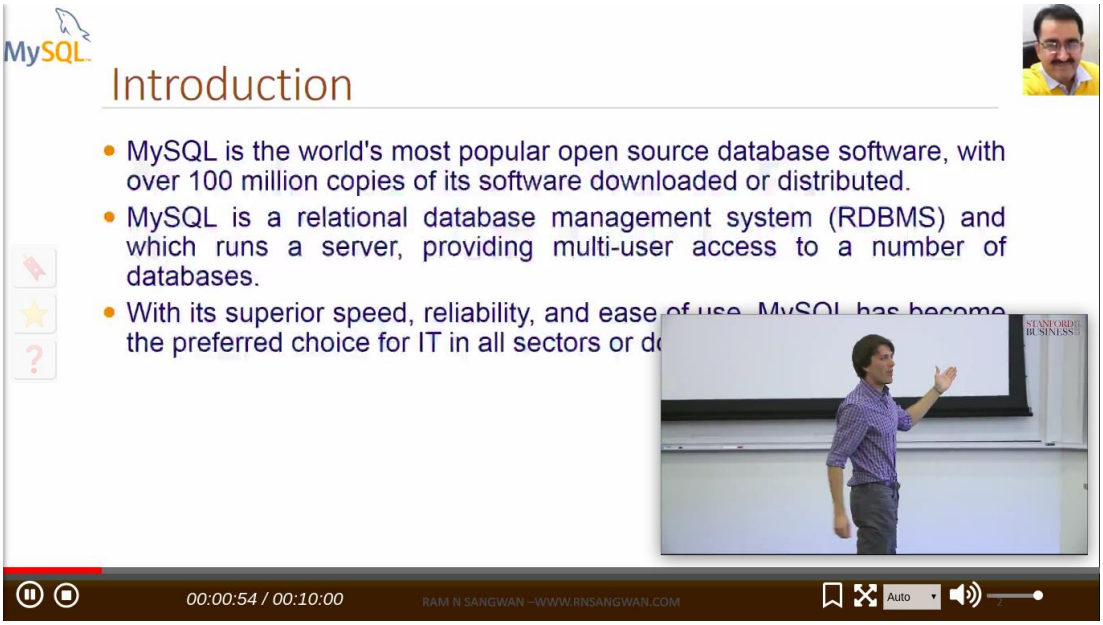

Figure 2.3.3.4 Playback player in full screen layout

Figure 2.3.3.5 Playback player in full screen layout

There is an option to change the quality of the video as shown in Figure 2.3.3.6. Available options are *High*, *Medium* and *Low*. The default value is set to *Auto*, so that the player will adapt to the available bandwidth of the internet connection and change the quality of the video accordingly.

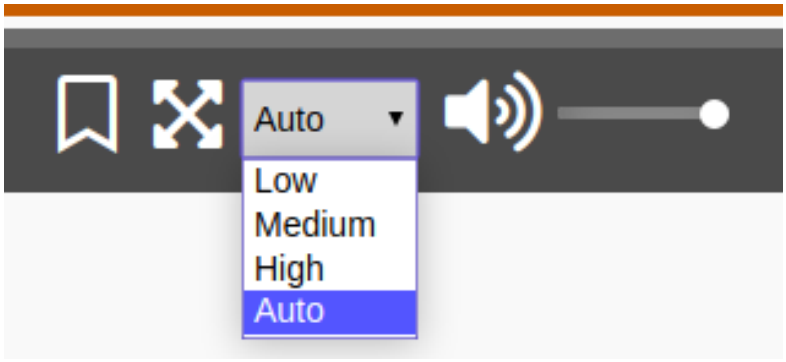

Figure 2.3.3.1 Quality Selector

Students can annotate the video with 3 different types of annotations. *Important, Unclear* and *Bookmark.* This can be done using a floating toolbar on top of the video that is made transparent to make less distracting to the viewer. It becomes opaque when the cursor is moved on top of it. This is shown in Figure 2.3.3.7.

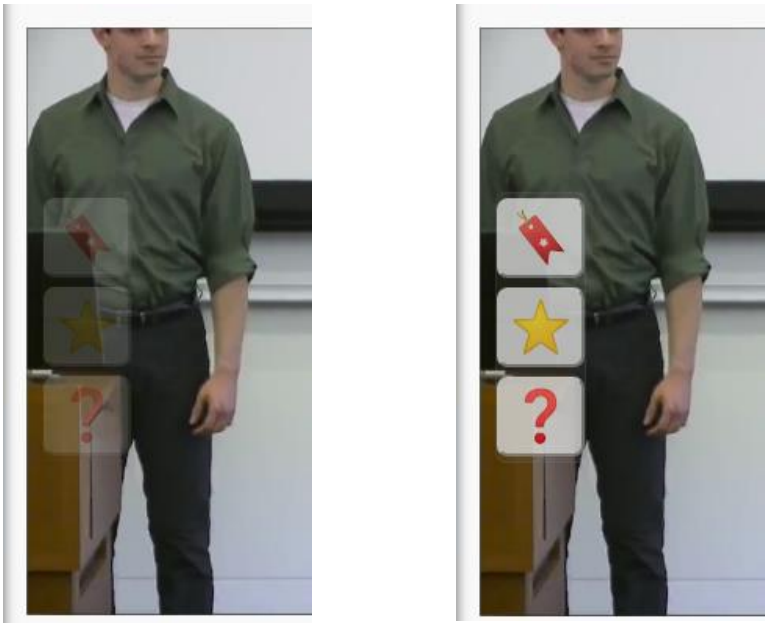

Figure 2.3.3.2 Annotation Buttons

These annotations are shown in a separate bar aligned with timeline. This bar can be toggled using the *Annotations* button in controls bar. This bar is shown in the figure 2.3.3.8.

Figure 2.3.3.3 Annotations bar

These annotations can be used to seek the video by clicking on any of it. It's timestamp and comment for the bookmark is also displayed as a tooltip on mouse hover. These annotations can be deleted by right clicking.

Lecturer Tracking

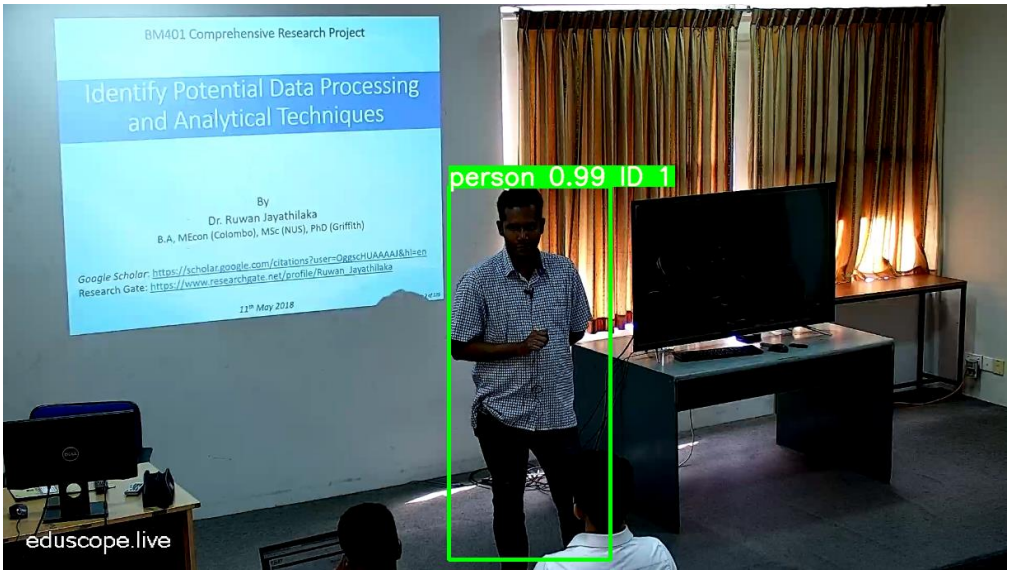

Figure 0.4 Tracking Sample 1

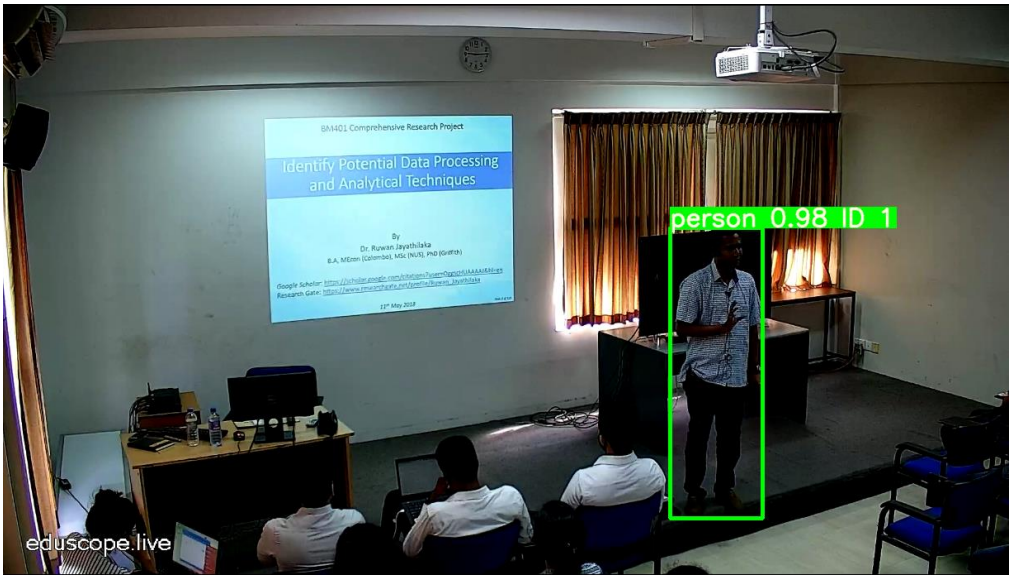

Figure 0.5 Tracking sample 2

3. RESULTS AND DISCUSSION

Online Streaming

We tested this process of the system using a AWS t2.micro instance (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only) running Ubuntu 18.04 LTS. System was capable of handling 10 simultaneous live stream sessions with 10 live viewers per each session only using 14% of the available CPU.

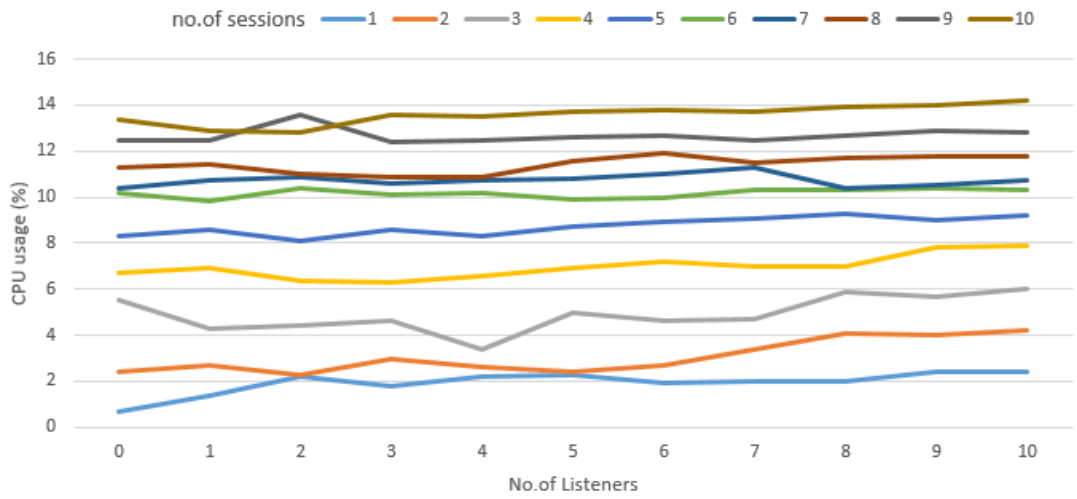

Figure 3.1 CPU consumption

Memory consumption for the same test scenario is given below.

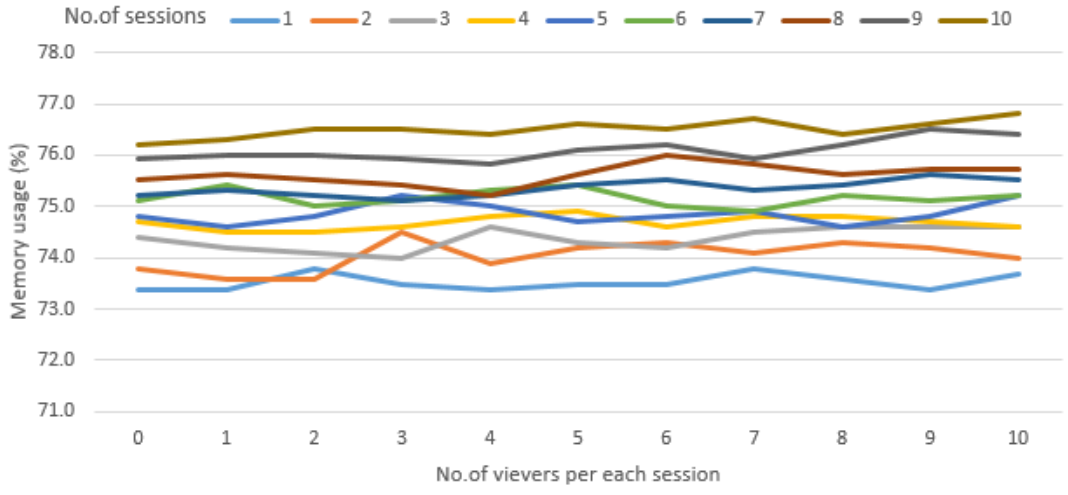

Figure 3.2 Memory Consumption

System consumed 3-4% of the available memory to cater the memory requirement of the platform for the above scenario.

In oder to provide such a reliable stream, the server should have a better network bandwidth as well. Graph shown in figure network consumption during above testing scenario. During this period 10 live sessions were running parallely and from them there were 20 streams tranfered in to the server. The maximum network bandwidth which has been consumed was 19.2MB/s.

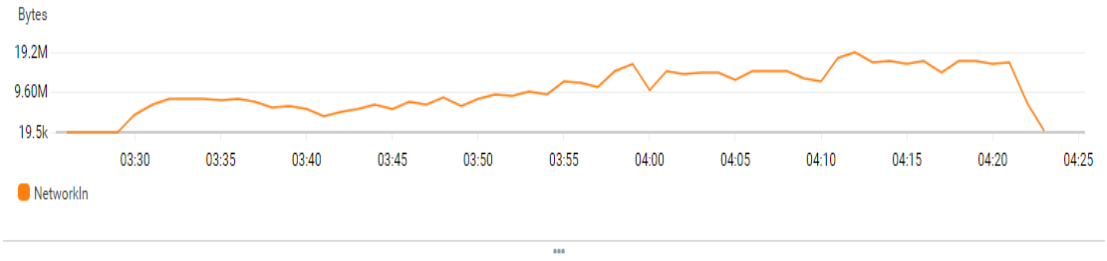

Figure 2.3. 1 Network In of AWS instance

Similarly, take a look at the Network out graph of the same instance for the same scenario. Average of around 14MB/s has been consumed as well as for the network out.

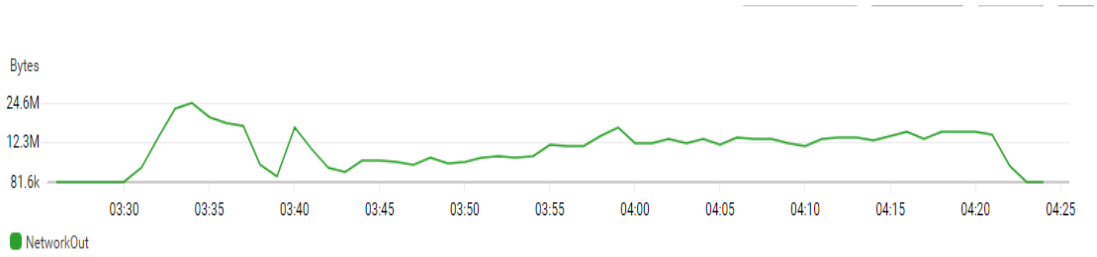

Figure 2.3. 2 Network Out of AWS instance

Those network graphs tend to grow rapidly with the increase of parallel sessions and connected student count as obvious. This is a must bear cost for the system since the application is totally based on internet.

A 60 minutes long video recording of 720p screen capture and 1080p camera takes approximately 600MB and 1000MB of storage respectively. Audio recording is 20MB per 60 minutes which makes the system to consume approximately 1620MB of storage per 1 hour long session to store the original streams. Cost for the storages also has to be bear by the client. Thses numbers are normal for such a quality video. But it is also possible to reduce the recording sizes by compromising some of the video quality as per the liking of the client.

A 60 minutes long video recording of 720p screen capture and 1080p camera takes 60MB and 200MB of storage respectively. Audio recording is 20MB per 60 minutes which makes the system to consume 280MB of storage per session to store the original streams.

Offline Playback

Encoding process was tested on a system with an Intel Core i5 CPU with 4 x 2.50GHz cores, 8GiB memory running 64-bit Ubuntu 18.04.2 LTS. Testing was done using 4 different input files with duration of 10 minutes. Each file was encoded to 1080p, 720p and 360p.

Time taken for each encoding process was compared with the input video's duration. All the encodings took less time than the original video duration. Figure 3.1 shows the comparison with encoding times as a percentage of the original video duration. 1080p encoding time is around 60% of the original video duration and 720p encoding time is around 33% of the original video duration. 360p video takes only around 19% of the original video duration.

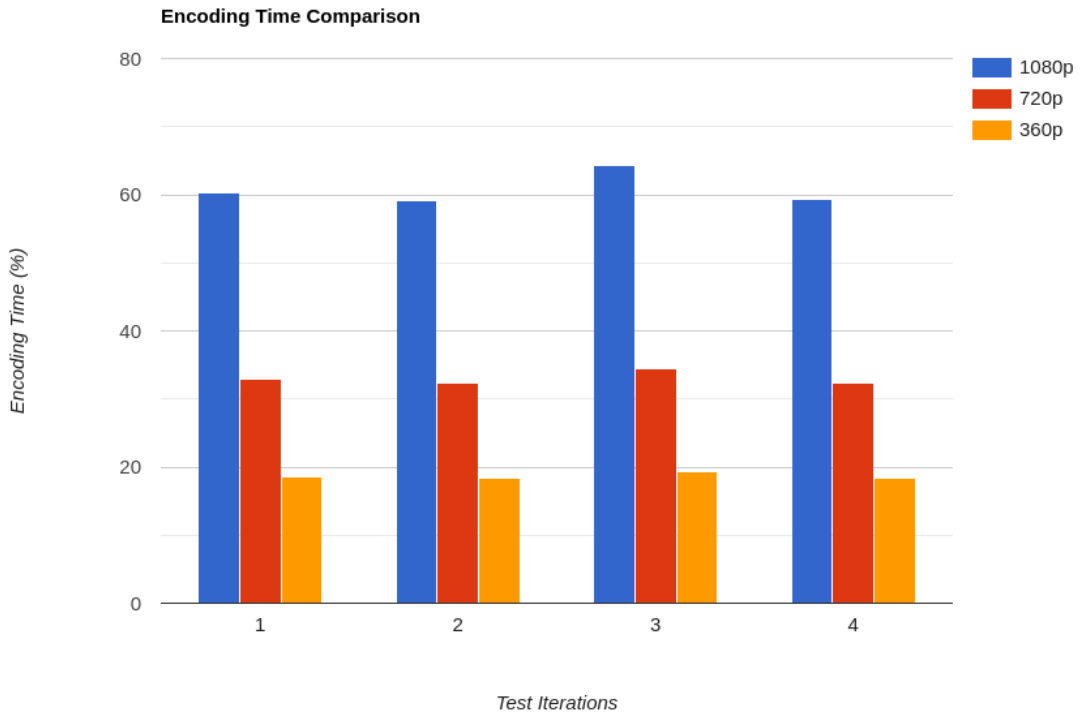

Figure 3.1: Encoding Time Comparison

The other aspect tested was the final file size of the encoded videos relative to the original file size. Figure 3.2 shows the comparison with each encoded file size as a percentage of the original file. 1080p output size is around 20% of the original file size and 720p output size is around 4% of the original file size. 360p video takes only around 1% of the original file size. However, this encoding process is very CPU intensive.

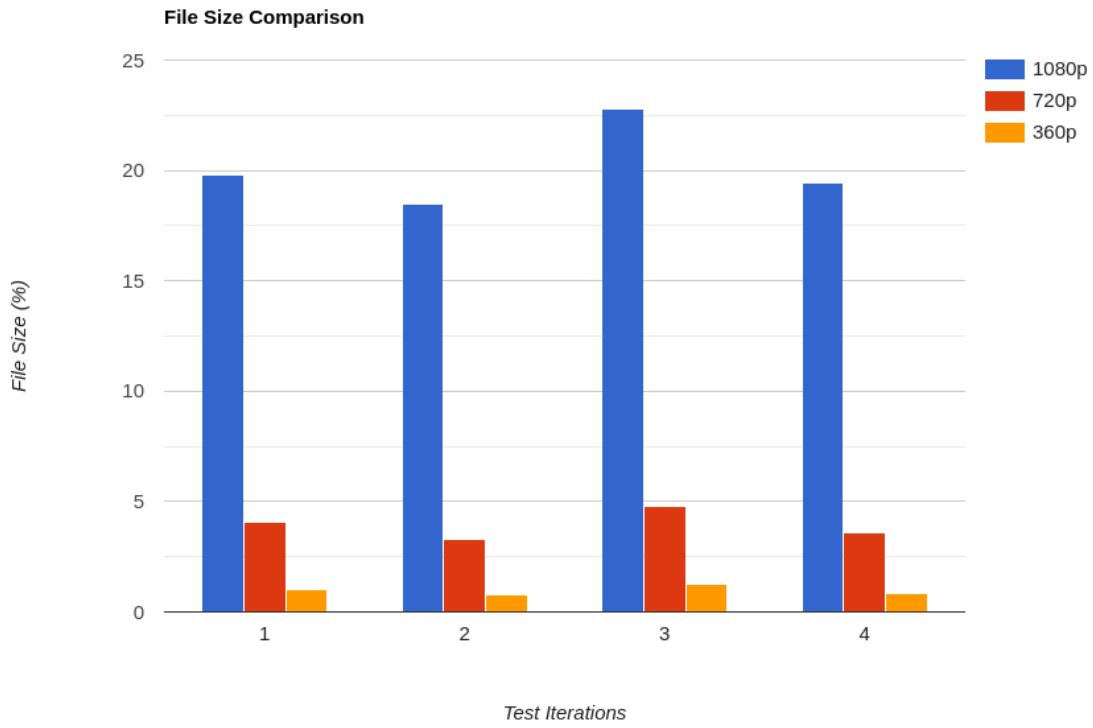

Figure 3.2: File Size Comparison

This reduction of file size is great for the users as they can play the videos with a low bandwidth internet connection.

Student Questioning and Whiteboard

Functionalities of the whiteboard and chat features were functioning as expected. Application is capable of communicating between lecturer's end and the students end of the application with minimum latency. Testing was done with different sessions to check whether drawings and chat were only communicating among the corresponding sessions only.

Lecturer Tracking

Tracking algorithm was tested on a core i5 3.7 GHz 8th generation CPU using 8GB Ram and in a GPU (NVidia GTX 1050) with the same CPU configuration. CPU performance was around 6-8 FPS. In GPU, FPS was around 20. GPU performance is sufficient to run the algorithm in real time.

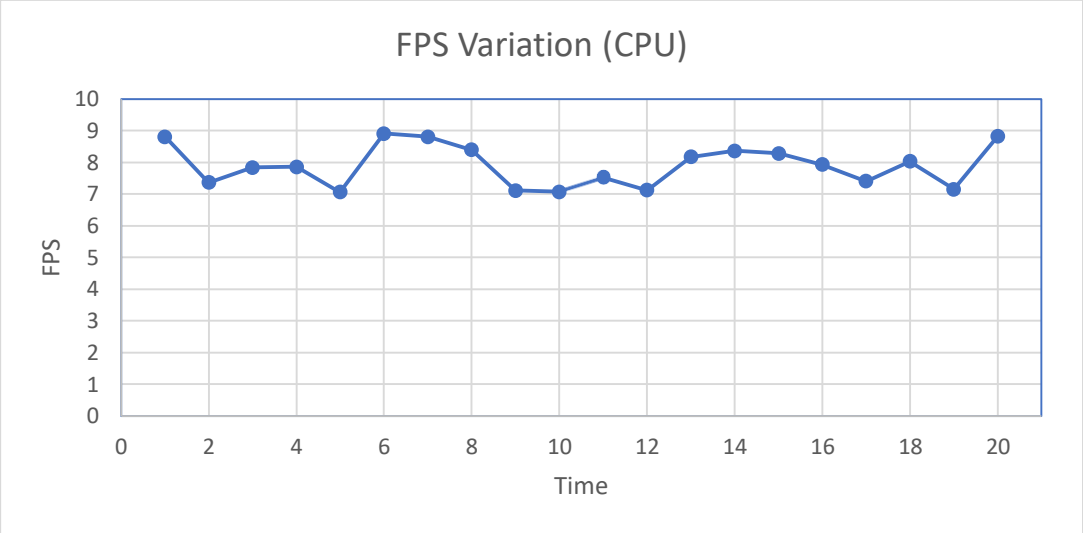

Figure 0.3 CPU Performance of the tracking algorithm

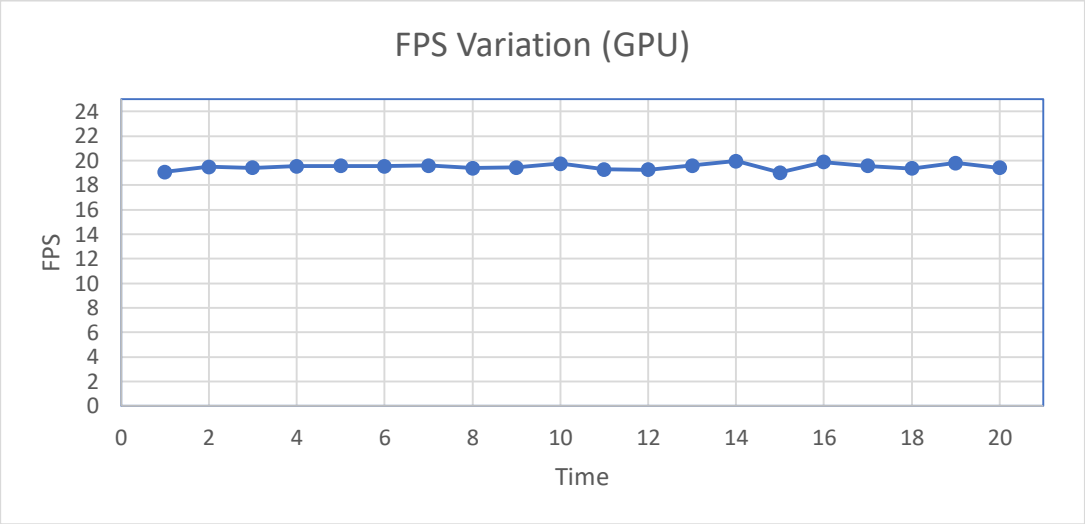

Figure 0.4 GPU Performance of the tracking algorithm

4. CONCLUSION

Implementing an actual real-time e-learning platform is not possible since anyway along the pipe line, we have to do some computational functions in order to deliver media across different clients. But we were able to achieve a near real-time solution for our problem reducing the latency to fractions of a second. All these results were possible because of using WebRTC as the base for the solution. For establishing near real-time remote communications, WebRTC is highly recommended than other technologies out there.

With video quality set to 360p, videos can be played without any buffering pauses given an internet connection with a minimum bandwidth of 128KB/s. With the quality set to *Auto*, videos will be played in highest quality available. If the prevailing bandwidth is not sufficient for the bitrate of the highest quality stream, videos will pause until player select the suitable stream. With a relatively consistent network bandwidth, videos will be played smoothly without any pauses in the middle. Both video streams are played in-sync with each other. If the videos get off-sync, player will automatically correct the sync by bringing back both videos to lower timestamp.

Each time a lecture is played, a list of time segments is stored in the server. These segments represent the parts of the lecture the user actually watched. The list has a reference to the original lecture session, student logged in, time the student logged in. Another list containing annotations added by the student for that particular lecture is also stored in the server with reference to student logged in and the original lecture session. Using these data, a detailed usage report can be generated.

Looking ahead, next steps for this solution will be introducing adaptive bitrate capability during a live stream. This will help the clients to access live streams at any condition of their network bandwidths.

Using automated PTZ camera improves student experience on e-learning platforms. yolo3 can continue tracking even in low light conditions with high accuracy. Deep sort and yolo3 both implemented using pytorch and it works well with a frame rate around 20. Camera controller smoothly drives the PTZ camera to obtain a good video feed of the lecturer.

Looking ahead, even though yolo3 works well in this scenario it is a general purpose object detector with a variety of 80 classes. Well optimized and specific purpose detection model can be trained using the previously recorded videos of the lecturer sessions by annotating frames as the image dataset to the model.

Yolo3 with deep sort requires a GPU to get higher FPS value. With the advancement of deep learning and the computing devices, if this algorithm can be more optimized in future versions, so that it can fit into a device like a raspberry pie, it will reduce the overall cost by tremendous amount.

5. REFERENCES

[1] S. Ouya, C. Seyed, A. B. Mbacke, G. Mendy and I. Niang, "WebRTC platform proposition as a support to the educational system of universities in a limited Internet connection context," 2015 5th World Congress on Information and Communication Technologies (WICT), Marrakech, 2015, pp. 47-52. doi: 10.1109/WICT.2015.7489643

[A] Janus.conf.meetecho.com. (2019). *Janus WebRTC Server: About Janus.* [online] Available at: https://janus.conf.meetecho.com/ [Accessed 6 Aug. 2019].

[2] Chris Curren, "*Strategies for e-Learning in Universities*",CSHE.7.04,2004

[3] H. M. Truong, "Integrating learning styles and adaptive e-learning system: Current developments, problems and opportunities," Computers in Human Behavior, 2015

[4] I U Cooray, W & M P Abhayawickrama, D & Ragel, Roshan. "Real time interactive lecture delivery system." 91-96. 10.1109/ICIAFS.2010.5715641.

[5] McCrohon M, Lo V, Dang J, Johnston C. "Video streaming of lectures via the Internet: An experience" ERIC Clearinghouse; 2001 Dec 9.

[8] D. Ehlers, "*Quality in e-Learning from a Learner's Perspective*", 2004,7

[1] Yueshi Shen, "*Live Video Transmuxing/Transcoding:FFmpeg vs TwitchTranscoder Part I "* https://blog.twitch.tv/live-video-transmuxing-transcoding-ffmpeg-vs-twitchtranscoder-part-i-489c1c125f28. [Accessed: 2019-05-13]

[2] Kurento, "Kurento Media Server Development Documentation" https://doc-kurento.readthedocs.io/en/6.10.0/#dev-docs [Accessed: 2019-05-13]

[13] Meetecho s.r.l, “Meetecho Janus Recordings”, https://janus.conf.meetecho.com/docs/recordings.html [Accessed: 2019-08-06]

[14] Rosebrock, A. (2019). YOLO object detection with OpenCV - PyImageSearch. [online] PyImageSearch. Available at: https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/ [Accessed 6 Aug. 2019].

[15] Medium. (2019). Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3. [online] Available at: https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088 [Accessed 6 Aug. 2019].

[16] Redmon, J. (2019). YOLO: Real-Time Object Detection. [online] Pjreddie.com. Available at: https://pjreddie.com/darknet/yolo/ [Accessed 6 Aug. 2019].

[17] En.wikipedia.org. (2019). VISCA Protocol. [online] Available at: https://en.wikipedia.org/wiki/VISCA_Protocol [Accessed 6 Aug. 2019].

[18] Medium. (2019). People Tracking using Deep Learning. [online] Available at: https://towardsdatascience.com/people-tracking-using-deep-learning-5c90d43774be [Accessed 16 Aug. 2019].