CLOUD-BASED LECTURE CAPTURING SYSTEM:

A CASE STUDY

IT15010322, L. C. Tennakoon

IT15021076, H. M. S. V. Mudalige

IT15087898, A. P. Jayasinghe

IT15081094, V. R. Wijayagunawardene

Degree of Bachelor of Science

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

September 2018

CLOUD-BASED LECTURE CAPTURING SYSTEM: A CASE STUDY

IT15010322, L. C. Tennakoon

IT15021076, H. M. S. V. Mudalige

IT15087898, A. P. Jayasinghe

IT15081094, V. R. Wijayagunawardene

Dissertation submitted in partial fulfillment of the requirements for the degree of Science

Department of Information Technology

Sri Lanka Institute of Information Technology

September 2018

DECLARATION

I declare that this is my 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 my 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, I hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The above candidate has carried out research for the B.Sc. Dissertation under my supervision.

Signature of the supervisor: Date:

ABSTRACT

Today, with the rapid growth of technology and usage of internet services, e-learning has become one of the latest trends in the education sector. As a result, students tend to prefer e-learning than being physically present in a lecture due to multiple reasons. This document examines an innovative approach in the form of smart cloud-based system to enhance the current e-learning procedures, particularly in universities. The "Lecture Capturing System" is a cloud-based web application which uses enhanced techniques to provide an interactive e-learning experience to users of the system. It has the ability to support multiple enterprise customers. It uses a facial recognition-based authentication process to allow remote users to login to the system. A Pan-Tilt-Zoom (PTZ) IP camera captures and tracks the lecturer during the lecture session and this is streamed live to remotely logged-in students. The lecturer can also share the computer screen if required. The camera intelligently identifies specific gestures performed by the lecturer to rotate towards the audience with the aid of gesture analyzing algorithms. Attendance of remote online students is marked automatically during a live-streaming lecture by using multiple facial recognition processes executing on the server. Offline recording of lectures is also supported after which the video is split into a series of chapters/thumbnails and the audio is converted to text; each chapter representing a presentation slide and the relevant text. Bandwidth and quota are managed intelligently to ensure the best possible transmission rate with minimum data consumption in order to avoid filling the link to capacity which would result in network congestion and poor performance of the network. This is a revolutionary system which is capable of taking e-learning to the next level as it provides a complete classroom experience and makes the whole learning and teaching process efficient and relaxing.

Keywords - Offline video upload, video thumbnails creation, quota management

ACKNOWLEDGEMENT

The work described in this report was carried out as my 4th year research project for the subject Comprehensive Design Analysis Project. The completed final project is the result of combining all the hard work of the group members and the encouragement, support and guidance given by many others. Therefore, it is my duty to express my gratitude to all who gave me the support to complete this major task.

I am deeply indebted to our supervisor Dr. Malitha Wijesundara, lecturers of Sri Lanka Institute of Information Technology, and external supervisor Mr. Pramadhi Atapattu whose suggestions, constant encouragement and support in the development of this research, particularly for the many stimulating and instructive discussions. I am also extremely grateful to Mr. Jayantha Amararachchi, Senior Lecturer (Higher Grade) who gave and confirmed the permission to carry out this research and for all the encouragement and guidance given.

I also wish to thank all our colleagues and friends for all their help, support, interest and valuable advices. Finally, I would like to thank all others whose names are not listed particularly but have given their support in many ways and encouraged me to make this a success.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Table 1: List of Abbreviations

Acronym	**Definition**
OBS Studio	Open Broadcaster Software Studio
FLV	Flash Video
AVC	Advanced Video Coding
AAC	Advanced Audio Coding
OpenCV	Open Source Computer Vision Library
RGB	Red-Green-Blue Color Model
MP3	MPEG-1 Audio Layer III / MPEG-2 Audio Layer III
CSV	Comma-Separated Values
HSV	Hue, Saturation, Value
PTZ	Pan Tilt Zoom
DESC	Description
IN	Input
OUT	Output
PR	Processing
DEP	Dependents
FRT	Finite Ridgelet Transform

1. INTRODUCTION

1.1 Problem to be addressed

Today, e-learning platforms are used for knowledge transfer through electronic media. This transfer can address several learning contexts, ranging from conventional classroom delivery to online and offline distance learning tactics.

There are a countless number of e-learning platforms which cover various aspects of learning such as video streaming, capturing the audience in the lecture hall if necessary, and screen sharing. But, the solution we propose is to handle advanced and enhanced features. Biometric recognition of participants to ensure authentication is proposed so that the attendance can be recorded. In addition to this, every lecture will be maintained as an mp4 video with the set of slides used, and the lecturer's voice relevant to each slide, which provides an easy way of reference to the students who missed a particular lecture. Another important aspect considered in this research is the way of interaction between the lecturer and the students who have any doubts to be cleared during the lecture. For this purpose, when the students who are physically present in the lecture hall are concerned, a gesture-based system is proposed. Here, when the lecturer notices a specific student with the gesture of asking a question, the lecturer will have to perform a gesture for the camera to turn towards the audience and focus on the specific student who has the doubt by once again detecting the gesture performed by the student. When a remotely logged in user has a question for the lecturer, he/she has to signal the lecturer using a specific command, and then the lecturer has to decide whether to give video and/or audio control over to the specific user. As real time video streaming consumes a lot of quota and bandwidth, the system has to intelligently manage data usage by ensuring the best possible transmission rate with minimum data consumption.

1.2 Background context

Today, communicating over the internet and doing things right from our home is becoming more and more practical because it makes day-to-day life very easier. People always try to find a way to use the most convenient way to complete their tasks without wasting too much time and energy.

The art of learning should evolve over time. Even today with the technology we have, we are still used to physically going to a lecture and listening to the lecturer in real-time and making notes of his/her teachings, what the lecturer sketches on whiteboards or his/her presentation. So, there is a higher chance that students forget what the lecturer said or taught the very next day if they were not properly documented. What if a student gets sick or due to unavoidable circumstances, misses a lecture, how is he/she going to learn the missed session? Something like a live streaming system with recording sessions of a lecture would help all the students even if they were present in the lecture itself. The ability to see what they have missed if the student comes to the lecture after it has started, a way to refresh their memories before attending the next lecture would result in a huge academic improvement.

E-Learning helps learning become possible to anyone who wants to learn something right from the comfort of their home. The main purpose of this system is to provide a more effective way to help students get learning materials and information from anywhere and to quickly recap any forgotten or missed lectures via previous recordings of the lecture sessions.

A system and method for an interactive, internet-based video conferencing multicast operation which uses a video production studio with a live instructor giving lectures in real-time to multiple participating students. The video conference multicasting permits the students to interact with the instructor and other installations during the course of the lecture and to later browse the recorded session without a hassle.

There are many advantages to this such as sharing a lecturer's computer screen with students in real time, lecturer communicating with students either by voice, video or both on request of the student. In our system we have added smart features such as intelligent bandwidth and quota management to make sure that we use the least possible data bandwidth to transfer the videos to the students.

Web Application development skills, Image processing, Machine learning and techniques are essential. In addition, some basic knowledge on lecturing processes and student behavior processes are also required throughout the development of the system in order to provide the best learning experience possible.

1.3 Research gap

Features that make the Lecture Capturing System functionalities unique from currently existing systems are mentioned in Table 2. It shows a comparison between the proposed system and other systems. When observing this table, we can see that the feature for creating thumbnail video chapters is a unique feature which is unavailable in any of the existing systems.

Table 2: Comparison with existing systems

Features	**Open Broadcaster Software**	**Panopto**	**Kaltura**	**The Smarter Video Platform**	**Lecture Capturing System**
Support one to many users to concurrently access a live stream.	✓	✓	✓	✓	✓
Gesture based system to detect the speaker, pan, tilt and zoom.					✓

Thumbnail chapters creation of lectures and voice-to-text conversion					✓
Screen sharing window for sharing the laptop screen	✓			✓	✓
Handle user management, course management, bandwidth and quota management	✓	✓	✓	✓	✓
Biometric facial recognition for authenticating users					✓
Capture the audience if required and focus on a guest speaker			✓	✓	✓
Intelligently manage data usage for minimum data usage					✓

1.4 Research questions

The Lecture Capturing System is not just another web-based e-learning application as it is unique in its own way such that it addresses the questions that are present in current systems and integrates together a very useful and distinctive e-learning system. Below are some of the main problems that led to the development of Lecture Capturing System.

To a student who is enrolled in a lecture

- No way of knowing what has happened on a day if he/she was unable to attend the lecture physically.
- Unable to see the whiteboard or the lecturer clearly if the lecture room is too crowded.
- No way to revive the last lesson the lecturer did if they have not taken down notes.
- No way to view what the lecturer did using his/her computer later on. (e.g.: typing a partial code, annotating a PowerPoint slide)
- Not being able to interact with the lecturer to solve a question if the lecture room is crowded.
- Entering the credentials to log in to a system is not fully efficient and is more vulnerable to password attacks and more security risks. If a password is forgotten by a user, there will be many login attempts, password reset requests through email and other security checks that should be done to authenticate the user, thus leading to more password attacks such as brute-force attacks to crack passwords.
- In current systems such as Panopto and Echo360, the session expiration time is too long leading to more security attacks due to the hassle of entering the credentials to log in to the system.
- Lack in awareness of password best practices when creating or updating passwords. Students or people in general follow a common formula or pattern

which is using words with numbers and a special character at the end. These patterns make it easier to remember the credentials while hackers are also aware of the common patterns used to create passwords. As a result, hackers can use this knowledge to input how their brute-force systems run through password combinations or crack passwords by making an educated guess.

- Having to watch an entire lecture video to find some minute detail. No way to just go to the presentation slide which the student is specifically looking for.

To a lecturer who is teaching a course

- Unable to do a lecture from home and upload it to a server so that all the students can watch it.
- Tracking the real attendance of the students is not completely accurate as friends of the students can mark the attendance for absent people.
- Sometimes sharing computer screen reveals all the files in the computer as well as open programs and browser tabs. There is a privacy issue here.
- The lecturer is not aware of a student's arrival and departure time from a lecture session.
- The lecturer is not aware whether a student in the lecture session or classroom has access to the lecture session, is authorized or enrolled to the course.

2. BODY OF THE REPORT

2.1 Addressing the Literature

In recent years, there has been a number of research efforts done to address the needs of smart e-learning management systems. Below are some of the software functionalities and technologies that have been done prior to our research. Undertaking a literature survey helps us to find and come up with the following.

Regardless of the enormous growth of e-learning (electronic learning) in education and its perceived benefits, the efficiency of such e-learning systems will not be fully utilized if the students are not inclined to accept and use the system.

As a result, successful implementation of e-learning tools depends on whether the students are willing to adopt and accept the technology. Thus, it has become imperative for e-learning system developers to understand the factors affecting the user acceptance of web-based learning systems in order to enrich the students' learning experience and to create a better product to fulfill the necessary requirements of the student.

"Use of E-Learning", a research was done to find University students' purpose to use e-learning [1]. In this research, Teknologi Malaysia University's students try to apply and use the theory of technology acceptance model (TAM). They have employed structural equation modeling (SEM) approach with a SmartPLS software to investigate students' adoption process. Discoveries indicate that the content of e-learning and self-efficacy have a positive impact and substantially associated with perceived usefulness and student satisfaction, which impact university students' purpose to use e-learning. Although e-learning has expanded acceptance in universities around the world, the study of the intention to use e-learning is still essentially unexplored in Malaysia. The developed model is employed to explain the university student's intention to use e-learning. The study concludes that university students in Malaysia have positive perceptions towards e-learning and intend to practice it for educational purposes.

E-Learning is reflected as an innovative approach to education delivery via electronic forms of information. Multiple researches have been done to find the best way to use the technology and to better fit the students' necessities [1], [2]. The main obstacles that need to be addressed are the insufficient financial support, inadequate training programs, lack of ICT infrastructure, equivocal policies and objectives, and lack of awareness, interest, and motivation toward e-learning technology are considered as the main obstacles to enhance e-learning in Iraqi universities. The lack of training programs and inadequate ICT infrastructure are considered as the key issues which obstruct advancing of the e-learning process in Iraq [2].

Online body tracking by a PTZ camera has been done before to automatically track a single person and focus on that person [3], [4]. Online human body tracking method by an IP PTZ camera based on fuzzy-feature scoring was done. At every frame, candidate targets are detected by extracting moving targets using optical flow, a sampling, and appearance. The target is determined among samples using a fuzzy classifier. Results show that the system has a good target detection precision (> 88%), and the target is almost always localized within 1/4th of the image diagonal from the image center [3]. Autonomous lecture recording with a PTZ camera [4]. This reaches the same viewing experience while watching lectures recorded by an automated system. To accomplish this, they have developed an automatic cameraman (PTZ camera-unit) that is able to, Detect and track a single person/lecturer, change between different types of shots, Listen to high-level instructions from a virtual or human director, and Take cinematographic rules into account

By tracking the lecturer, he is framed well in the picture at any moment and viewers can't be distracted. Takes cinematographic rules into account, which ensures that the viewer remains focused and the viewing experience is aesthetically more interesting. The action axis is determined by calculating the direction of movement and the gaze orientation. A PID control loop ensures smooth movement of the camera. Because of the speed of the algorithm, it will be easy to downscale for embedded hardware and still perform the calculations real-time.

Remote controlling of the PTZ camera system for lecture rooms [5]. This consist of a simple and inexpensive software solution for remote management of PTZ camera systems. This provides the ability for users to remotely control the PTZ camera system from one place with the simultaneous image capturing ability. Users of this application are able to choose a number of presets and this functionality is provided programmatically by sending queries to CCTV system, which then responds back to the controller. All of the responses are processed immediately into the desired form and stored in a selected row in SQLite database. This method of data storage doesn´t require the installation of SQL database, which makes the solution easier to apply. The program was created using the programming language C#. But this software solution does not support real-time tracking of a person, just several predefined presets so that feature can be improved to real-time operation in Lecture Capturing System. OpenTrack - Automated Camera Control for Lecture Recordings [6] records lecture sessions automatically without the need for a human camera person. A Tabletop Lecture Recording System [7]. This research presents a lecture recording system that employs gestures and digital cameras to facilitate remote distance teaching. Virtual Cameraman [8] uses two PTZ cameras having different utilities. One is named full-shot PTZ camera and the other is movement PTZ camera. To get camera movement information, the system first obtains continuous images from full-shot PTZ camera and theSpn extracts four fuzzified movement features which can represent four characters of audiences' motions respectively. On the other hand, an automatic camera movement model (ACMM) is constructed by recording photographers' habit of CM styles and shot types. The proposed system can select suitable CM styles and shot types by inputting the fuzzified motion features into the ACMM. After that, the system chooses the main target in the input frames obtained from the full-shot PTZ camera by using five aesthetic criteria. Finally, the system operates the movement PTZ camera to finish recording.

Real-time tracking of a non-rigid target with a moving pan-tilt-zoom (PTZ) camera.

The tracking of the object and control of the camera is handled by one computer in real time. The main contribution of the paper [9] is method for target representation, localization and detection, which takes into account both foreground and background properties, and is more discriminative than the common color histogram based back-projection. A Bayesian hypothesis test is used to decide whether each pixel is occupied by the target or not. The target representation is suitable for use with a Continuously

Adaptive Mean Shift (CAMSHIFT) tracker. Experiments show that this leads to a tracking system that is efficient and accurate enough to guide a PTZ camera to follow a moving target in real time, despite the presence of background clutter and partial occlusion.

Human tracking applications with a Global Positioning System (GPS) can get the user's position in real time on the open area. But if the user goes into the room or building, then the human tracking application cannot get the user's position as the GPS signal cannot get through the wall. By using RFID, human tracking application can perform tracking in a certain closed area by providing room's or building's information that is entered. The IP camera as part of the system will send the images as visualization inside the room. In this research, human tracking system is built on a specific area that has lots of room or building such as a theme park or sports club. System is designed using an RFID reader, RFID tags, IP Camera, the database server and Android smartphones. The research was done inside Syahdan Campus, Bina Nusantara University – Jakarta. The result shows that the application is working with 100% tapping and mapping accuracy [10].

IP-based physical security products like IP network camera (IPNC) are becoming essential in construction of a modern security system. In order to guarantee interoperability among them, ONVIF has been a de facto standard communication framework. In addition to core specification, ONVIF defines many services. ONVIF Event service is supposed to provide notification messages to registered clients when events happen, which is an essential mechanism to be support to make IPNC intelligent. In this paper, we report our efforts to implement ONVIF Event service for the smart IPNC. First, we design S/W architecture, necessary data structures, and workflow of ONVIF Event service according to ONVIF Event service specification. Then, we implement the design about ONVIF Event service by extending TI’s IPNC reference RDK S/W package [11]. Testing via an Open source ONVIF client verifies our implementation works properly.

Tracking people or moving objects across a PTZ camera and maintaining a track within a camera is a challenging task in applications of video surveillance. In this paper we propose a novel object positioning tracking framework, PTZ camera-based position tracking system, which is also known as PCTS, can help estimate the position of an object by using camera parameters, pan and tilt. From object motion vector, the relevant information such as time, background and geographic parameters can be recorded in database as well [12]. In the experiment, the change of a person' position is recorded by the PTZ camera in real time. The PCTS provides a feasible solution for position analysis and security surveillance services in future.

Real-time person tracking has been implemented before, but not quite as what we have planned on doing; real-time broadcasting of the footage without any delay. The complete package of having the lecture capturing along with audience if necessary, screen sharing, Face Recognition based remote login and attendance marking for online participants, viewing the lecture in real-time with added features such as reading what the lecturer has told and intelligently generating chapters on the video according to the lecture slides played alongside with this makes Lecture Capturing System a perfect complete package of e-learning. A thorough research related to e-learning systems has led to the identification of some of the most influential factors used in the field of information systems research. More specifically, characteristics as well as the limitations, weaknesses, and strengths of web-based learning systems. Student variables, such as technical issues and adapting to the new ways are important variables that influence student learning, especially in a collaborative e-learning environment. Understanding these variables is now helpful for developers to design eloquent educational activities to promote student knowledge construction and make learning more effective and appealing. In particular, this research helps to better understand the characteristics of students and to comprehend what the students expect from the learning management systems. This can help the developers achieve the most effective deployment of such systems and also helps them improve their strategic decision making about technology in the future, they can decide on the best approach that fit their students before implementing any new technology.

Several work exist in facial recognition-based authentication, attendance marking and bandwidth management areas. Haar classifier for face detection, Eigen face algorithm for face recognition, image normalization and histogram normalization for image and contrast enhancement is proposed [13]. The system uses IP Camera mounted in front of a classroom which continuously capture image of the entire class at set interval, throughout the period of a lecture and sends the images over the internet to a cloud server for processing. The server processes the images by detecting the human faces contained, extract the faces and matches them with the enrolled faces of the students stored on the database. Before the images are used for detection and recognition purposes, the images are normalized or converted to gray scale which enhances the accuracy of face detection and recognition. Histogram normalization is then proposed for contrast enhancement [13].

Sujata G. Bhele and V. H. Mankar has described wide range of methods used for face recognition which includes Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Independent Component Analysis (ICA) and Gabor wavelet soft computing tool like Artificial Neural Networks (ANN) for recognition and various hybrid of this techniques. In addition, the methods that challenges face recognition such as pose variation, illumination and facial expressions are described further [14].

In terms of accuracy and speed of face detection and recognition processes, a new approach is proposed by evaluating various face detection and recognition methods such as Haar-like features, LBP, SVM, Adaboost algorithm, Gabor features and HOG by performing tests on face rich databases in terms of subjects, pose, emotions, race and light [15]. AdaBoost classifier is used with Haar and Local Binary Pattern (LBP) features whereas Support Vector Machine (SVM) classifier is used with Histogram of Oriented Gradients (HOG) features for face detection evaluation in [15]. Haar-like features is the proposed face detection solution in [15] which has reported relatively well than LBP method. In terms of recognition, Gabor method is the proposed solution in [15] as it’s qualities overcome datasets complexity.

Traditionally in live streaming, data is served by putting a huge network load on the servers. A new approach is proposed which introduce two new WebRTC-based communication protocols [16] known as WebPeer and CodedWebPeer designed especially for browser based P2P streaming [17]. Two network metrics called network health to measure overall data saturation and network stability to show if the peers in the network are able to serve each other without the help of the server are proposed. The study [17] shows through measurements that by applying network coding, network health is increased by up to 100% without changing the cache size or number of peers. Furthermore, network stability is achieved using up to half the cache compared to the other approaches, without increasing the servers load. A caching system is proposed to reduce bandwidth costs when live streaming [17].

A research conducted by M. Y. Abbass and HyungWon Kim revolves around a novel method used for image separation. It incorporates the property of pyramid component extracted from the image and a finite ridgelet transform (FRT) to obtain a precise analysis of the images and thus correctly separate the images even in a highly noisy environment [18]. Jiandong Cao describes an urban intelligent traffic monitoring system which uses a recognition algorithm based on Haar features combined with AdaBoost classifier [19].

2.1.1 Existing Applications

i. Panopto - Lecture Capture Software

Panopto is an easy-to-use video platform for training, presenting, and communicating that enables users to record videos and rich media presentations and push out to subscribers in many different formats. Panopto is built with the flexibility to record any combination of video sources, in any configuration, in classrooms of any size. And Panopto scales with ease to meet institution’s needs from small departmental deployments to campus-wide installations [20].

ii. BigBlueButton

BigBlueButton is an open-source web collaboration software utilized by education organizations for e-learning and training. The software offers numerous options for customization and integration as per requirements of the users. BigBlueButton enables users to conduct web-conferencing and share documents, audio and video files for online learning. The software's "whiteboard" feature allows presenters to mark valuable topics in the presentation. In addition, its "polling" feature engages learners and helps the presenter to receive feedback. BigBlueButton's "desktop sharing" feature extends beyond slides and allows moderators to share their screen with the audience enabling a better understanding of topics. BigBlueButton supports multiple users in a video conference with no cap on numbers of active webcams. The software also supports voice conferencing via Voice Over IP (VOIP) without additional hardware requirements [21].

iii. Echo360

Echo360 combines video management with lecture capture and active learning to increase student success.

Echo360 keeps notes linked to class presentations and videos so that students can jump straight from their own words to those of the instructor and replay the entire learning experience.

High-quality live streaming supports remote learners and classroom overflow situations. The streaming experience leverages Echo360's engagement tools so students can engage with classmates and the instructor no matter where they are.

Built on a scalable cloud architecture, the platform allocates the resources needed to process peak loads automatically. Videos are uploaded and processed in real time so the optimized version is available as soon as class is over [22].

iv. Kaltura

The Kaltura Video Player SDK includes a rich set of APIs for player embedding, customization, white-labeling and integration via JavaScript or ActionScript 3.0. By leveraging the Kaltura Player you can create your own custom players with less effort and at no cost [23].

- Endless flexibility for creating your own custom design and playback experiences
- Automatically switch between HTML5 video and Flash, maintaining a unified look & feel
- Work with any type of streaming protocols, from adaptive streaming, HTTP streaming, and DRM
- Increase engagement with smart and dynamic playlists, related, and more
- Optimize SEO using Kaltura's SEO best practices embedding guidelines.

v. Open Broadcaster Software (OBS) (Software Tool)

OBS is a free and open-source streaming and recording program maintained by the OBS Project. The program has support for Windows 7 and later, OS X 10.10 and later, and Ubuntu 14.04 and later.

OBS is a free and open-source software suite for recording and live streaming. Written in C and C++, OBS provides real-time source and device capture, scene composition, encoding, recording, and broadcasting. Transmission of data is primarily done via the Real Time Messaging Protocol (RTMP) and can be sent to any RTMP supporting destination, including many presets for streaming websites [24].

2.2 Use Cases

Some use case scenarios that were identified in the designed system relative to the video chapter creation context can be described as follows:

Table 3: Use case scenario for offline screen capture

Use Case No	01
Use Case Name	Offline screen capturing and uploading
Actors	Lecturer
Pre-Conditions	OBS Studio (including the developed plugin) is installed in the lecturer's computer.
Main Success Scenario	1. The lecturer records the lecturer's desktop screen offline using OBS Studio. 2. The lecturer logs into the server with the OBS Studio plugin using a valid username and password. 3. The recorded video is uploaded to the server.
Post Conditions	The recorded video is successfully uploaded to the remote server.
Extensions	2a. The lecturer enters an invalid username/password to connect to the remote server. 2a1. The plugin displays an error saying "Invalid username/password. Try again." 3a. The internet connection is interrupted during the video transfer. 3a1. The plugin displays an error "Internet connection disrupted. Please upload the video again". 3a2. The lecturer uploads the video again.

Table 4: Use case scenario for video thumbnails creation

Use Case No	02
Use Case Name	Video thumbnails creation
Actors	Lecturer

Pre-Conditions	Lecturer is logged into the web application.
Main Success Scenario	1. The lecturer views the list of the recorded lectures. 2. The lecturer selects one video and clicks on the “Create Video Thumbnails” button. 3. A series of video thumbnail chapters is created from that video.
Post Conditions	The recorded video is split into a series of thumbnail chapters, and a success message is displayed to the user.
Extensions	2a. The video is corrupted so the system is unable to process the file. 2a1. The system displays an error saying “Corrupted video file. Please try again.”

Table 5: Use case scenario for quota management

Use Case No	03
Use Case Name	Edit the data quota of a user/users
Actors	Administrator
Pre-Conditions	The administrator is logged into the system.
Main Success Scenario	1. The administrator selects the “Quota Management” tab. 2. The list of users is displayed filtered by user type, month, and year. 3. The administrator clicks on the “Edit monthly quota for all” link. 4. A new value is entered for the monthly data quota value.
Post Conditions	A message saying “Monthly quota successfully updated” is displayed.
Extensions	4a. The administrator enters an invalid value (e.g. letters) in the monthly quota field. 4a1. The system displays an error saying “Invalid value. Please enter a number.”

2.3 Methodology

This section describes how the Lecture Capturing System will be designed and implemented; explaining the process of each functionality, their flow in the system and the technologies used for their implementation.

The system is a cloud-based web application which is capable of supporting multiple enterprise customers. Remote students can access the system after logging in via biometric authentication (facial recognition). A PTZ IP camera provides a continuous video stream of the lecturer and audience to the central server via the Kurento media server during a lecture session. The server broadcasts this stream live to the remote students. The lecturer's screen is also shared if required. The attendance of remote students is marked by capturing their faces through their webcams and running a neural network algorithm on the server for identification. Lecturers are also able to do an offline lecture recording and then upload it to the server where this will be split into chapters and the audio to text. Figure 1 shows the system with the main components and their interactions.

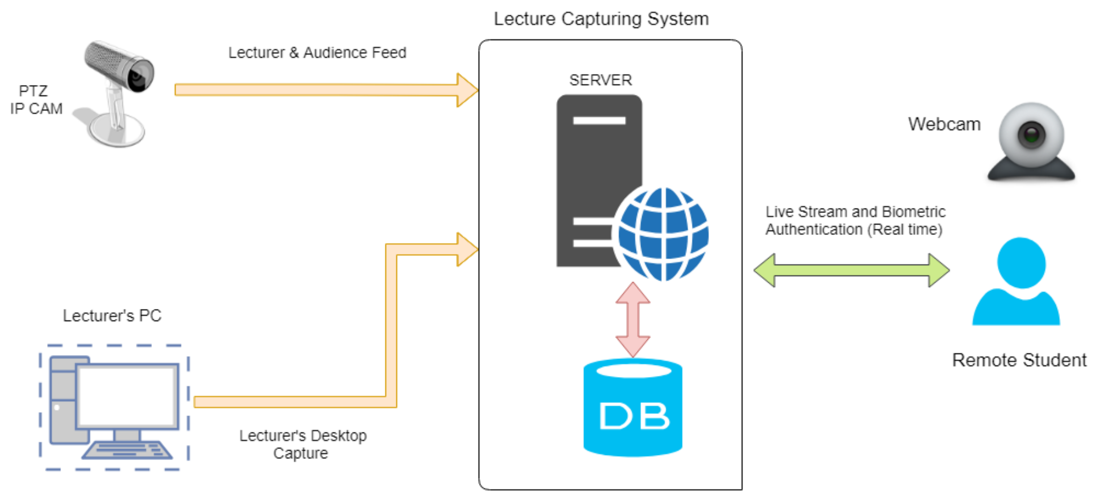


Figure 1: System Architecture

2.3.1 Audio and Video Conferencing

An IP camera will be used to track the lecturer's movement and gestures in front of the camera and produce the necessary PTZ signals to pan, tilt and zoom accordingly thus ensuring that the lecturer's actions are always recorded without missing any detail. This video recording will be immediately compressed on the fly in order to reduce its file size, then streamed live and also saved in the database for backup purposes and viewing later. Therefore, the students have the choice of attending the lecture via the live stream or listening to the lecture later. This is very beneficial to students since they can also attend lectures without being physically present (remotely) at the lecture. During the live streaming session, the system will decide which video resolution (e.g. 480p, 720p, 1080p) to use for the playback at the student's end depending on the speed of his/her internet connection.

Live streaming is achieved via Kurento which is a WebRTC (Web Real-Time Communications) media server and a set of client APIs. During the live streaming session, the lecturer also has the ability to share his/her entire computer screen with the participating students if required, making certain that not even the most minute detail is not missed. In the case of having low bandwidth to support this feature, the lecturer has the option to disable the IP cameras in order to save bandwidth. This mode of lecturing provides better participation and interaction between the lecturers and students. An example of this is if a student wants to ask a question, the control would be given to the particular student by the lecturer and the application would support audio only, video only or both audio-video sources of the particular student. But the lecturer has the ability to get back the control of the audio and video sources of the system when required.

2.3.2 Live Stream – Auto Tracking Lecturer

Granting multiple platforms support live streaming of a video, what makes Lecture Capturing System unique is its ability to automatically focusing on the lecturer in real-time, lecturer is framed well in the picture at any moment and viewers can't be distracted. Takes cinematographic rules into account, which ensures that the viewer remains focused and the viewing experience is aesthetically more interesting and gives the viewers the best viewing angle all the time without a need for a separate cameraman.

An IP camera will be used to track the lecturer's movement and gestures in front of the camera and produce the necessary PTZ signals to pan, tilt and zoom accordingly thus ensuring that the lecturer's actions are always recorded without missing any detail. This video recording will be immediately compressed 'on -the-fly' in order to reduce its file size, then streamed live and also saved in the database for backup purposes and viewing later. Therefore, the students have the choice of attending the lecture via the live stream or listening to the lecture later. This is very beneficial to students since they can also attend lectures without being physically present (remotely) at the lecture.

2.3.3 Obtaining the Live Stream

WebRTC (Web Real-Time Communication)

WebRTC is a set of protocols, mechanisms, and APIs that provide browsers and mobile applications with Real-Time communication capabilities over peer-to-peer connections [25]. WebRTC has proven to be a technology that allows well defined communication between browsers without the mediation of any kind of intermediate software as shown in Figure 2.

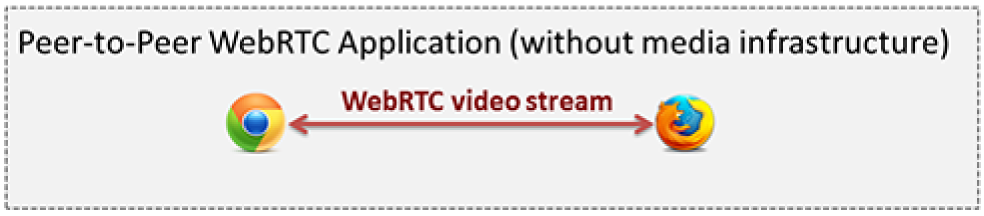

Peer-to-Peer WebRTC Application (without media infrastructure)
WebRTC video stream

Figure 2: Peer-to-Peer WebRTC

This method of using WebRTC protocol directly can easily be used with simple web applications.

Lecture capturing system requires features such as live video and audio streaming, video recording, and audio/video conferencing. For this purpose, there is a need of a WebRTC Media Server to mediate the communication between browsers as shown in Figure 3.

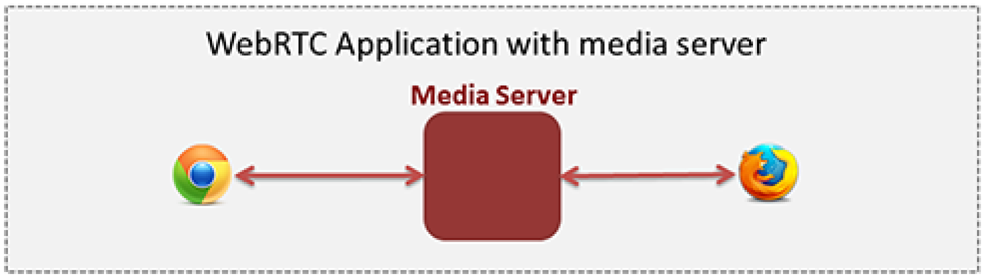


Figure 3: WebRTC Application

Kurento Media Server

Kurento is a WebRTC Media Server and a set of client APIs that simplify the development of advanced video applications for web and smartphone platforms. Its features include group communications, transcoding, recording, mixing, broadcasting and routing of audiovisual flows.

With Kurento, it’s an easy task to add third-party media processing algorithms to any WebRTC application, like integrating Computer Vision, Augmented Reality, video indexing, and speech analysis.

Kurento Media Server is ideal to be used with the Lecture Capturing system because of all the features that are mentioned above. Furthermore, Kurento Media Server is an open source software which provides all of the following advantages:

- Kurento Media Server and applications can be collocated, escalated, or distributed among different machines.
- Developers do not need to be aware of internal complexities of Kurento Media Server.
- Ability to support multiple video and audio sources.
- Streaming from sources such as PTZ cameras.
- End-to-end communication capability.

The main component that our Lecture Capturing System uses is the ability of Kurento Media Server to obtain a live stream from an IP camera.

Obtaining an RTSP stream using Kurento Media Server

The default video source supported by Kurento Media server is the webcam installed in the computer. However, this application requires to capture the lecturer's live video using an IP camera, and stream it to all remotely logged users.

The camera used in Lecture Capturing System uses the RTSP (Real Time Streaming Protocol) protocol for establishing a RTP (A Transport Protocol for Real Time Applications) media session.

To integrate such an IP camera with a webRTC application, it is first required to achieve media interoperability [26].

Kurento Media Server provides a set of endpoints that are designed for this purpose.

- The Principal component which is responsible for capturing a video stream from an IP camera is the PlayerEndpoint.
- Kurento Media Server WebRTCEndpoint facilitates publishing media streams to WebRTC browsers with full termination of RTSP feedback.
- Kurento Media Server agnostic media capability performs, transparently for the developer, all appropriate trans-codifications when two incompatible media elements are connected, just by connecting the PlayerEndPoint to the WebRTCEndpoint.

2.3.4 Camera Movement with Detection

OpenCV Object Detection using Haar feature-based cascade classifiers is an effective object detection. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images [27].

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the Figure 4 are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

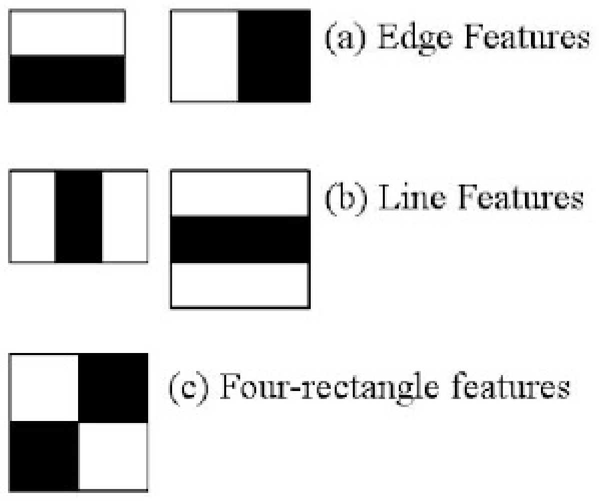


Figure 4: Haar features

Now, all possible sizes and locations of each kernel are used to calculate lots of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find the sum of the pixels under white and black rectangles. To solve this, they introduced the integral image. However large your image, it reduces the calculations for a given pixel to an operation involving just four pixels.

But among all these features we calculated, most of them are irrelevant. For example, consider Figure 5. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applied to cheeks or any other place is irrelevant.

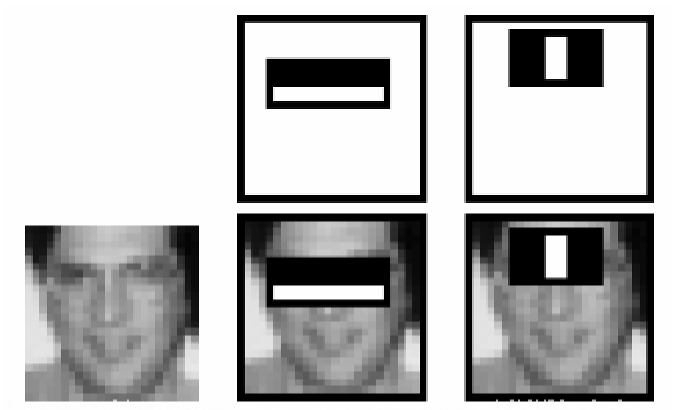


Figure 5: Features Calculated

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then the same process is done. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found).

The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

So now we take an image. Take each 24x24 window. Apply 6000 features to it. Check if it is face or not.

In an image, most of the image is non-face region. So, it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. This way, we spend more time checking possible face regions.

For this they introduced the concept of Cascade of Classifiers. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. (Normally the first few stages will contain very many fewer features). If a window fails the first stage, discard it. We don't consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region.

Haar-cascade Detection in OpenCV

OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc. Those XML files are stored in the opencv/data/haarcascades/ folder. For this detecting lecture purpose we have used Facial Detection Haar cascade [28].

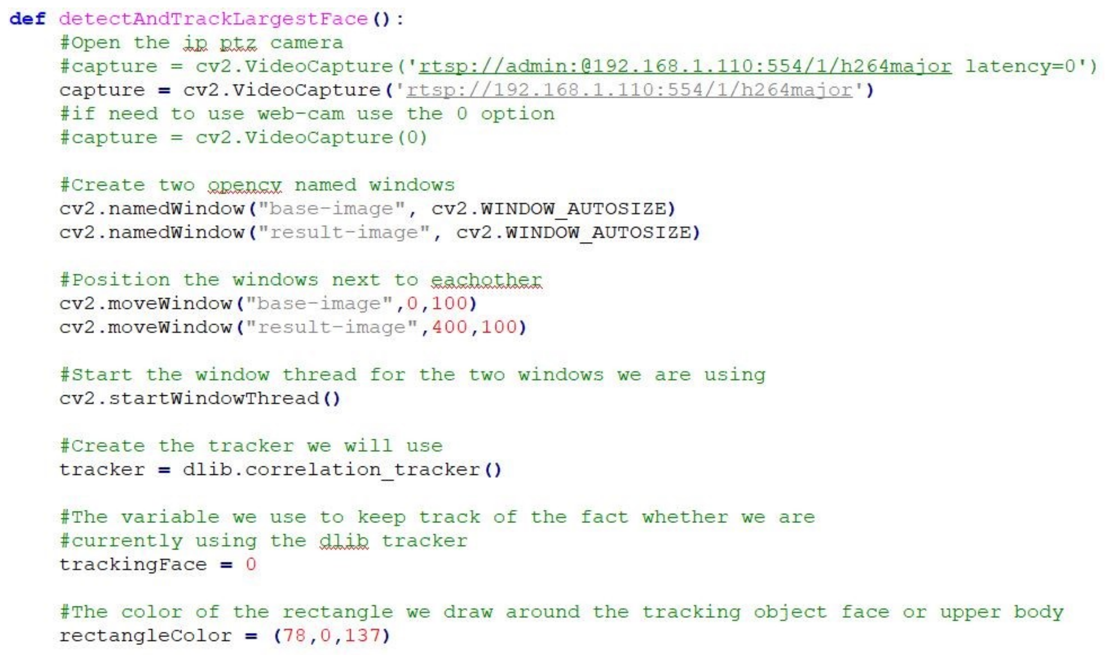
First we need to load the required XML classifiers. Then load our input video stream from IP Camera in grayscale mode.

Now we find the faces in the image. If faces are found, it returns the positions of detected faces as Rect(x,y,w,h). Once we get these locations, we can calculate if the camera needs to turn to make sure the lecturer is detected in the middle.

This lecture tracking script uses OpenCV's Haar Cascade Classifier to do the person detection task. First it initializes a face cascade using the frontal face Haar cascade provided with the OpenCV library. Then it starts to detect And Track the Largest Face it can find, if not tracking Face or lost the tracked face again it uses Haar cascade detector to detect face and then correlation tracker to follow it using dlib.

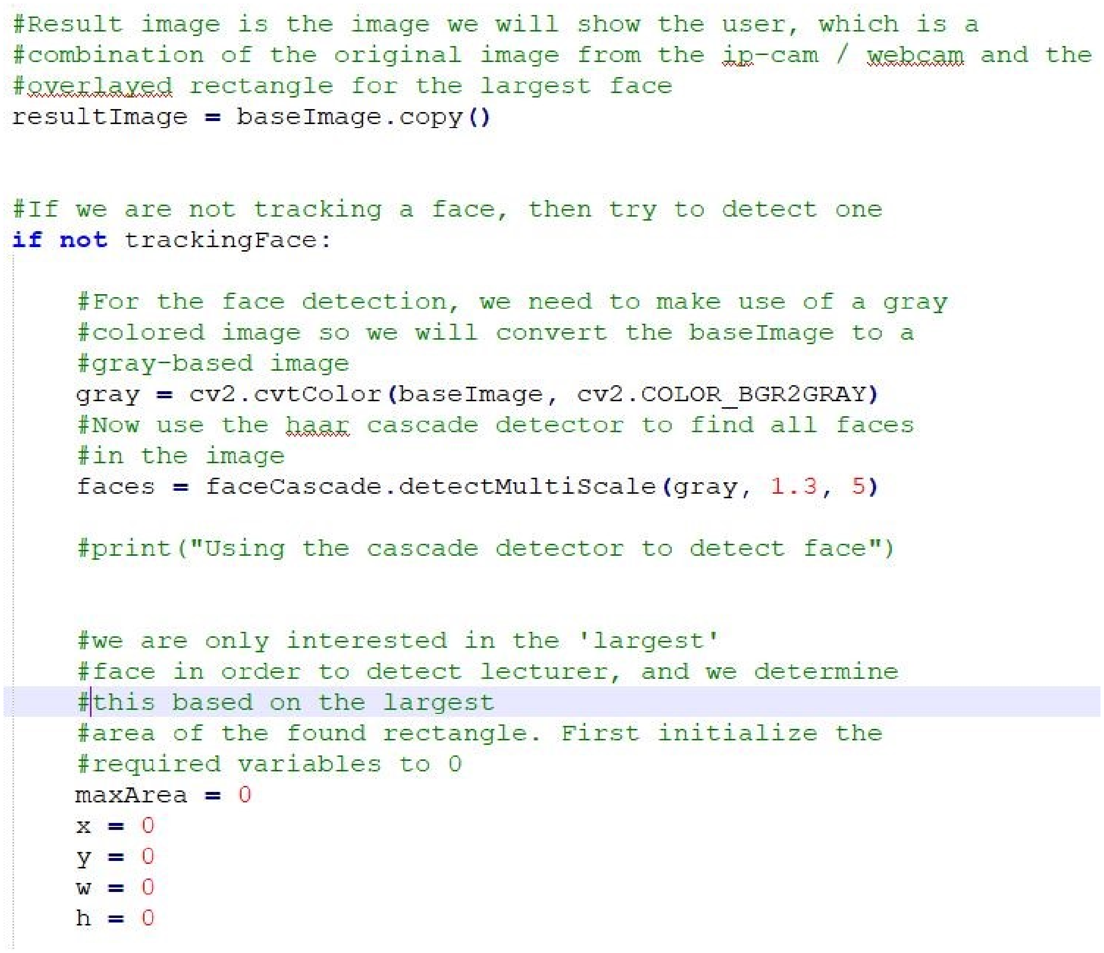
Both methods require to scan each the whole frame with a sliding window. The algorithm then tries to find the features of a person in each window position. These methods are too expensive to perform in each frame if we want to run our person tracker on restricted hardware like a Raspberry Pi.

For this reason, we combine the person detector with a correlation tracker. The correlation tracker expects a region of interest and starts tracking the pixels inside that region. In subsequent frames it tries to find where the pixels have most likely moved. This is much faster and more robust than trying to find the person in each and every frame again.



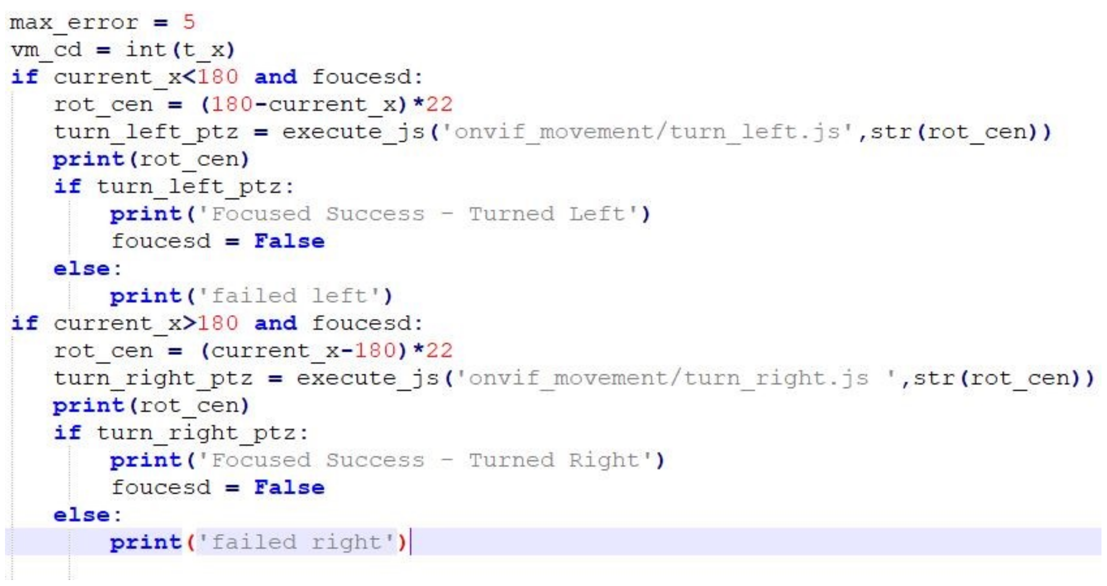
```
def detectAndTrackLargestFace():
    #Open the ip ptz camera
    #capture = cv2.VideoCapture('rtsp://admin:@192.168.1.110:554/1/h264major latency=0')
    capture = cv2.VideoCapture('rtsp://192.168.1.110:554/1/h264major')
    #if need to use web-cam use the 0 option
    #capture = cv2.VideoCapture(0)

    #Create two opencv named windows
    cv2.namedWindow("base-image", cv2.WINDOW_AUTOSIZE)
    cv2.namedWindow("result-image", cv2.WINDOW_AUTOSIZE)

    #Position the windows next to eachother
    cv2.moveWindow("base-image",0,100)
    cv2.moveWindow("result-image",400,100)

    #Start the window thread for the two windows we are using
    cv2.startWindowThread()

    #Create the tracker we will use
    tracker = dlib.correlation_tracker()

    #The variable we use to keep track of the fact whether we are
    #currently using the dlib tracker
    trackingFace = 0

    #The color of the rectangle we draw around the tracking object face or upper body
    rectangleColor = (78,0,137)
```

Figure 6: Detecting largest face in the view to lock in for tracking



```
#Result image is the image we will show the user, which is a
#combination of the original image from the ip-cam / webcam and the
#overlayed rectangle for the largest face
resultImage = baseImage.copy()


#If we are not tracking a face, then try to detect one
if not trackingFace:

    #For the face detection, we need to make use of a gray
    #colored image so we will convert the baseImage to a
    #gray-based image
    gray = cv2.cvtColor(baseImage, cv2.COLOR_BGR2GRAY)
    #Now use the haar cascade detector to find all faces
    #in the image
    faces = faceCascade.detectMultiScale(gray, 1.3, 5)

    #print("Using the cascade detector to detect face")


    #we are only interested in the 'largest'
    #face in order to detect lecturer, and we determine
    #this based on the largest
    #area of the found rectangle. First initialize the
    #required variables to 0
    maxArea = 0
    x = 0
    y = 0
    w = 0
    h = 0
```

Figure 7: If not tracking face trying to detect face of the lecturer



```
max_error = 5
vm_cd = int(t_x)
if current_x<180 and foucesd:
    rot_cen = (180-current_x)*22
    turn_left_ptz = execute_js('onvif_movement/turn_left.js',str(rot_cen))
    print(rot_cen)
    if turn_left_ptz:
        print('Focused Success - Turned Left')
        foucesd = False
    else:
        print('failed left')
if current_x>180 and foucesd:
    rot_cen = (current_x-180)*22
    turn_right_ptz = execute_js('onvif_movement/turn_right.js ',str(rot_cen))
    print(rot_cen)
    if turn_right_ptz:
        print('Focused Success - Turned Right')
        foucesd = False
    else:
        print('failed right')
```

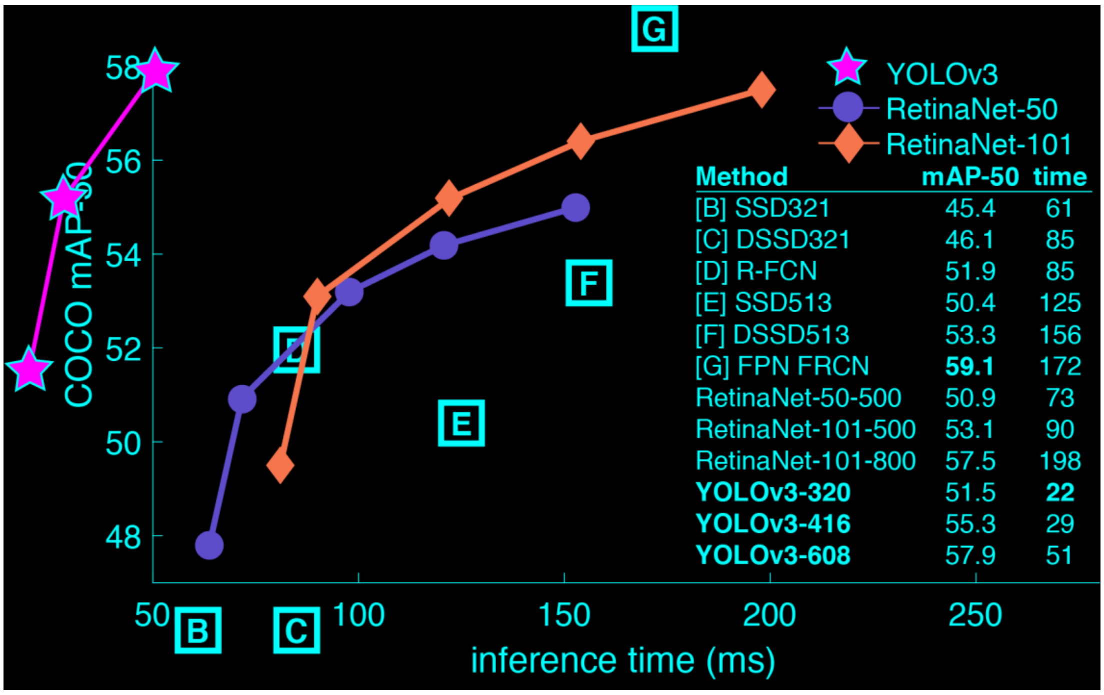
Figure 8: Move the camera to center the lecturer

Person Detection – Secondary

Uses YOLO and Deep Sort with TensorFlow to track the lecturer in the stage. Uses a pre trained model to detect humans [29].

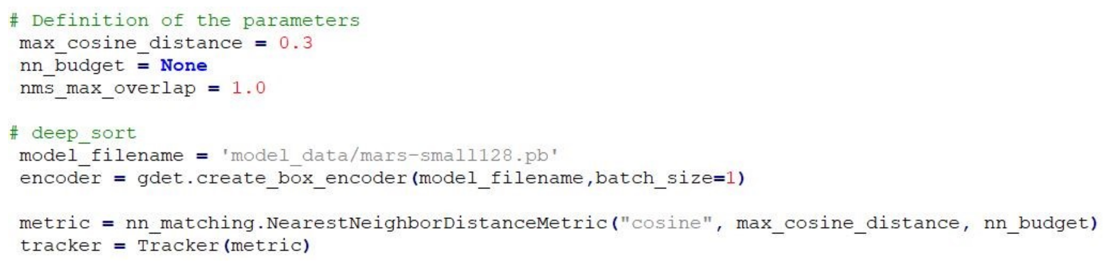
Uses NVIDIA CUDA to use TensorFlow GPU which is extremely fast than using the CPU version.

You only look once (YOLO) is a state-of-the-art, real-time object detection system. On a Pascal Titan X it processes images at 30 FPS.



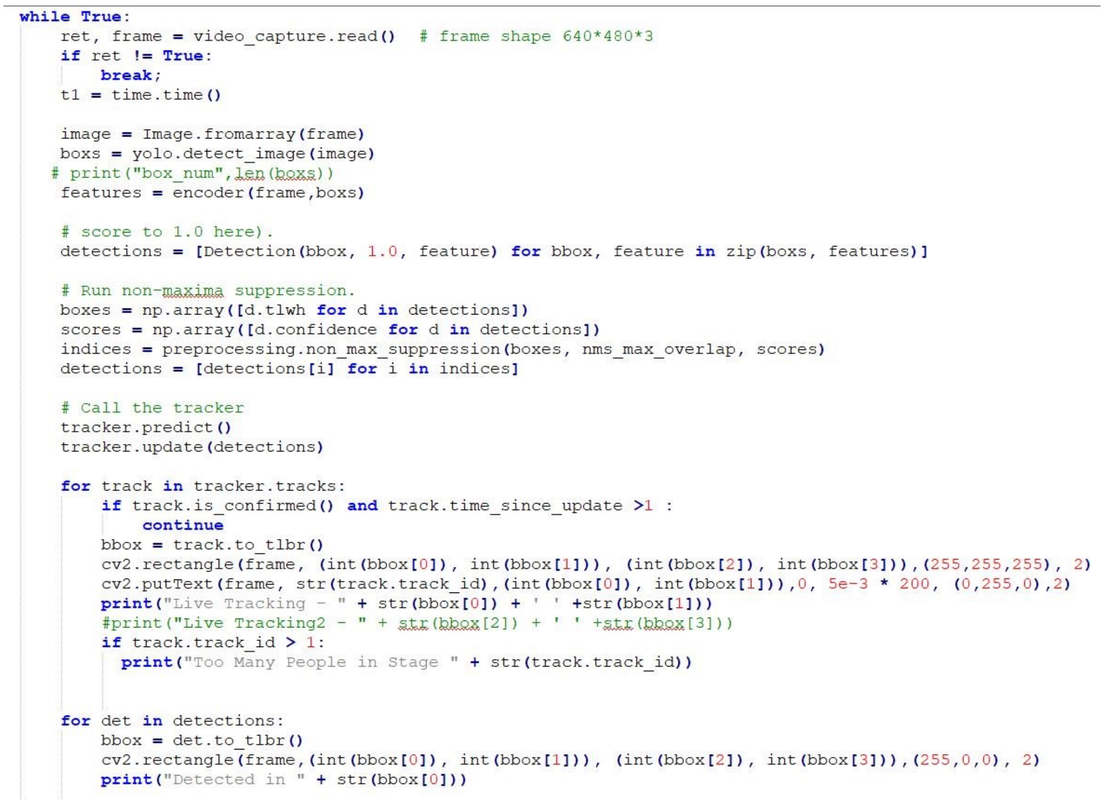
Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Figure 9: Yolo Inference Time Comparison Chart



```
# Definition of the parameters
 max_cosine_distance = 0.3
 nn_budget = None
 nms_max_overlap = 1.0

# deep_sort
 model_filename = 'model_data/mars-small128.pb'
 encoder = gdet.create_box_encoder(model_filename,batch_size=1)

 metric = nn_matching.NearestNeighborDistanceMetric("cosine", max_cosine_distance, nn_budget)
 tracker = Tracker(metric)
```

Figure 10: Load Person Model Data



```
while True:
    ret, frame = video_capture.read()  # frame shape 640*480*3
    if ret != True:
        break;
    t1 = time.time()

    image = Image.fromarray(frame)
    boxs = yolo.detect_image(image)
   # print("box_num",len(boxs))
    features = encoder(frame,boxs)

    # score to 1.0 here).
    detections = [Detection(bbox, 1.0, feature) for bbox, feature in zip(boxs, features)]

    # Run non-maxima suppression.
    boxes = np.array([d.tlwh for d in detections])
    scores = np.array([d.confidence for d in detections])
    indices = preprocessing.non_max_suppression(boxes, nms_max_overlap, scores)
    detections = [detections[i] for i in indices]

    # Call the tracker
    tracker.predict()
    tracker.update(detections)

    for track in tracker.tracks:
        if track.is_confirmed() and track.time_since_update >1 :
            continue
        bbox = track.to_tlbr()
        cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])),(255,255,255), 2)
        cv2.putText(frame, str(track.track_id),(int(bbox[0]), int(bbox[1])),0, 5e-3 * 200, (0,255,0),2)
        print("Live Tracking - " + str(bbox[0]) + ' ' +str(bbox[1]))
        #print("Live Tracking2 - " + str(bbox[2]) + ' ' +str(bbox[3]))
        if track.track_id > 1:
          print("Too Many People in Stage " + str(track.track_id))


    for det in detections:
        bbox = det.to_tlbr()
        cv2.rectangle(frame,(int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])),(255,0,0), 2)
        print("Detected in " + str(bbox[0]))
```

Figure 11: Detect and Track Lecturer

2.3.5 PTZ Camera Movement

ONVIF stands for Open Network Video Interface Forum. It's an open industry standard that provides interoperability among IP security devices such as security cameras, video recorders, software, and access control systems [30].

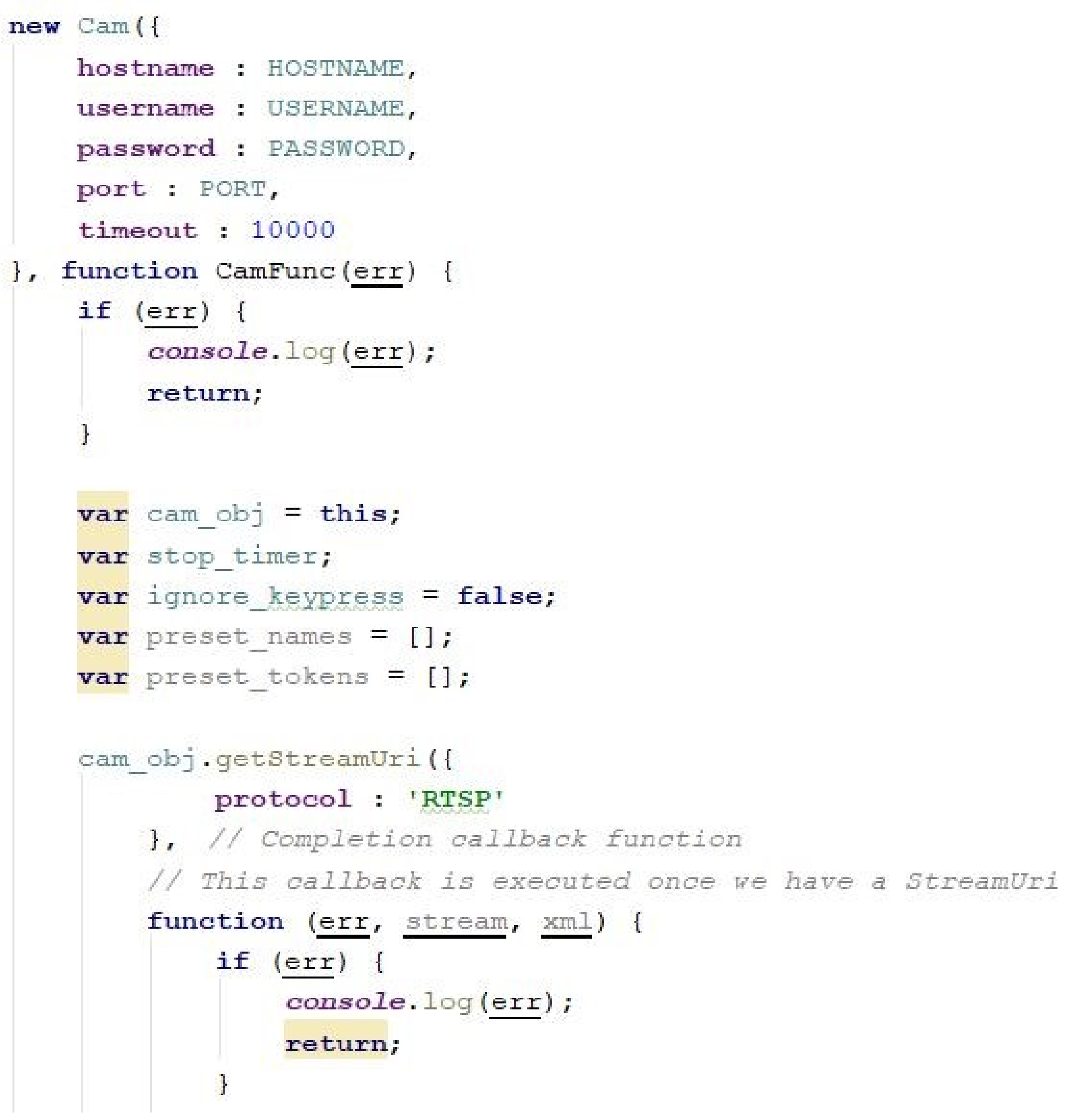
Since we are using ONVIF protocols to move the camera it allows the compatibility from different vendor's devices so LCS will have the support by most IP based security devices manufacturers giving it an added benefit of not limiting the system to a specific brand of IP Cameras.

The node-onvif is a Node.js module which allows you to communicate with the network camera which supports the ONVIF specifications.

The ONVIF (Open Network Video Interface) is an open industry forum promoting and developing global standards for interfaces of IP-based physical security products such as network cameras. The ONVIF specifications are available in their web site.

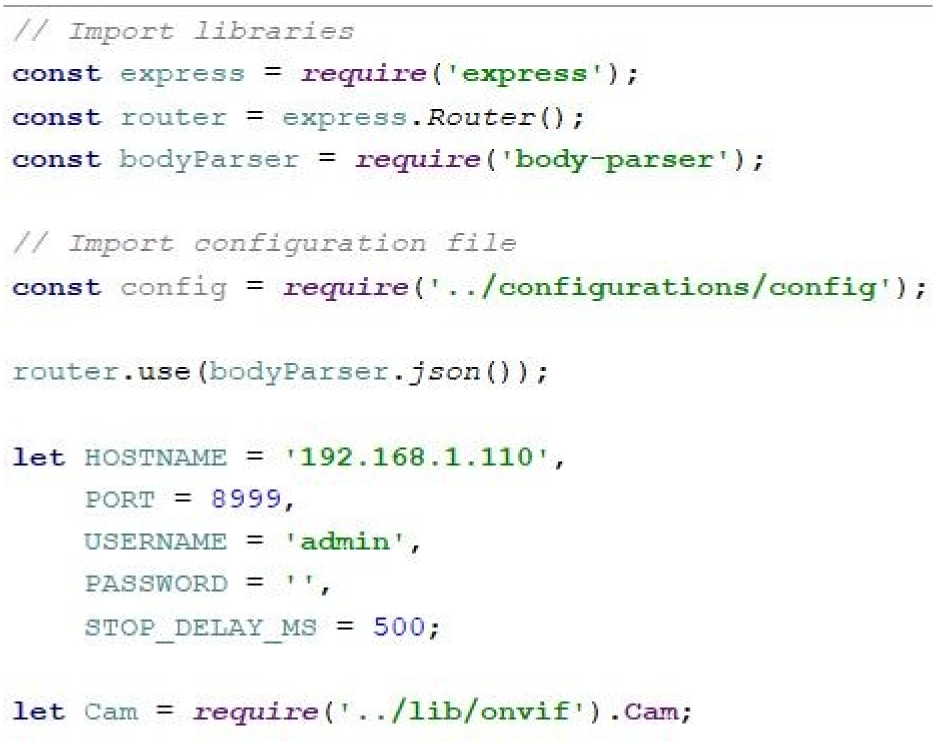
Recently, most of network cameras for business support the ONVIF standard. Furthermore, some network cameras for home support it though the implementation is partial. The node-onvif allows you to control network cameras which implement the ONVIF standard.

With this there are separate commands to move the camera movement left, right, up or down at selected speeds and can freeze movement according to the requirements.



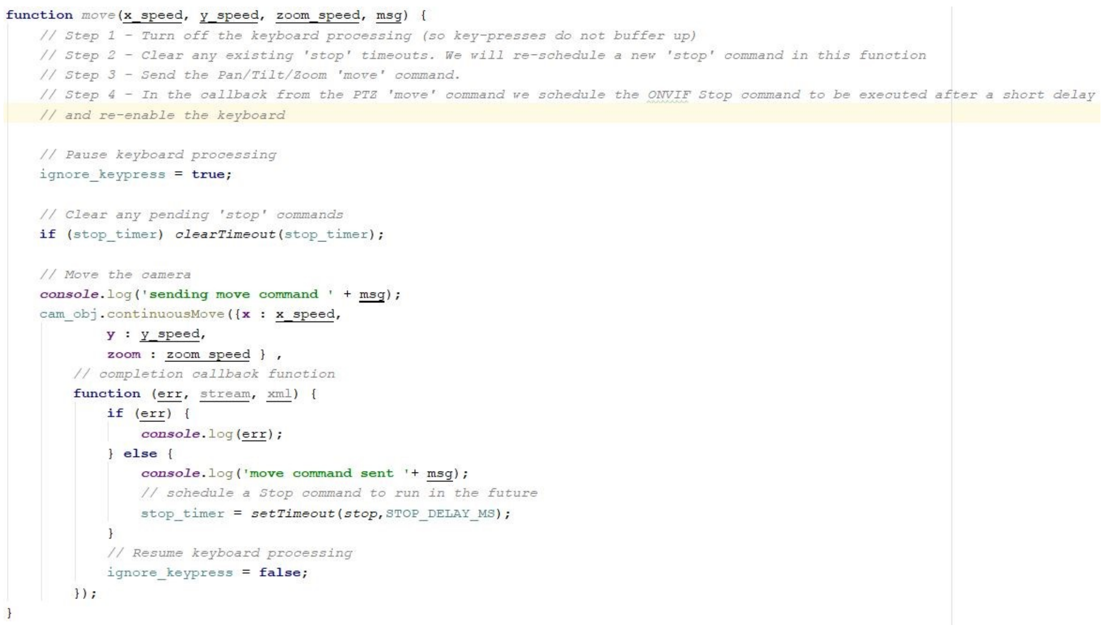
```
new Cam({
    hostname : HOSTNAME,
    username : USERNAME,
    password : PASSWORD,
    port : PORT,
    timeout : 10000
}, function CamFunc(err) {
    if (err) {
        console.log(err);
        return;
    }

    var cam_obj = this;
    var stop_timer;
    var ignore_keypress = false;
    var preset_names = [];
    var preset_tokens = [];

    cam_obj.getStreamUri({
            protocol : 'RTSP'
        },  // Completion callback function
        // This callback is executed once we have a StreamUri
        function (err, stream, xml) {
            if (err) {
                console.log(err);
                return;
            }
```

Figure 12: Get Stream URL of the IP Camera



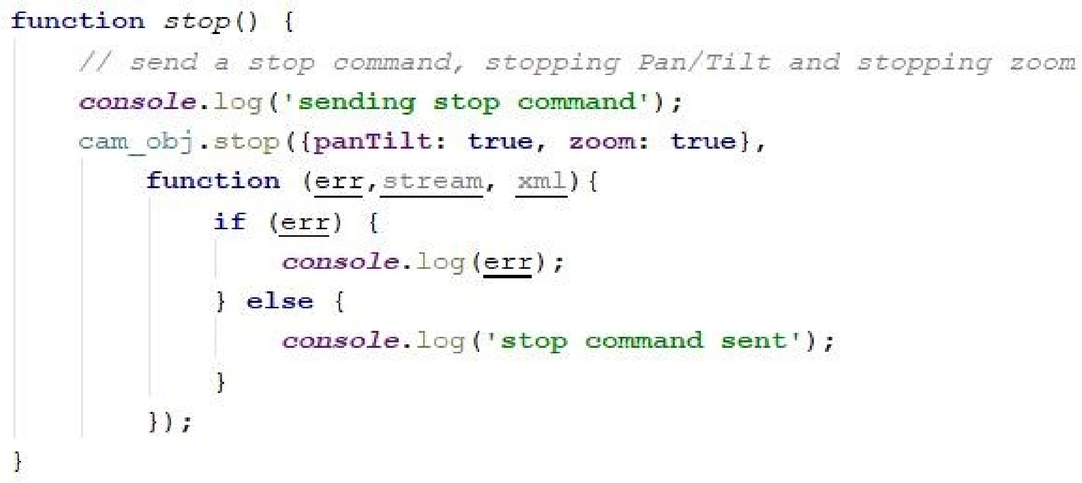
```
// Import libraries
const express = require('express');
const router = express.Router();
const bodyParser = require('body-parser');

// Import configuration file
const config = require('../configurations/config');

router.use(bodyParser.json());

let HOSTNAME = '192.168.1.110',
    PORT = 8999,
    USERNAME = 'admin',
    PASSWORD = '',
    STOP_DELAY_MS = 500;

let Cam = require('../lib/onvif').Cam;
```

Figure 13: Connect to IP Camera



```
function move(x_speed, y_speed, zoom_speed, msg) {
    // Step 1 - Turn off the keyboard processing (so key-presses do not buffer up)
    // Step 2 - Clear any existing 'stop' timeouts. We will re-schedule a new 'stop' command in this function
    // Step 3 - Send the Pan/Tilt/Zoom 'move' command.
    // Step 4 - In the callback from the PTZ 'move' command we schedule the ONVIF Stop command to be executed after a short delay
    // and re-enable the keyboard

    // Pause keyboard processing
    ignore_keypress = true;

    // Clear any pending 'stop' commands
    if (stop_timer) clearTimeout(stop_timer);

    // Move the camera
    console.log('sending move command ' + msg);
    cam_obj.continuousMove({x : x_speed,
            y : y_speed,
            zoom : zoom_speed } ,
        // completion callback function
        function (err, stream, xml) {
            if (err) {
                console.log(err);
            } else {
                console.log('move command sent '+ msg);
                // schedule a Stop command to run in the future
                stop_timer = setTimeout(stop,STOP_DELAY_MS);
            }
            // Resume keyboard processing
            ignore_keypress = false;
        });
}
```

Figure 14: Camera Movement Function



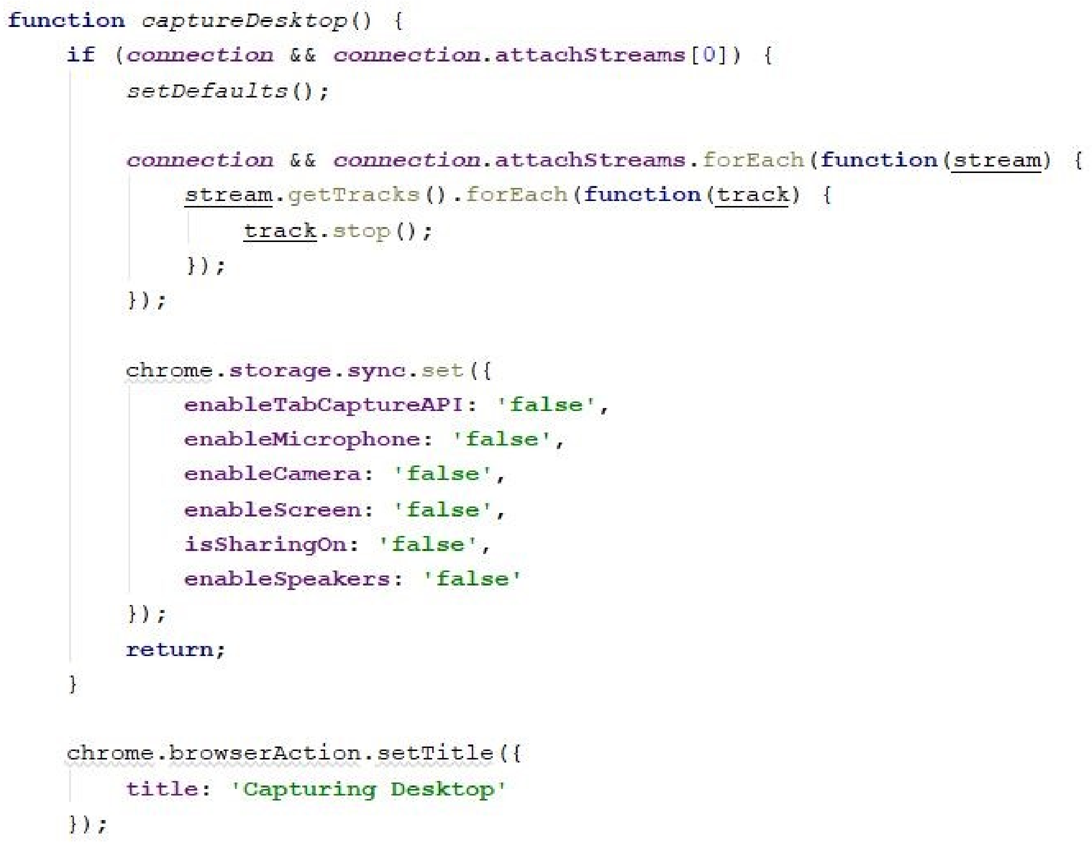
```
function stop() {
    // send a stop command, stopping Pan/Tilt and stopping zoom
    console.log('sending stop command');
    cam_obj.stop({panTilt: true, zoom: true},
        function (err,stream, xml){
            if (err) {
                console.log(err);
            } else {
                console.log('stop command sent');
            }
        });
}
```

Figure 15: Stop Camera Movement

2.3.6 Easy Screen Share Browser Extension

With the ability to share the screen either completed or selected custom application window right from the web browser and start streaming it along the main live stream makes the lectures task at ease and more efficient. This is helpful because most of the times when the lecturer starts to plug his laptop to the main projector in a normal classroom all of his desktop content open tabs on web browser, everything is visible to the students, privacy is a concern and it’s a hassle to switch sharing the screen on and off all the time to the lecturer. With this Easy Screen Share feature lecturer can stream his webcam footage alongside with the shared screen if required (If in front of the laptop blocking the main camera view).

This extension simply initializes socket.io and configures it in a way that single audio/video/screen stream can be shared/relayed over users without any bandwidth/CPU usage issues. This uses RTCMultiConnection is a WebRTC library that is used for WebRTC streaming [31].



```
function captureDesktop() {
    if (connection && connection.attachStreams[0]) {
        setDefaults();

        connection && connection.attachStreams.forEach(function(stream) {
            stream.getTracks().forEach(function(track) {
                track.stop();
            });
        });

        chrome.storage.sync.set({
            enableTabCaptureAPI: 'false',
            enableMicrophone: 'false',
            enableCamera: 'false',
            enableScreen: 'false',
            isSharingOn: 'false',
            enableSpeakers: 'false'
        });
        return;
    }

    chrome.browserAction.setTitle({
        title: 'Capturing Desktop'
    });
```

Figure 16: Desktop Capture Initialization

2.3.7 Gesture based camera control

When it comes to learning, there are obvious moments of direct questioning from the lecturer, by the members of the audience. In such situations, although the crowd present in the classroom can see the person who is asking the question, the remote users will have no clue, unless the person is focused by the camera. For this purpose, when a student indicates that he/she has a doubt, the lecturer will perform a predefined gesture towards the camera which will recognize it and turn towards the audience. The main objective of having gesture-based recognition is ensure remote users get the feeling of being in a real classroom, other than getting the feeling of watching a conventional video.

Gesture recognition can be addressed as a challenging task to computer vision due to the segmentation of the foreground object (in this case, the hand gesture), from a cluttered background. Here, the most obvious reason is because of the semantic gap involves when a human looks at an image, and when a computer or a camera looks at the same image. The human eye can easily identify separately the components of an image, but, a computer or a camera sees an image as a 3-Dimensional matrix. In this scenario, it is necessary to mimic mammalian vision using a PTZ camera to recognize a hand gesture performed by the lecturer in order to turn it towards the audience by 180^{o}.

Currently, there are two types of approaches in gesture recognition that are used in computer vision. They are namely; "Data-Glove based" and "Vision based" approaches. The first approach uses sensor devices for digitizing hand and finger motions into multi-parametric data. The extra sensors make it easy to collect hand configuration and movement. In vision-based gesture recognition, only a camera is required to capture a gesture [32].

Because "Data-glove based" approach is complex and requires more sophisticated equipment, using this method would be cumbersome especially when it comes to the lecture capturing system. In this system, "Vision based" gesture capturing approach is used as it is the most suitable way to overcome the problem we are trying to address in this thesis document.

Under "Vision based" gesture capturing, there are two subcategories to be considered.

1. 3D hand model-based approaches
2. Appearance based approach

In this solution, appearance-based approach is mainly used as it is less complex, easier to handle, efficient. The two main technologies used here are OpenCV and Python.

The methodology can be split into two major steps:

1. Identification and segmentation of the hand region from the video sequence.
2. Counting the number of fingers from the segmented hand region.

Identification and segmentation of the hand region from the video sequence.

The hand region can be referred to as the foreground component and everything else exist in the background of the video sequence considered. This is done by running averages for 30 frames of the video sequence. In that way, the system will know the background of the video, and anything else which comes into it later on will be considered as the foreground object/s. Here, attention should be paid to the way of analyzing the foreground object/s alone and recognizing what a hand object is.

Background subtraction is the method that is used to separately identify the hand gesture from the background. After the background is identified through running averages, the absolute difference between the background and the current frame (which is updated over time) is calculated. This results in an image which holds the newly added foreground image, which is the hand. The resulting image is known as the difference image.

Detecting the hand region from this difference image is done by applying motion detection and thresholding. Here, only the hand region is visible, while all other unwanted regions are painted in black.

As an example, here is how thresholding is applied to a hand showing 5 fingers towards the camera.

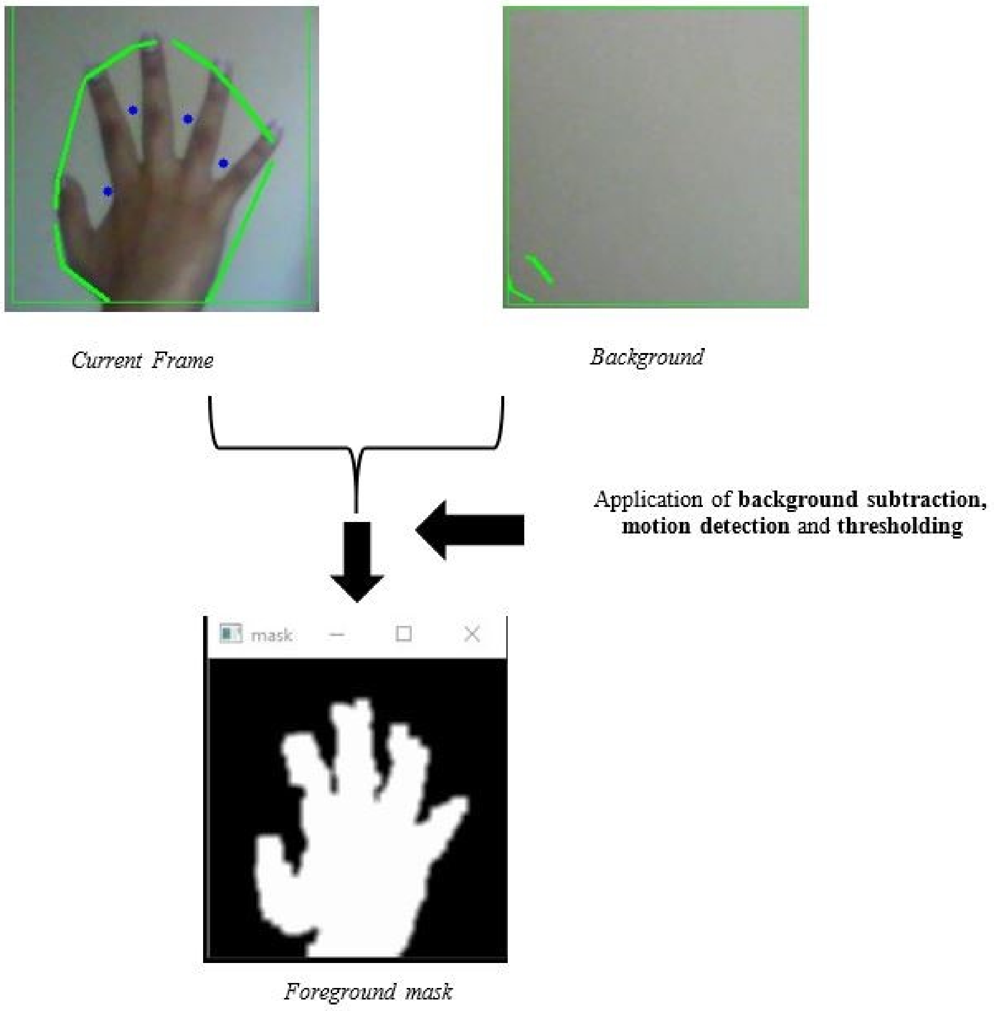

Current Frame
Background
Application of background subtraction, motion detection and thresholding
mask
Foreground mask

Figure 17: Application of background subtraction, motion detection and thresholding to the background and current frame

The next step in detecting the hand area is contour extraction. Contour is the outline boundary of an object located in an image.

Counting the number of fingers from the segmented hand region

The human hand is a complex anatomical structure consisting of many connected parts and joints, involving complex relations between them, providing a total of roughly 27 degrees of freedom.

In the figure above, after identifying the skin color pixels, a convex hull is drawn around the region that was identified as the hand in green color, after which the number of defects is calculated. Defects refer to the number of angles that are formed in between the fingers. This way, the algorithm calculates the number of fingers held towards the camera using the following formula.

Number of fingers = Number of defects + 1

In the above example, I have taken 5 fingers. The blue dots indicate the number of defects formed within the convex hull/ hand area. i.e. 4 defects.

The above algorithm is developed to recognize any number of fingers from 1 to 5 according to convenience.

2.3.8 Facial Recognition based user login

A student or a lecturer can login to the system using the webcam. Initially, the administrator of the Lecture Capturing System should register the user by uploading quality images and relevant details of the user. After registering a user, the server will train the face recognition classifier with the newly uploaded images of a user along with the existing images of users. Thereafter the user will be authenticated from the face recognition process through the webcam only if the confidence threshold of the face recognition classifier is greater than 90%. If it is less than 90%, the user will not be authenticated. In terms of security, session handling will take place after face recognition-based login.

In terms of the general architecture of the face recognition-based login function, two steps are considered

1. Detection stage

LCS system search for the face region (displayed by rectangle) in the whole video stream

2. Recognition stage

Contrasting the face image obtained above to the face image trained in the database, and predicting the user registered.

If the system face recognition is successful, the recognition result will be displayed in white text inside a green rectangle on the webcam feed along with the confidence percentage. If failed, system will pop a warning. Face Recognition used in the system follows three main steps.

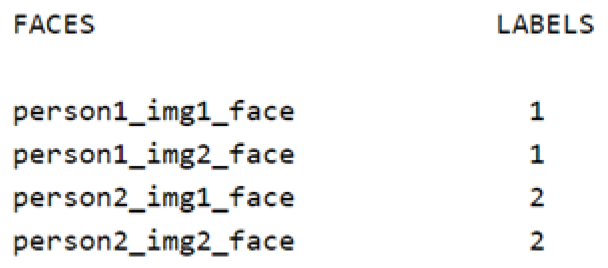
1. Prepare training data

OpenCV computer vision library, Python and Numpy is used as dependencies to implement face recognition function in the system [27]. OpenCV provides two pre-trained and ready to be used face detection classifiers called Haar classifier and LBP classifier [33]. Haar Cascade classifier is used as the face recognition classifier and Local Binary Patterns (LBP) classifier is used as the face recognition classifier to detect and recognize faces in this system. LBP is a type of visual descriptor used for classification in computer vision. The LBP classifier is used due to its main advantages such as shorter training time, high accuracy rate in difficult lighting conditions which will be useful when detecting faces through the webcam and computationally simple and fast [33]. A formal description of the LBP algorithm can be given as follows.

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c)$$

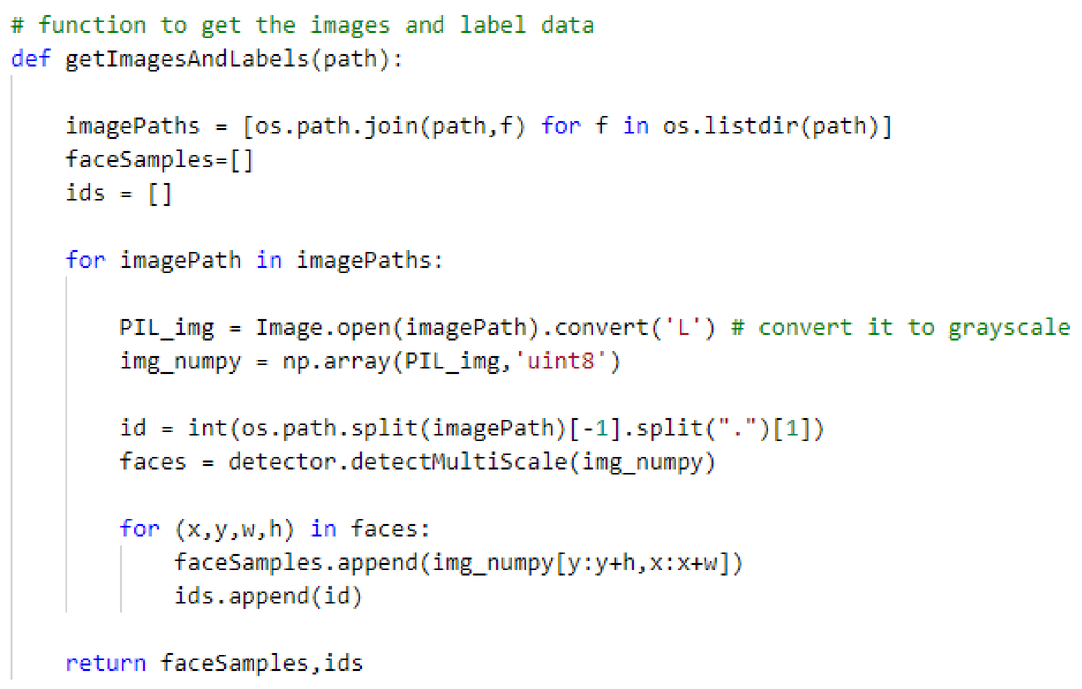
Figure 18: LBP algorithm

The training dataset consists 30 images for each user and each user is assigned a label (e. g. s1, s2) upon registering to the system. Furthermore, this step will read all the images of a person and apply face detection to each one using LBP classifier. Then, add each face to face vectors with the corresponding person label extracted. Finally, the data preparation step will produce following face and label vectors [33].



FACES	LABELS
person1_img1_face	1
person1_img2_face	1
person2_img1_face	2
person2_img2_face	2

Figure 19: Face and label vectors

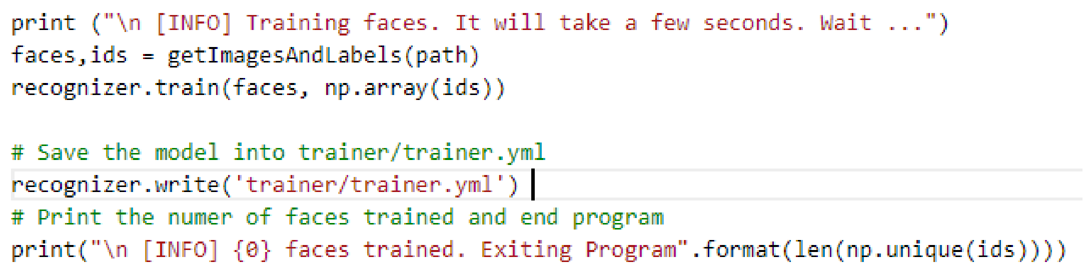


```
# function to get the images and label data
def getImagesAndLabels(path):

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []

    for imagePath in imagePaths:

        PIL_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')

        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)

        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)

    return faceSamples,ids
```

Figure 20: Prepare Training data function

2. Train face recognizer

The face and label vectors returned from the data preparation step (according to Figure 20 'getImagesAndLabels' function) will be converted to a Numpy array and passed to the OpenCV Haar Cascade face recognizer for training [33]. The statistical data returned from the face recognizer will be saved in a YAML file.

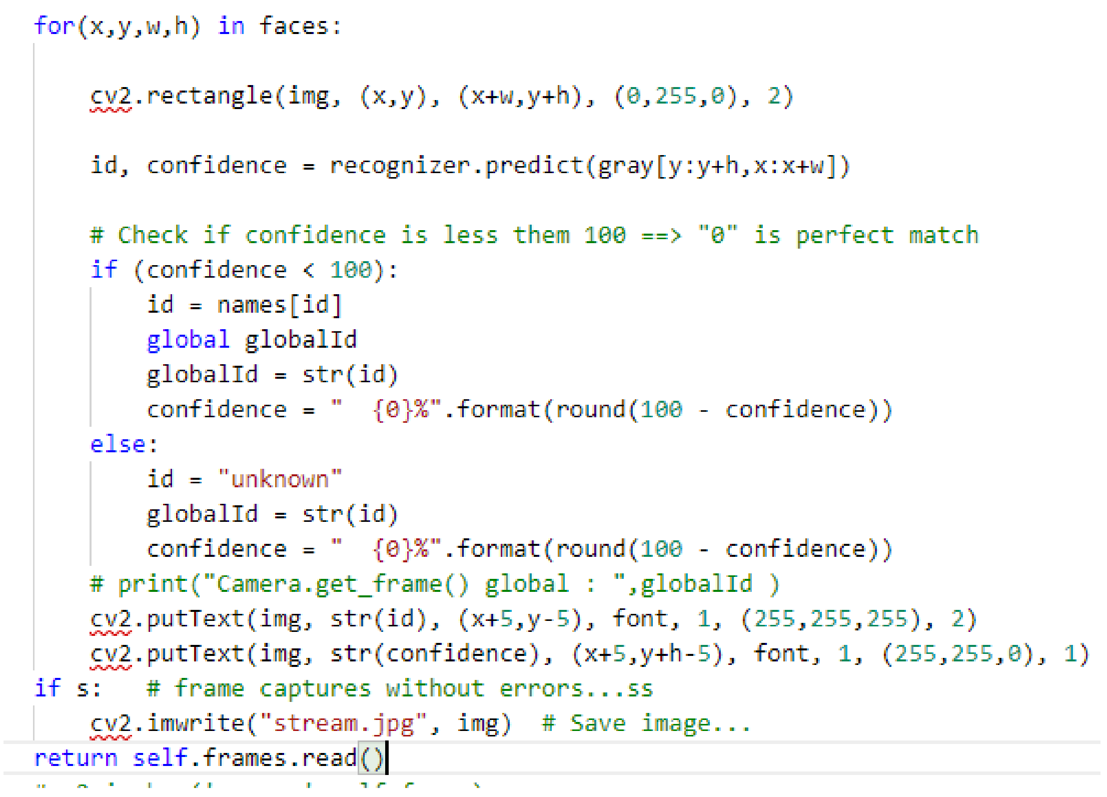


```
print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))

# Save the model into trainer/trainer.yml
recognizer.write('trainer/trainer.yml')
# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained. Exiting Program".format(len(np.unique(ids))))
```

Figure 21: Haar Cascade Recognizer function

3. Prediction

Once the user is navigated to the login page, the server automatically detects the face from the Haar classifier, predicts the face by calling the trained OpenCV Haar face recognizer, returns the predicted name of the user associated with the label and live streams the response from the server (recognized face plot, name of the user, confidence threshold) to the login page. Thereafter, the user will get logged in to the system after the user clicks the face login button.



```
for(x,y,w,h) in faces:

    cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

    id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

    # Check if confidence is less them 100 ==> "0" is perfect match
    if (confidence < 100):
        id = names[id]
        global globalId
        globalId = str(id)
        confidence = "  {0}%".format(round(100 - confidence))
    else:
        id = "unknown"
        globalId = str(id)
        confidence = "  {0}%".format(round(100 - confidence))
    # print("Camera.get_frame() global : ",globalId )
    cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255), 2)
    cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)
if s:   # frame captures without errors...ss
    cv2.imwrite("stream.jpg", img)  # Save image...
return self.frames.read()
```

Figure 22: Prediction Implementation

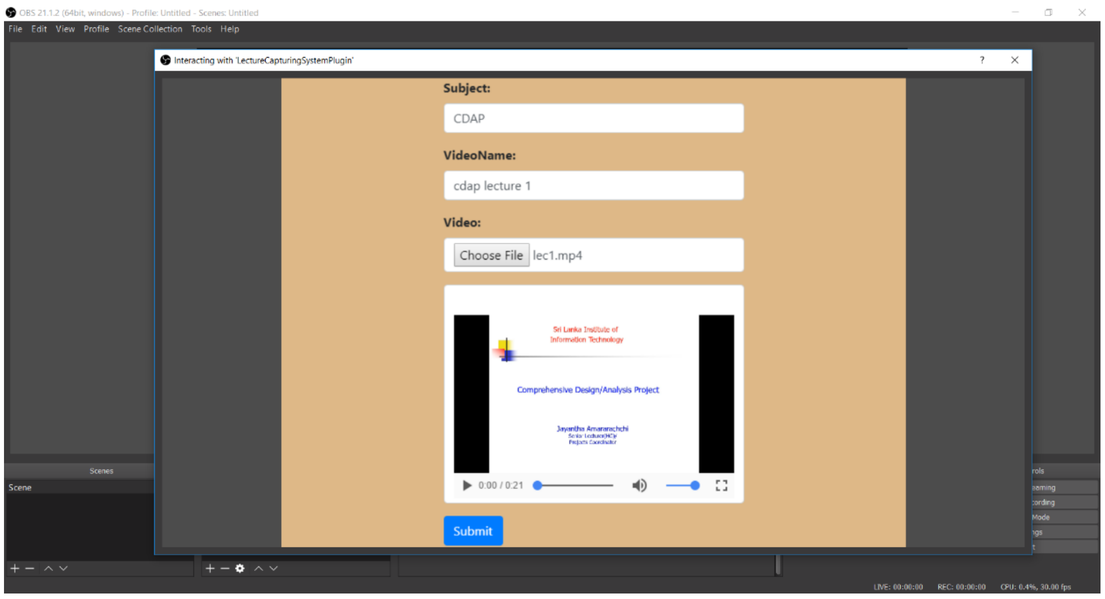
2.3.9 Automated Facial Recognition based Attendance Marking.

Using facial recognition, the attendance is marked automatically for the students who are present in the lecture room and also the students who are logged in remotely through the Lecture Capturing System during the live streaming lecture session. The administrator and the lecturer are able to view, modify and filter attendance of students. A student is able to view his/her attendance with the aid of the filtering options available. Some noticeable advantages of this feature is that it will add an extra layer of security to the system to ensure that only authorized persons gain access to the university's content. A comprehensible advantage of this method of biometric authentication of students can be noted during the time of an online exam to verify that the person on the other end is actually who they claim to be. Also, this feature will solve the problem of students marking attendance for other students.

2.3.10 Open Broadcaster Software (OBS) Studio plugin

During the lecture session, the desktop screen of the lecturer's computer as well as the lecturer's voice will be recorded from beginning to end of the lecture session. This will be achieved by using a free and open-source video recording software known as Open Broadcaster Software (OBS) Studio [24]. This also applies to an offline remote recording of a lecture session. For example, a lecturer can do a lecture recording from the comfort of their home.

A plugin was implemented as shown in Figure 23 for the OBS Studio software which allows the lecturer to upload the recorded video directly to the server for further processing.


OBS 21.1.2 (64bit, windows) - Profile: Untitled - Scenes: Untitled
File Edit View Profile Scene Collection Tools Help
Interacting with 'LectureCapturingSystemPlugin'
Scenes
Scene
Choose File
Submit
LIVE: 00:00:00 REC: 00:00:00 CPU: 0.4%, 30.00 fps

Subject: CDAP

VideoName: cdap lecture 1

Video: lec1.mp4

Figure 23: OBS Studio Plugin Main Interface

The video will then be saved in Flash Video (FLV) format which is a container for an h.264/AVC video track and an Advanced Audio Coding (AAC) audio track. H.264 is an industry standard for video compression; the process of converting digital video into a format that takes up less capacity when it is stored or transmitted. An encoder converts video into a compressed format and a decoder converts compressed video back into an uncompressed format [34]. AAC is a technique used for compressing and encoding scheme digital audio files. AAC technology is used for coding audio files at medium to high bit rates [35]. The FLV file format is used because this container is designed to be started and stopped at any time, resulting in a stable consistent output video file [36].After recording the lecturer's desktop screen while s/he conducts a lecture, the designed plugin would upload this video to the remote server as shown in Figure 24 based on predefined settings at the click of a button. These settings can be changed by the lecturer to suit their needs (e.g. upload video now or at a later scheduled time). Once the video is uploaded to the remote server, the next level of processing would be done at the server.

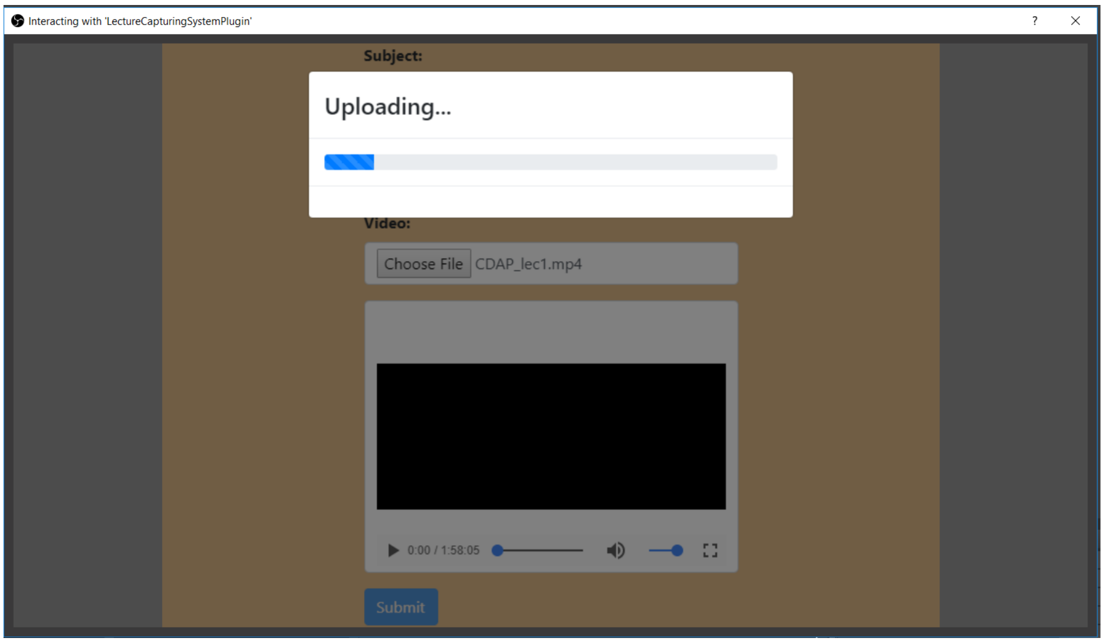


Figure 24: Video Uploading Process

2.3.11 Video Thumbnails/Chapters Creation

The lecturer can view the list of videos which have been uploaded to the server. Out of this list, the lecturer can select a video to be converted into a series of thumbnail chapters.

Each frame in the video is analyzed by the PySceneDetect algorithm which is implemented using Python. The PySceneDetect algorithm makes use of the OpenCV, NumPy, and FFmpeg libraries for execution. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library of programming functions [37]. It is used for the video analysis process. NumPy is a library for the Python programming language, which adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. It is used as the multi-dimensional container for the generic arbitrary data types (the video frames). FFmpeg is a suite of libraries and programs for handling video, audio, and other multimedia files and streams [38].

There are two main detection methods which PySceneDetect uses namely threshold-detection and content-aware detection. Each mode has slightly different parameters, and is described in detail below:

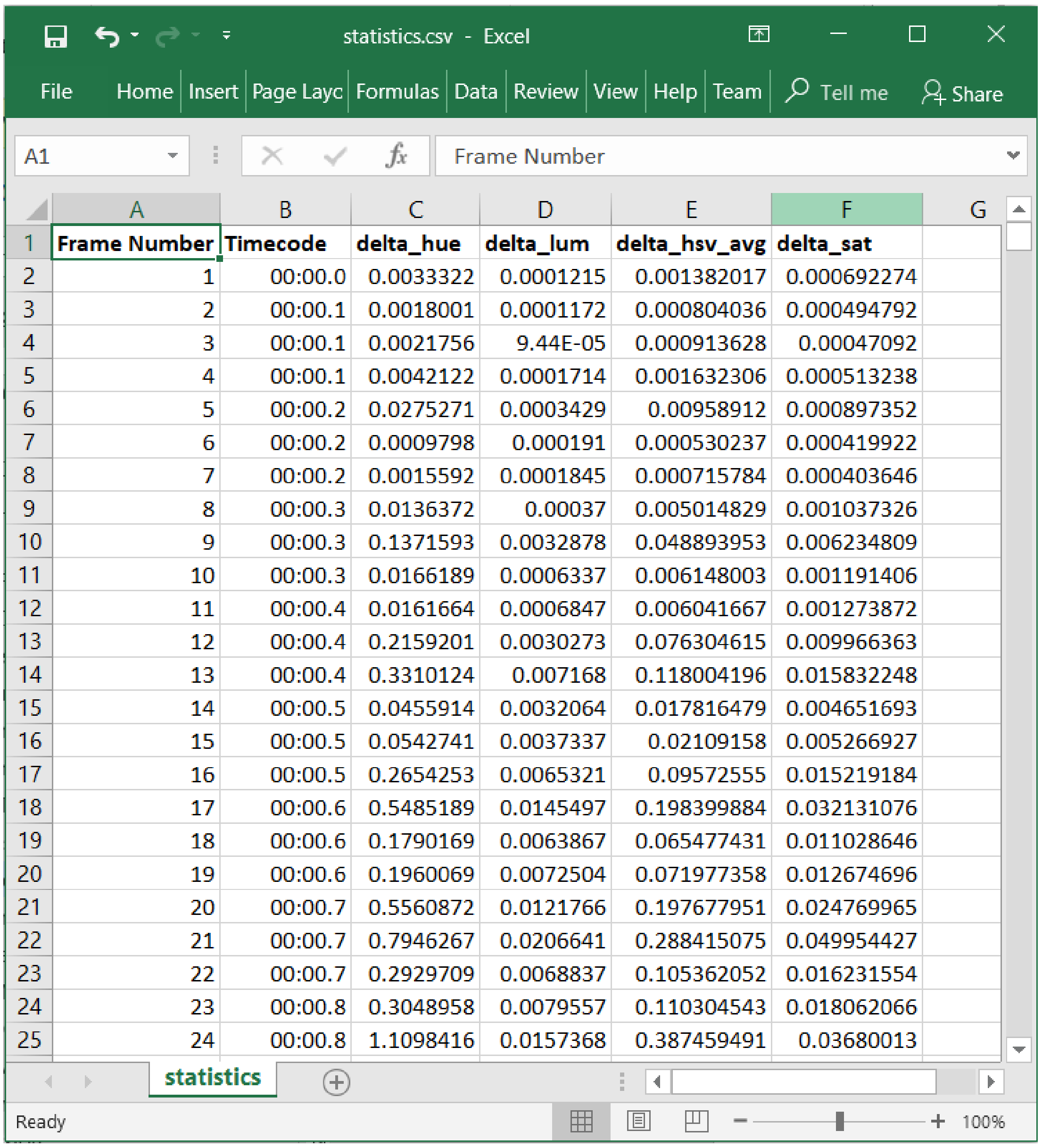
- Threshold-detection - Compares the intensity/brightness of the current video frame with a set threshold, and triggers a scene cut/break when this value crosses the threshold value. The threshold value is computed by averaging the Red-Green-Blue (RGB) values for every pixel in the frame, yielding a single floating-point number representing the average pixel value (from 0.0 to 255.0).
- Content-aware detection - Finds areas where the difference between two subsequent video frames exceeds the threshold value that is set and then trigger a scene cut. This allows you to detect cuts between scenes both containing content, rather than how most traditional scene detection methods work. With a properly set threshold, this method can even detect minor, abrupt changes [39]. This method takes in the threshold and minimum-scene-length in frames (optional) as input parameters.

Table 6 shows the complete list of various input parameters which can be specified and their usage when conducting the video-splitting process [40].

Table 6: Input Parameters for PySceneDetect

Parameter	Usage
-i	Input video filename.
-o	Output video filename.
-d	Detection method (detect-threshold / detect-content).
-s / --stats	Generate a statistics file.
-t / --threshold	Threshold value.
list-scenes	Exports the list of scenes to a .CSV file.
save-images	Saves a given number of frames from every detected scene as images, by default JPEG.
split-video	Split the input video automatically.
-q / --quiet	Suppresses debug console logs output printed to the terminal.
--start	Time in video to begin detecting scenes.
--duration	Maximum time in video to process.
--end	Time in video to end detecting scenes.

In order to calculate the threshold value, a .CSV file was generated by using the **--stats (-s)** parameter such as the one depicted by Figure 25.

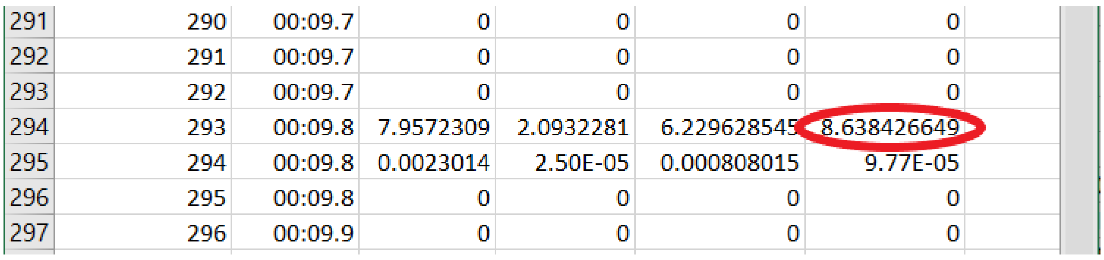


Frame Number	Timecode	delta_hue	delta_lum	delta_hsv_avg	delta_sat
1	00:00.0	0.0033322	0.0001215	0.001382017	0.000692274
2	00:00.1	0.0018001	0.0001172	0.000804036	0.000494792
3	00:00.1	0.0021756	9.44E-05	0.000913628	0.00047092
4	00:00.1	0.0042122	0.0001714	0.001632306	0.000513238
5	00:00.2	0.0275271	0.0003429	0.00958912	0.000897352
6	00:00.2	0.0009798	0.000191	0.000530237	0.000419922
7	00:00.2	0.0015592	0.0001845	0.000715784	0.000403646
8	00:00.3	0.0136372	0.00037	0.005014829	0.001037326
9	00:00.3	0.1371593	0.0032878	0.048893953	0.006234809
10	00:00.3	0.0166189	0.0006337	0.006148003	0.001191406
11	00:00.4	0.0161664	0.0006847	0.006041667	0.001273872
12	00:00.4	0.2159201	0.0030273	0.076304615	0.009966363
13	00:00.4	0.3310124	0.007168	0.118004196	0.015832248
14	00:00.5	0.0455914	0.0032064	0.017816479	0.004651693
15	00:00.5	0.0542741	0.0037337	0.02109158	0.005266927
16	00:00.5	0.2654253	0.0065321	0.09572555	0.015219184
17	00:00.6	0.5485189	0.0145497	0.198399884	0.032131076
18	00:00.6	0.1790169	0.0063867	0.065477431	0.011028646
19	00:00.6	0.1960069	0.0072504	0.071977358	0.012674696
20	00:00.7	0.5560872	0.0121766	0.197677951	0.024769965
21	00:00.7	0.7946267	0.0206641	0.288415075	0.049954427
22	00:00.7	0.2929709	0.0068837	0.105362052	0.016231554
23	00:00.8	0.3048958	0.0079557	0.110304543	0.018062066
24	00:00.8	1.1098416	0.0157368	0.387459491	0.03680013

Figure 25: Generated Statistics File

This gives statistics/analysis of frame-by-frame video metrics such as frame number, timecode, hue, saturation, luminance, and HSV average which assist in determining the correct input threshold parameter.

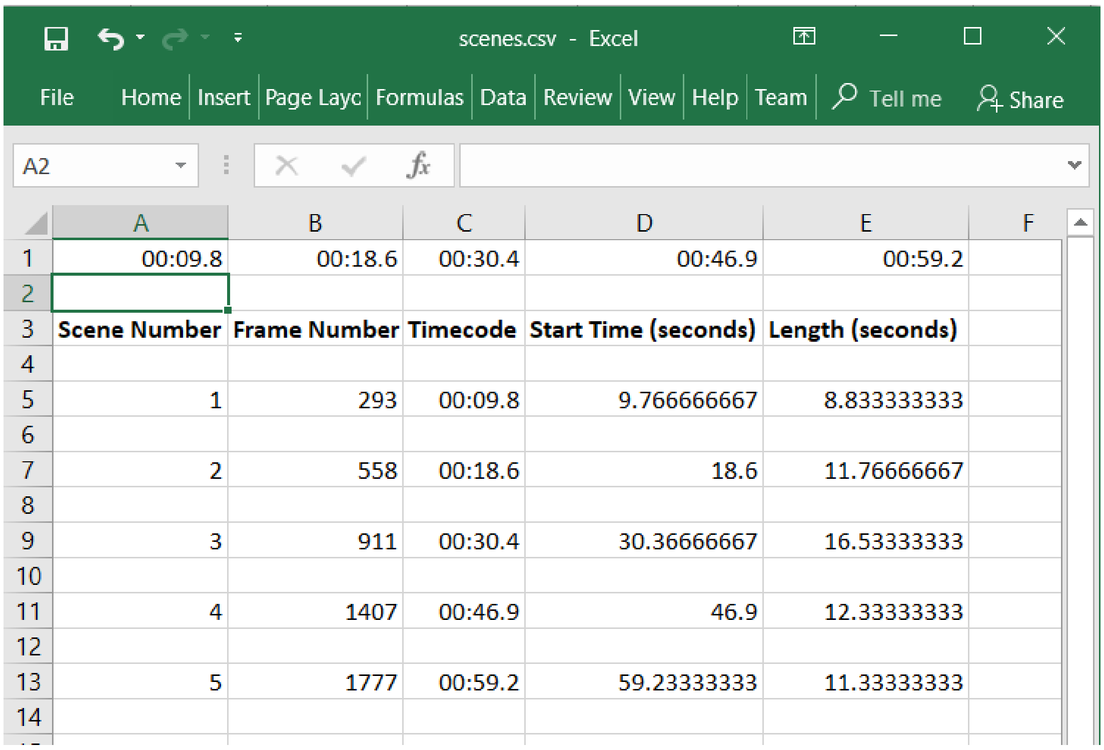
Due to the better performance of the content-aware detection method, it was used for the video splitting process. It compares the difference in content between adjacent frames against a set threshold/score, which if exceeded, triggers a scene cut. It checks for changes in color and intensity - namely the average HSV color space difference (difference in hue, saturation, and luminance of the frame) – between video frames [41]. If this calculated value is very high than the preceding and following values, it means that there has been a scene change as shown in Figure 26.



291	290	00:09.7	0	0	0	0
292	291	00:09.7	0	0	0	0
293	292	00:09.7	0	0	0	0
294	293	00:09.8	7.9572309	2.0932281	6.229628545	8.638426649
295	294	00:09.8	0.0023014	2.50E-05	0.000808015	9.77E-05
296	295	00:09.8	0	0	0	0
297	296	00:09.9	0	0	0	0

Figure 26: Detecting a Scene Change from Statistics

Therefore, the video is split at this time frame. For Figure 26, the video split occurs at 00:09.8. This process is repeated for the entire length of the video clip until the entire video clip is analyzed and all the video chapters are created. If required, specifying the input parameter **list-scenes** also exports the list of scenes to a .CSV file with frame numbers, timecodes, start times, and length of each scene as shown in Figure 27. This gives us a more detailed knowledge about the video chapters created.



	A	B	C	D	E	F
1	00:09.8	00:18.6	00:30.4	00:46.9	00:59.2	
2						
3	Scene Number	Frame Number	Timecode	Start Time (seconds)	Length (seconds)	
4						
5	1	293	00:09.8	9.766666667	8.833333333	
6						
7	2	558	00:18.6	18.6	11.76666667	
8						
9	3	911	00:30.4	30.36666667	16.53333333	
10						
11	4	1407	00:46.9	46.9	12.33333333	
12						
13	5	1777	00:59.2	59.23333333	11.33333333	
14						

Figure 27: List of Video Chapters (Scenes/Thumbnails)

Following this process is the real-time speech transcription (audio-to-text conversion) of each video chapter. First, the audio is extracted from the video chapters in MP3 format by the FFmpeg library. Next is the speech transcription procedure which is achieved via the Watson Speech-To-Text algorithm. This service leverages machine intelligence to transcribe the human voice accurately [42]. The service combines information about grammar and language structure with knowledge of the composition of the audio signal. resulting in a remarkable average accuracy level of 88.3%.

Therefore, the end result would be a set of videos along with their respective audio, presentation slide, and text. These videos would be stored in the database so that students can access them any time after the lecture session to further understand and clarify their knowledge Some noticeable advantages of this feature are that students can watch the parts of the lecture which they didn't understand properly, and also skip to various lecture slides instead of watching the entire lecture. Even if the accent of the lecturer is unclear, students can still understand what has been said clearly due to the speech transcription feature.

2.3.12 Course Management

The administrator can manage courses conducted at the university. When a new course is introduced to the university, the administrator can add it to the system. If any changes are proposed to an existing course (e.g. change the number of credits in the course), the administrator is able to update its details. There would be occasions when an existing course learning material gets outdated; in this case, the administrator can delete this course from the system.

2.3.13 Quota Management

The administrator is able to manage the internet quota allocation for users from the dashboard. The list of users along with their usage statistics such as used quota and remaining quota can be viewed filtered by user type (e.g. lecturer, student), month, and year. This monthly quota can be edited by the administrator for a single user (e.g. specific lecturer's id) or all users of a particular user type (e.g. all students).

2.3.14 Bandwidth Management

The data size which is passed from client to the node server and vice versa, is reduced using bandwidth optimization techniques such as compression and clustering. The administrator can monitor bandwidth using the bandwidth monitoring dashboard which consist of traffic usage, system information, CPU load, alerts to notify exceeded predefined threshold settings and attacks and much more that is accessible only to the administrator of the system.

2.4 Research Findings

In face recognition function, converting the RGB images to gray scale or normalizing the images will increase the accuracy of the face detection and recognition processes.

Suppose a user contains 12 face images which is trained and not in gray scale, the face detection algorithm will detect 40% from all the images. If the images are in gray scale or normalized, the detecting percentage is higher than 90%.

For face detection and recognition, Haar Cascade classifier is the better choice when compared and tested with LBP classifier in terms of accuracy. EigenFaces algorithm was also tested in face detection and recognition functions, However, EigenFaces algorithm considers illumination as an important feature. In consequence, lights and shadows are picked up by EigenFaces, which classifies them as representing a ‘face’. Therefore, EigenFaces algorithm is not suitable for face recognition functions available in the LCS system when false predictions, lighting conditions and security standards are considered.

With regards to the video chapter creation feature, each frame in the video is analyzed by an algorithm for changes in color and intensity - namely the average HSV color space difference (difference in hue, saturation, and luminance of the frame). If this calculated value is very high than the preceding and following values, it means that there has been a scene change. Therefore, the video is split at this time frame. This process is repeated for the entire length of the video clip until the entire video clip is analyzed and all the video chapters are created. This process was run over repeated 100-iteration cycles which produced an average accuracy of 95% which is a very satisfactory value.

Following this process is the real-time speech transcription (audio-to-text conversion) of each video chapter. This is achieved via the Watson Speech-To-Text API. This service leverages machine intelligence to transcribe the human voice accurately. The service combines information about grammar and language structure with knowledge of the composition of the audio signal resulting in a remarkable accuracy level of 88.3%.

3. RESULTS AND DISCUSSION

3.1 Evidence

The following figures show the results achieved from the research project and the new methodologies found to address further research in the undergraduate context.

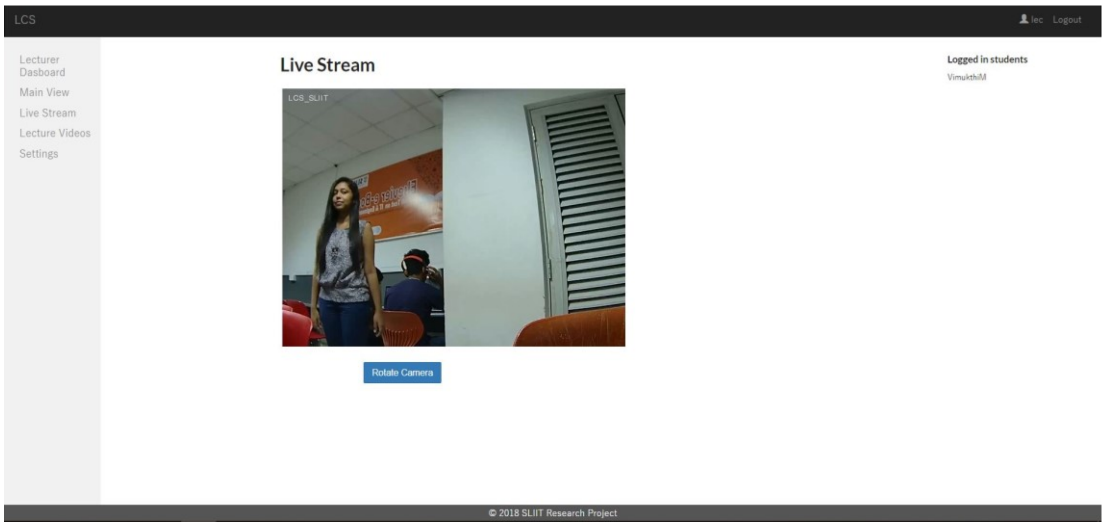


Figure 28: Live Stream Main View

LCS consist of a web application for the lecturer to log in and start live streaming process. Figure 28 illustrates the main view from the IP camera as in real-time. In Figure 29 it shows that if the lecturer chooses he/she can disable the auto tracking features and control everything manually right from the web application if the need arises. This advanced controls view enables lecturer to take command of the camera and do the movement manually if required. Move camera to any direction and to zoom in and zoom out as required. Also, lecturer has the ability to stop any scripts to disable automatic tracking of the lecturer.

Example: A special Question and Answers session with another colleague at a specific location or a scientific experiment on stage.

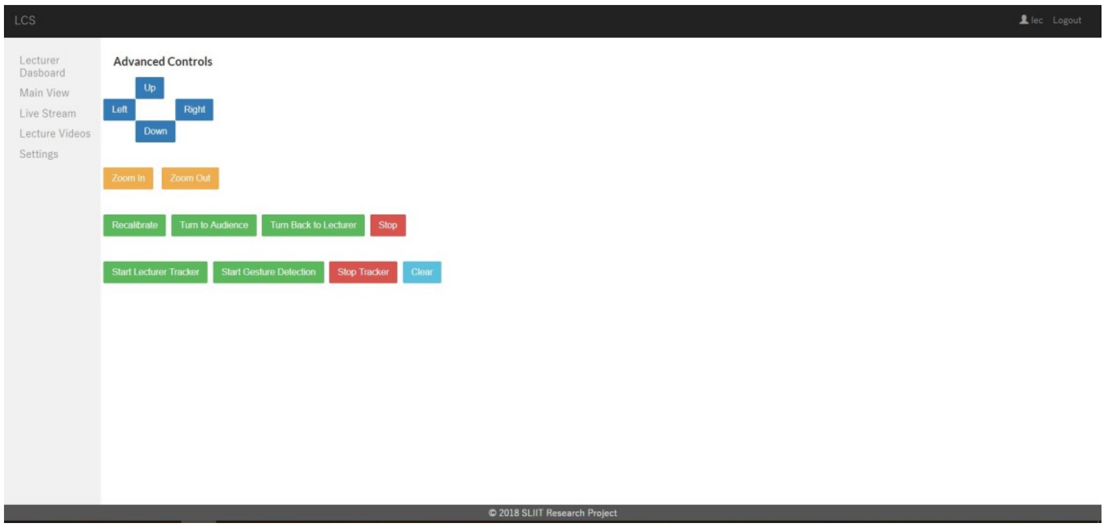


Figure 29: Advanced Controls UI

Figure 30 illustrates the main view of the Screen Sharing extension which enables lecturer to choose whether to share the screen or screen and webcam or just the webcam footage.

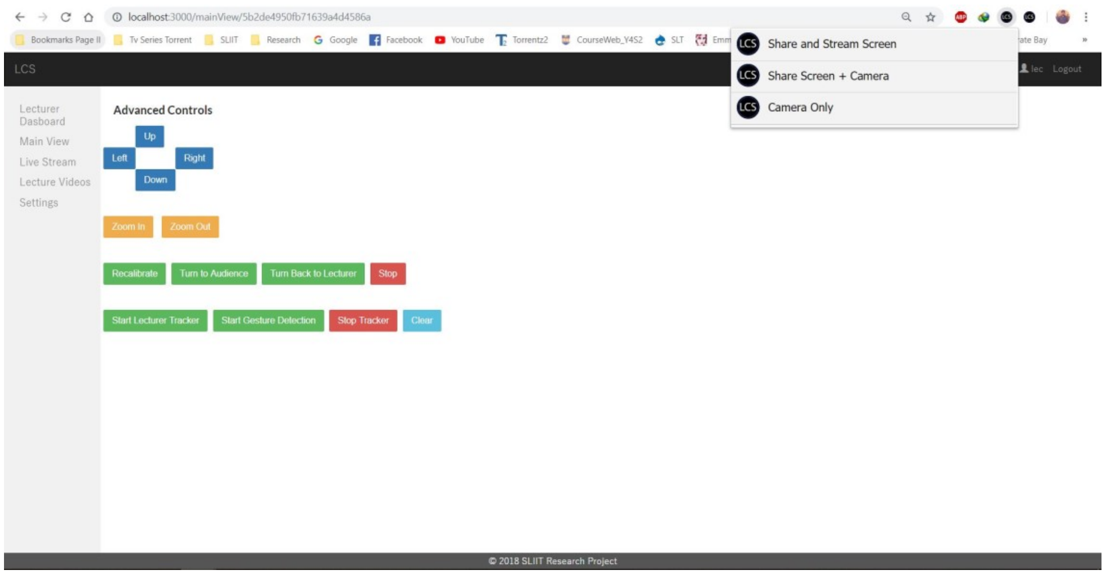
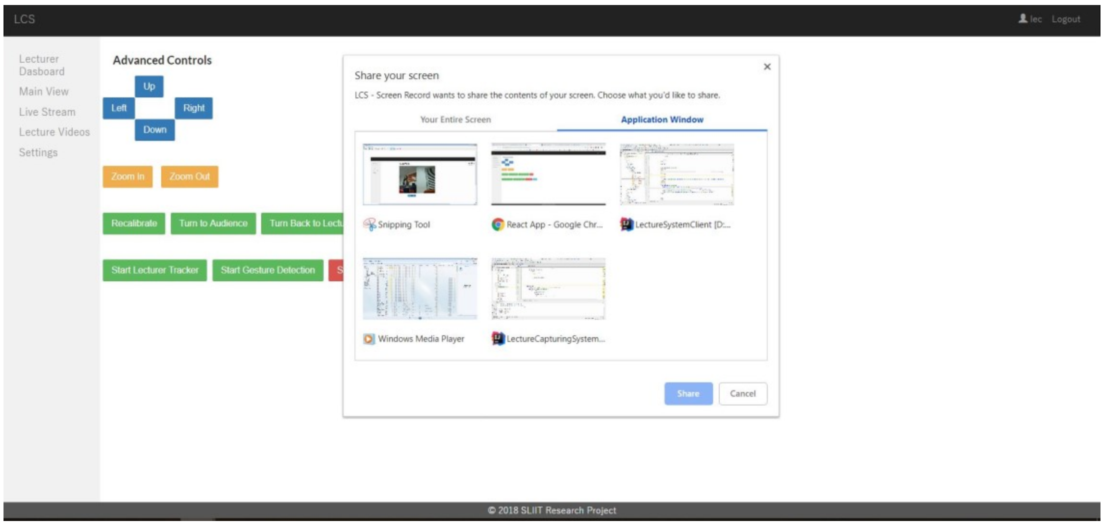


Figure 30: Main View of the extension

With Figure 31 view, lecturer should select the sharing screen, either full screen or one of the specific application window had he/she selected screen sharing. If the lecturer selected full screen sharing everything will be shared from the lecturer's laptop screen including taskbar and all open programs. If selected an Application Window only the selected application will stream to the students, this is really helpful if the lecturer is using his/her own laptop with private files and programs open in the background. Security and Privacy vice this gives lecturer a peace of mind.



LCS - Screen Record wants to share the contents of your screen. Choose what you'd like to share.

Figure 31: Share Screen Window

In order to test our tracking of the lecturer, we use ONVIF Device Manager V2.2.25 provided by Synesis, which is an open source desktop application used to manage ONVIF IP devices.

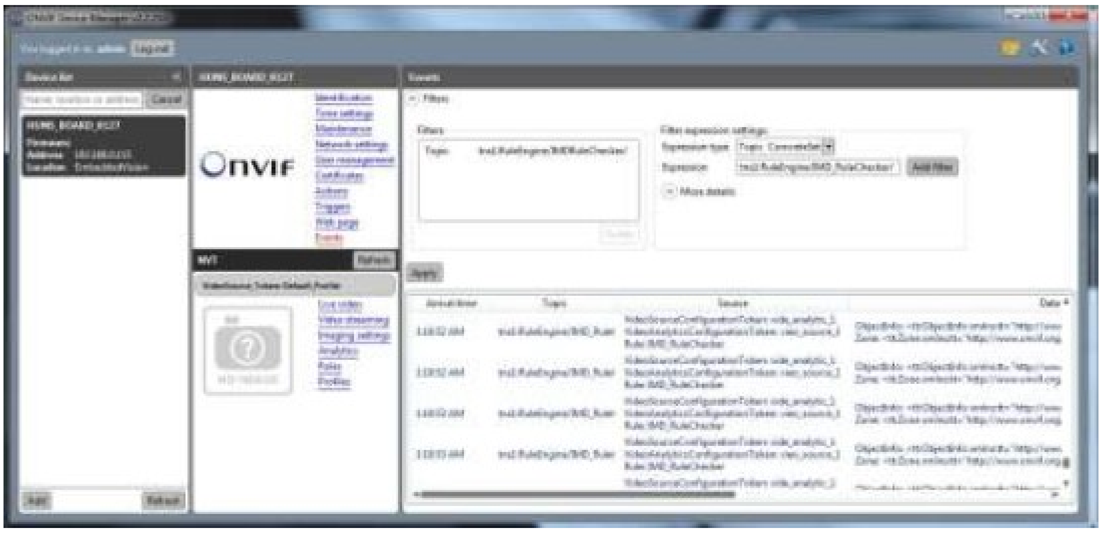
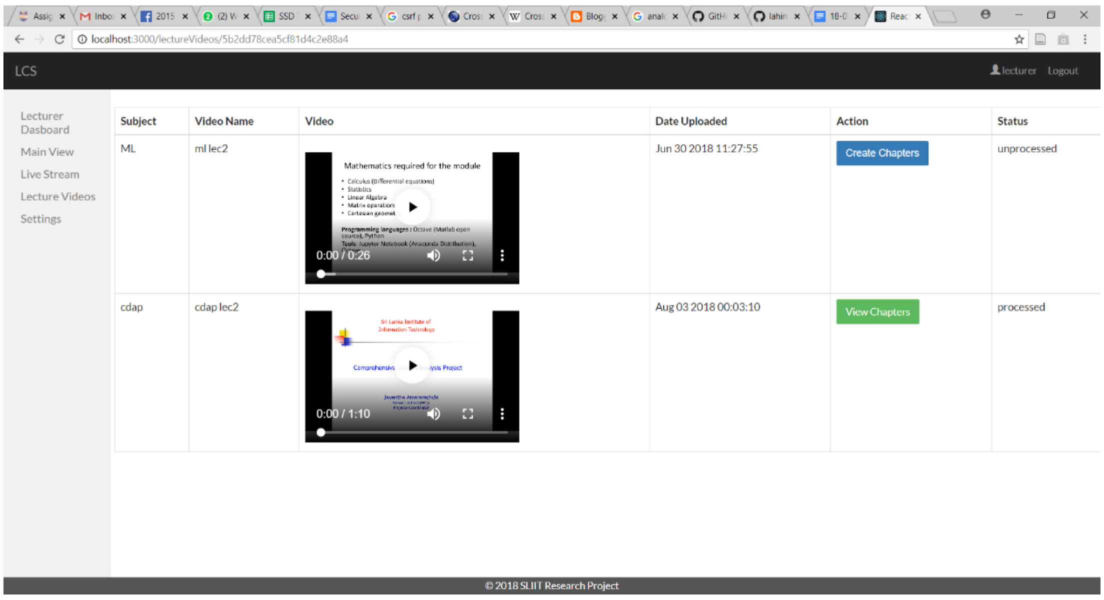


Figure 32: ONVIF Device Manager

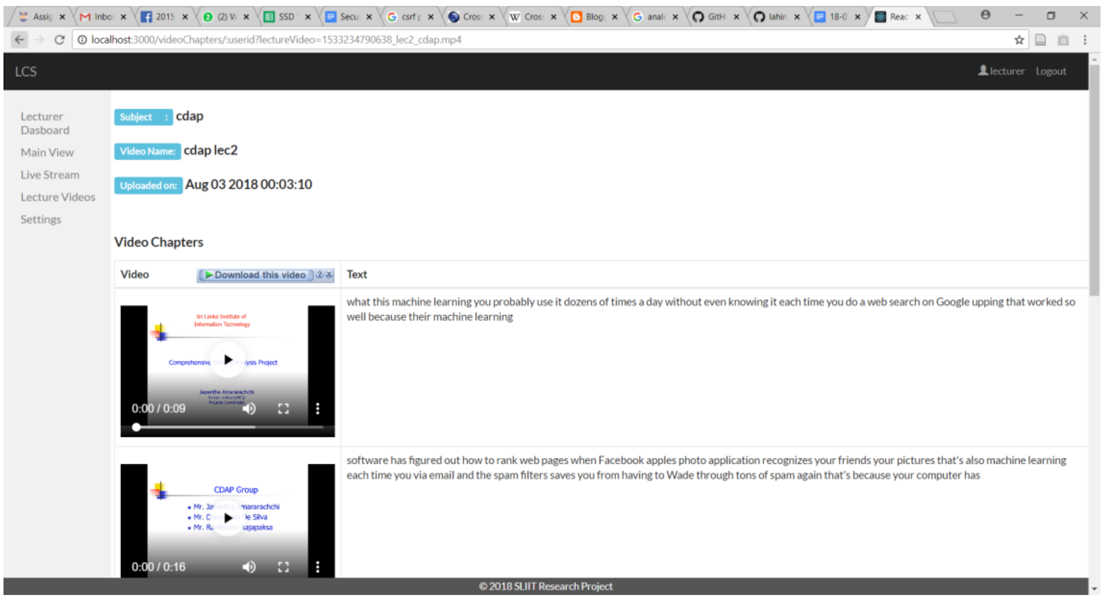
Figure 33 shows the list of videos which were uploaded to the server via the designed OBS Studio plugin. The full-length videos could be watched online or downloaded for later viewing.


localhost:3000/lectureVideos/5b2dd78cea5cf81d4c2e88a4
LCS
lecturer
Logout
Lecturer Dasboard
Main View
Live Stream
Lecture Videos
Settings
© 2018 SLIIT Research Project

Subject	Video Name	Video	Date Uploaded	Action	Status
ML	ml lec2	0:00 / 0:26	Jun 30 2018 11:27:55	Create Chapters	unprocessed
cdap	cdap lec2	0:00 / 1:10	Aug 03 2018 00:03:10	View Chapters	processed

Figure 33: Interface to create or view video chapters

Once the user clicks on the "Create Chapters" button, the video chapter creation process begins. This analyzes and processes the selected video to produce a list of chapters (thumbnails) each representing a PowerPoint presentation slide and the relevant text (converted from the speech of the lecturer for that length of time) which is shown in Figure 34.


localhost:3000/videoChapters/:userid?lectureVideo=1533234790638_lec2_cdap.mp4

LCS | lecturer | Logout

Lecturer Dasboard
Main View
Live Stream
Lecture Videos
Settings

Subject : cdap

Video Name: cdap lec2

Uploaded on: Aug 03 2018 00:03:10

Video Chapters

Video	Text
0:00 / 0:09	what this machine learning you probably use it dozens of times a day without even knowing it each time you do a web search on Google upping that worked so well because their machine learning
0:00 / 0:16	software has figured out how to rank web pages when Facebook apples photo application recognizes your friends your pictures that's also machine learning each time you via email and the spam filters saves you from having to Wade through tons of spam again that's because your computer has

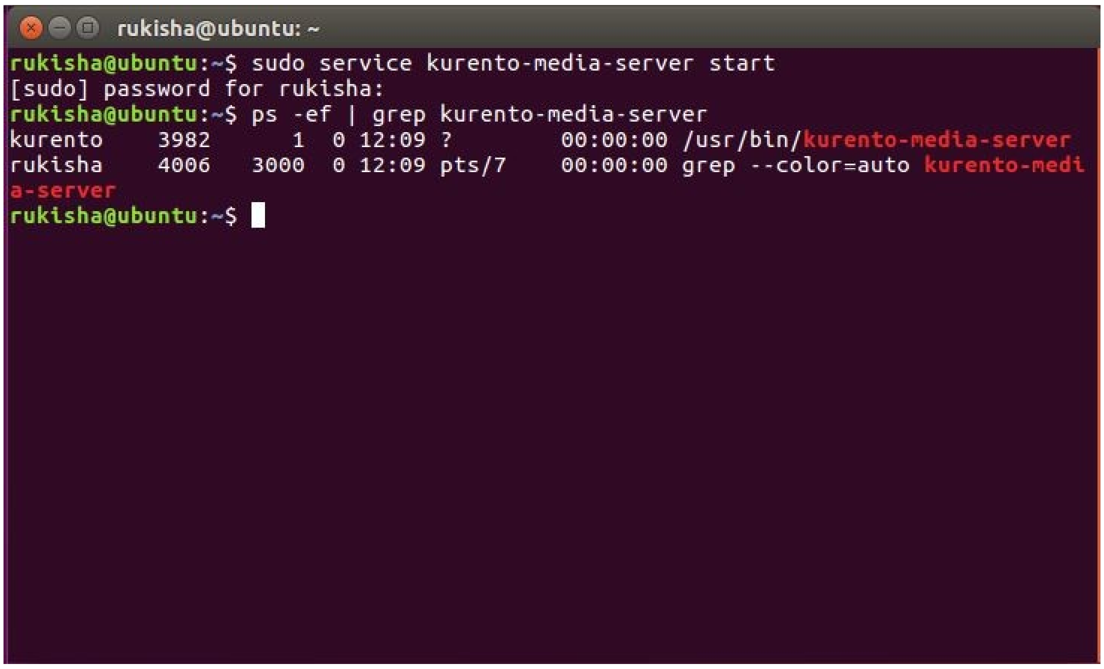
© 2018 SLIIT Research Project

Figure 34: Created video chapters and their text

In terms of face recognition-based login, the LBP classifier reported an accuracy level of 91.33% for a particular user in terms of face detection by maintaining a shorter training time. In contrast with the Haar classifier which reported 81.05% for a user and took a longer training time, LPB classifier has surprisingly outperformed Haar classifier when detecting faces. Since the lecture capturing system face recognition-based login should be fast and should maintain an accuracy level greater than 90%, LBP classifier is the ideal solution which is used currently in the lecture capturing system. However, the results are derived by allocating 12 medium quality training images captured from a webcam for each user. Therefore, the results reported by the classifiers were not satisfactory and the accuracy can change if each user is allocated more high-quality images for training.

In the scenario of gesture-based camera control, the appearance-based approach described under methodology gives an accuracy of 93.25% based on the set of input gestures given. As the number of defects is calculated by applying the cosine rule to all the triangles identified in the hand region, it is highly important that there should be optimum lighting level in the lecture hall. Furthermore, there should be no other skin coloured objects in the area of interest where the gesture is performed (i.e. the green coloured rectangular box region). This is because skin tone thresholding is involved by pixel selection.

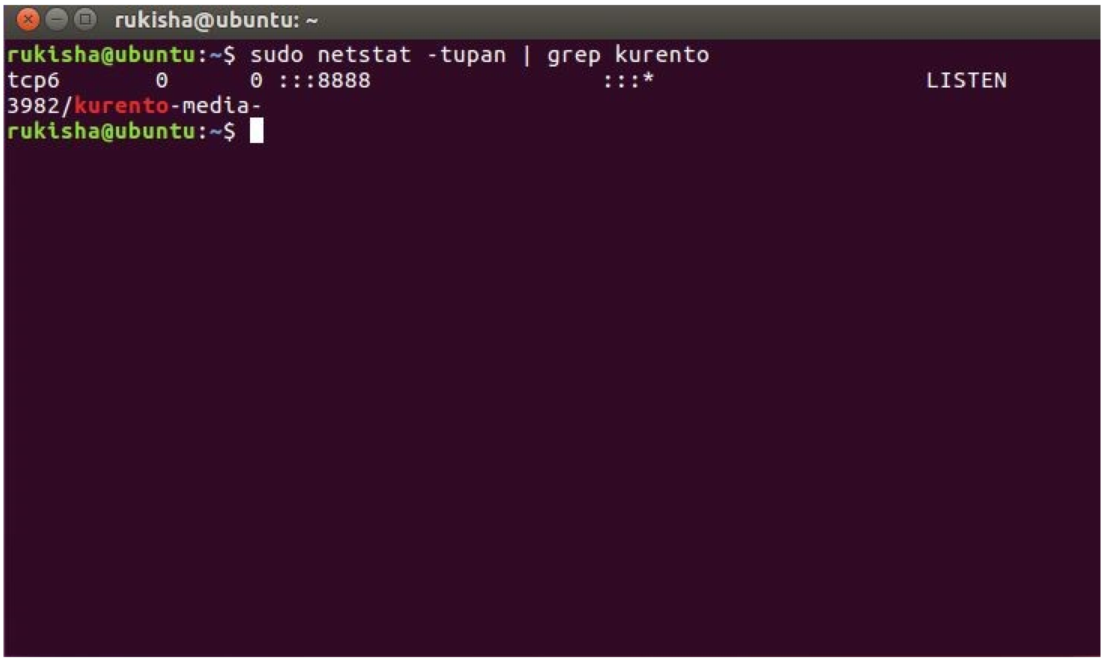
When obtaining the live stream using Kurento Media Server, it is necessary for Kurento Media Server to be up and running on a Linux based operating System. Kurento Media Server exclusively supports Ubuntu 14.04 and Ubuntu 16.04 versions. After installing and running Kurento Media Server as mentioned in Kurento documentation [43], it can be tested and verified whether Kurento Media Server is up and running as shown in the figure below.



```
rukisha@ubuntu:~$ sudo service kurento-media-server start
[sudo] password for rukisha:
rukisha@ubuntu:~$ ps -ef | grep kurento-media-server
kurento    3982      1  0 12:09 ?        00:00:00 /usr/bin/kurento-media-server
rukisha    4006   3000  0 12:09 pts/7    00:00:00 grep --color=auto kurento-medi
a-server
rukisha@ubuntu:~$
```

Figure 35: Output of the processes of Kurento Media Server when it is up and running

To verify further the port that Kurento Media Server Listens on, the following can be done.



```
rukisha@ubuntu:~$ sudo netstat -tupan | grep kurento
tcp6       0      0 :::8888                 :::*                    LISTEN
3982/kurento-media-
rukisha@ubuntu:~$
```

Figure 36: Result indicates that Kurento has opened the port 8888 to send and receive requests using Kurento Protocol

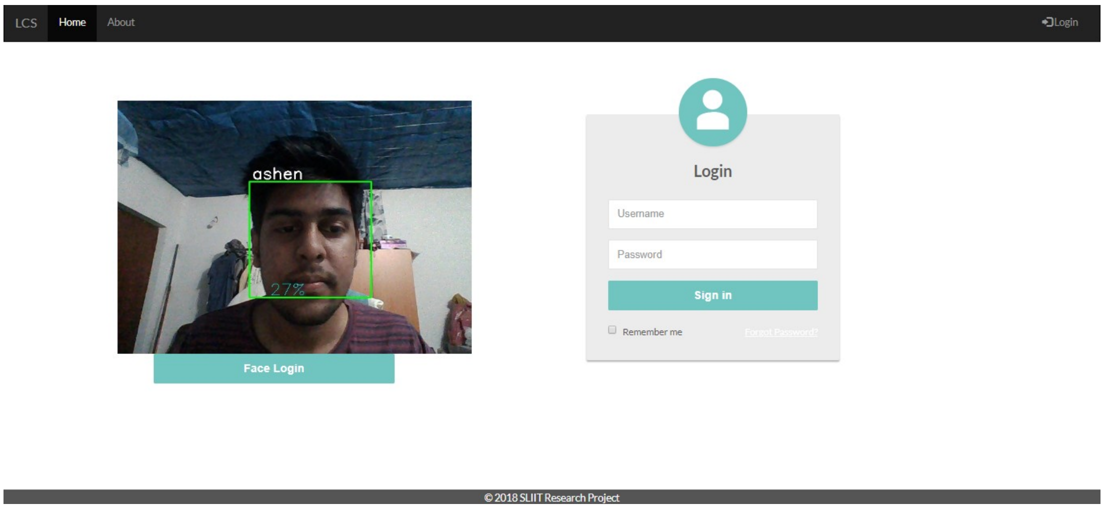
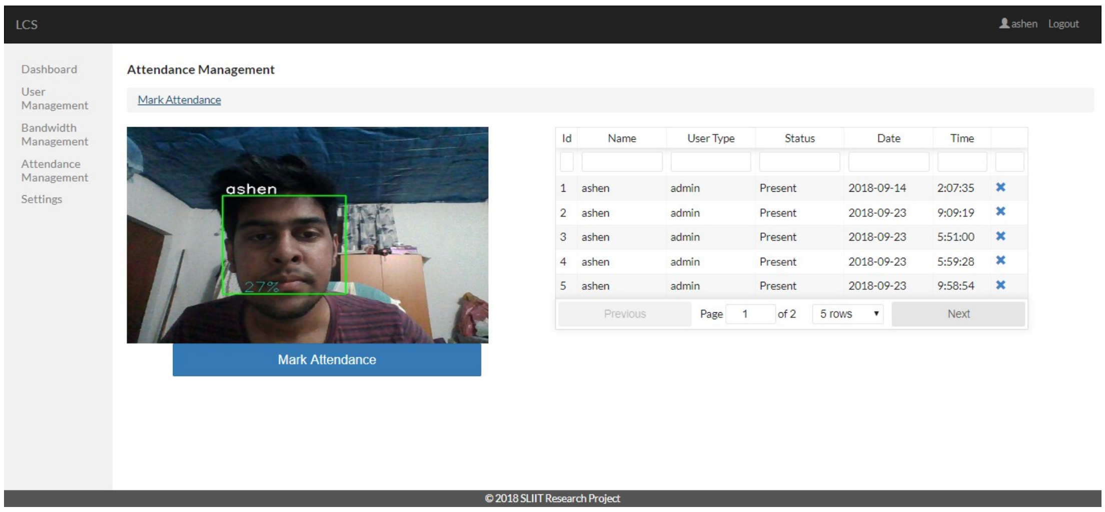
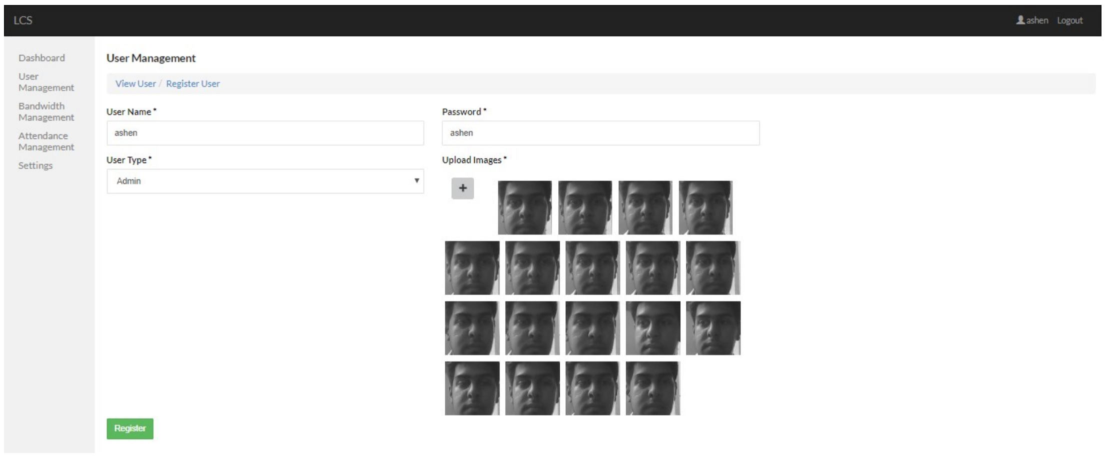


Figure 37: Face Login Interface



Id	Name	User Type	Status	Date	Time	
1	ashen	admin	Present	2018-09-14	2:07:35	✖
2	ashen	admin	Present	2018-09-23	9:09:19	✖
3	ashen	admin	Present	2018-09-23	5:51:00	✖
4	ashen	admin	Present	2018-09-23	5:59:28	✖
5	ashen	admin	Present	2018-09-23	9:58:54	✖

Figure 38: Attendance Marking Interface



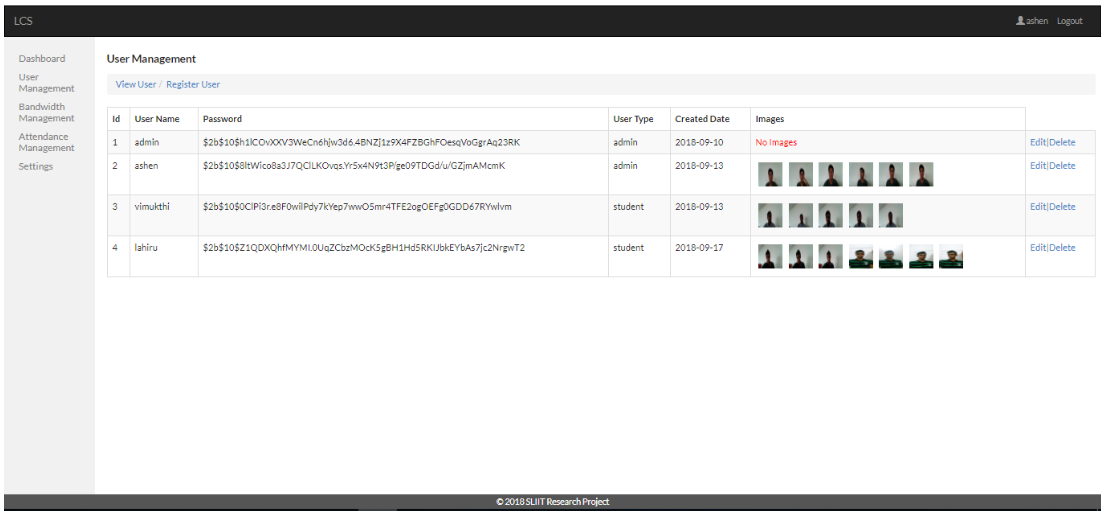
User Name * ashen

Password * ashen

User Type * Admin

Upload Images *

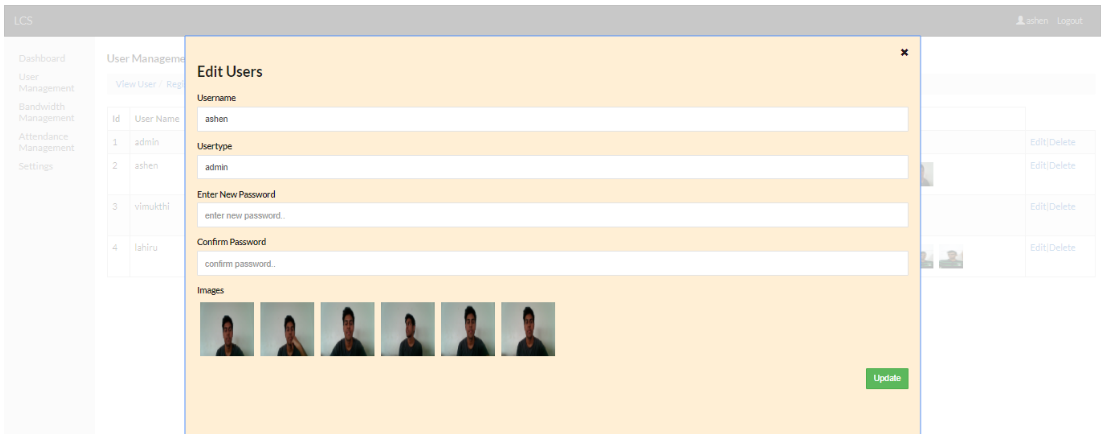
Figure 39: User Registration Interface



User Management

Id	User Name	Password	User Type	Created Date	Images
1	admin	$2b$10$h1lCOvXXV3WeCn6hjw3d6.4BNZj1z9X4FZBGhFOesqVoGgrAq23RK	admin	2018-09-10	No Images
2	ashen	$2b$10$8ltWico8a3J7QClLKOvqs.Yr5x4N9t3P/ge09TDGd/u/GZjmAMcmK	admin	2018-09-13	
3	vimukthi	$2b$10$0ClPi3r.e8F0wilPdy7kYep7wwO5mr4TFE2ogOEFg0GDD67RYwlvm	student	2018-09-13	
4	lahiru	$2b$10$Z1QDXQhfMYMI.0UqZCbzMOcK5gBH1Hd5RKIJbkEYbAs7jc2NrgwT2	student	2018-09-17	

Figure 40: : User List Interface



Edit Users

Username	ashen
Usertype	admin
Enter New Password	enter new password..
Confirm Password	confirm password..
Images	

Figure 41: Edit User Interface

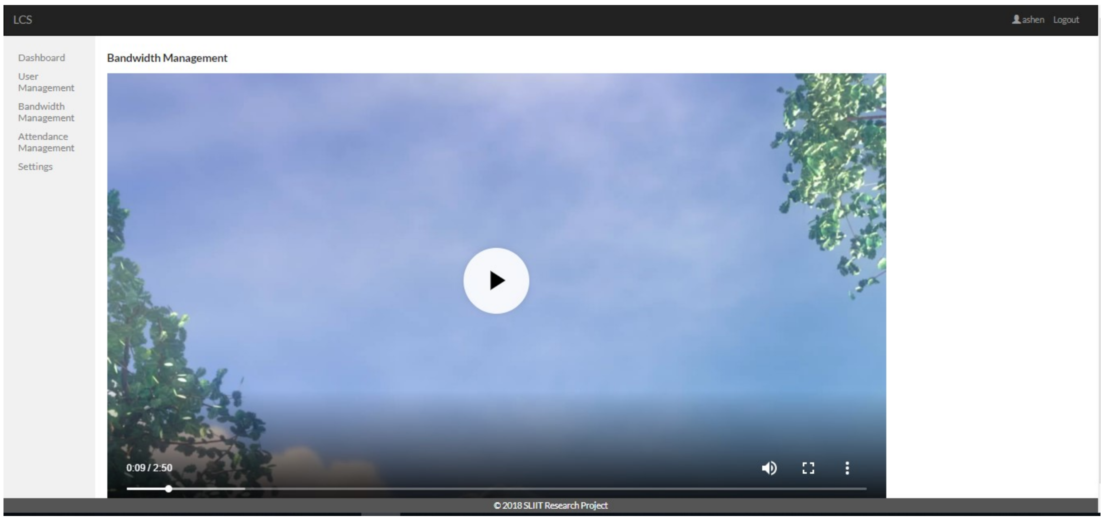
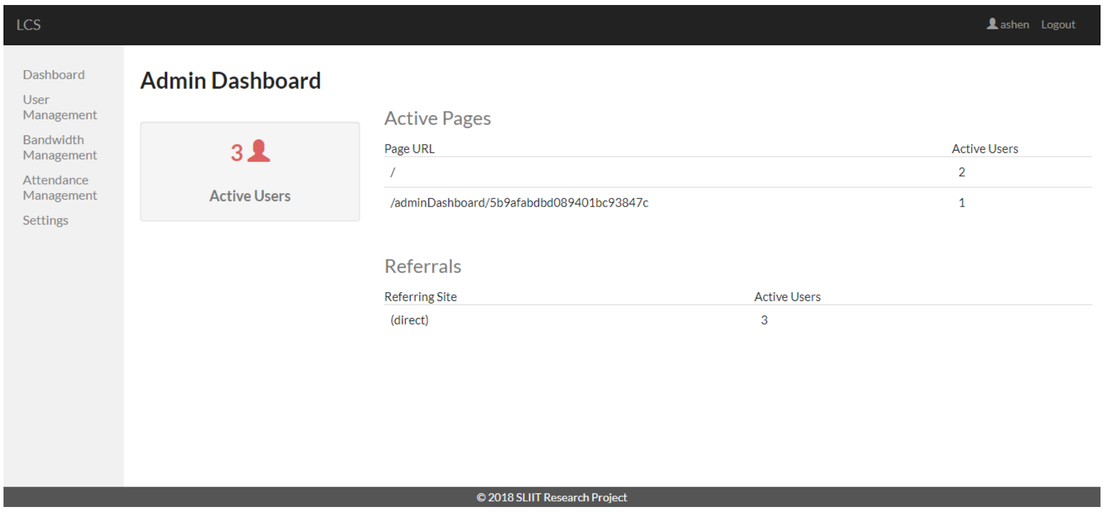


Figure 42: Bandwidth Managment Interface



Admin Dashboard

3 Active Users

Active Pages

Page URL	Active Users
/	2
/adminDashboard/5b9afabdbd089401bc93847c	1

Referrals

Referring Site	Active Users
(direct)	3

Figure 43: Dashboard Statistics Interface

Accuracy has been calculated by testing 30 faces for each user and accurate recognition rates are mentioned in the table below where rates are derived using the formula:

Accuracy % = 100 - Confidence Index

The confidence index will return zero if it will be considered a perfect match in detection or recognition. If not an ‘unknown’ label is put on the face.

Average	LBP classifier	Haar classifier
Accuracy of recognition	60%	90%
Processing time for encoding and training a user with 30 face images (in seconds)	1.8min	2.1min

Testing with different face datasets from 50 – 100 range when training images. A laptop with i7, 8th generation and 8GB RAM is used to obtain the below result.

Images per user	LBP Classifier	Haar Classifier
50 Images	2.5min	4min
80 Images	3.1min	5min
100 Images	3.9min	6.9min

Accuracy rate when input devices were switched between Webcam and IP-Camera.

Device	Accuracy
Webcam	Totally depends on the predefined resolution and fps on the camera.

IP Camera	Totally depends on the predefined resolution and fps on the camera.

Accuracy rate with training dataset containing with and without blur images.

Type of Image	Accuracy
With Blur Images	15%
Without Blur Images	80%

Amount of main memory with is used to execute the algorithm is defined as memory used and given in MB.

Algorithm	LBP Algorithm	Haar-Like Algorithm
Memory Used	123MB	290MB

In terms of face recognition-based login, the LBP classifier reported an accuracy level of 70.33% for a particular user in terms of face detection by maintaining a shorter training time. In contrast with the Haar classifier which reported 81.05% for a user and took a longer training time, LPB classifier has underperformed Haar classifier when detecting faces. Since the lecture capturing system face recognition-based login should be fast and should maintain a higher accuracy level greater than 80%, Haar classifier is the ideal solution which is used currently in the lecture capturing system. However, the results are derived by allocating 30 medium quality training images captured from a webcam for each user. Therefore, the results reported by the classifiers were not satisfactory and the accuracy can change if each user is allocated more high-quality images for training.

Comparison between Haar Classifier and LBP Classifier

1. LBP Classifier is faster than Haar classifier.
2. Haar classifier uses floats to do all the calculations while LBP classifier use integers.
3. LBP classifier is less accurate than Haar classifier.
4. Haar-like features in the Haar Cascade classifier work best for frontal face detection.
5. Haar features are good at detecting edges and lines which is effective in face detection.

3.2 Discussion

A complete testing cycle was conducted on the system to make sure all the functionalities identified in the requirement gathering phase were fulfilled. Below are the different types of tests that were conducted on the system.

- **Unit Testing -** Each unit of the system was tested to ensure that each individual unit is bug-free and functional.
- **Component Testing** – Several bug free units were combined together and tested as a component.
- **Integration Testing** – In this testing phase the relationships and communication between assembled components was tested for expected functionality.
- **System Testing** – All the different components were combined together and the whole system was tested to verify its functionality.

- **Regression Testing** - Regression testing is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features [44]. This ensured that any new changes made did not lead to any bugs in the completed system.

With regards to facial recognition, the testing planning includes planning for several functions like:

- Face Login
- Unknown face login effort
- Credential Based Login
- Attendance Marking
- Filter Attendance of a student
- User Registration

Some test cases are shown below.

Table 7: Test Case – Face Login

Id	TC01
Title	Face Login
Prerequisite	Server started User registered (Registration includes user's face data prepared and trained by the recognizer) in the system
Test Action	1. Start LCS application 2. Click the face login button
Expected Result	User will get automatically logged in to the application.

Table 8: Test Case - Unknown Face Login Effort

Id	TC02
Title	Unknown Face Login Effort
Prerequisite	Server started User not registered in the system
Test Action	1. Start LCS application 2. Click the face login button
Expected Result	User will not be logged in to system and the user will get an error warning below the face login button.

Table 9: Test Case - Credential Based Login

Id	TC03
Title	Credential Based Login
Prerequisite	Server started User registered in the system
Test Action	1. Start LCS application 2. Enter the username and password 3. Click the login button

Expected Result	User will get logged in to the application.

Table 10: Test Case - Attendance Marking

Id	TC04
Title	Attendance Marking
Prerequisite	Server started User registered in the system. Students available in the class room.
Test Action	1. Start LCS application 2. Login to the application by face login or normal credential based login. 3. Navigate to Attendance Management Page by clicking the ‘Attendance Management’ tab in the navigation panel. 4. Click the ‘Mark Attendance’ button to mark the attendance of all students
Expected Result	The students who are registered and recognized by the application will be marked as ‘Present’ and the result will be immediately saved in the database and will be available in the attendance filtering table.

Table 11: Test Case - Filter attendance of a student

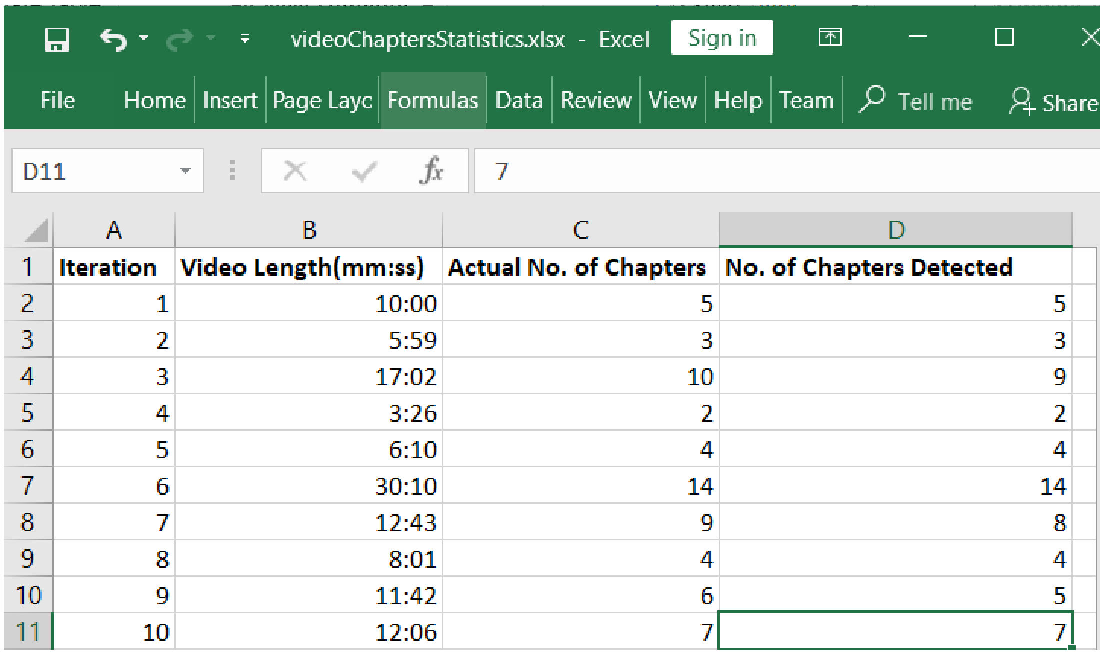
Id	TC05
Title	Filter Attendance of a student
Prerequisite	Server started User registered in the system.
Test Action	1. Start LCS application 2. Login to the application by face login or normal credential based login. 3. Navigate to Attendance Management Page by clicking the ‘Attendance Management’ tab on the navigation panel. 4. Enter the name, date, or time of a student in the filtering text boxes available in the table.
Expected Result	The related record will be filtered out and displayed to the user.

Table 12: Test Case – User Registration

Id	TC06
Title	User Registration
Prerequisite	Server started User registered in the system.

Test Action	1. Start LCS application 2. Login to the application by face login or normal credential based login using admin credentials. 3. Navigate to User Management Page by clicking the ‘User Management’ tab in the navigation panel. 4. Click the ‘Register User’ link available in the tab. 5. Enter the username, password, select the user type from the dropdown and select the related face photos of the user and upload to the page by clicking the ‘plus icon’ in the registration form. 6. Click the ‘Submit’ button
Expected Result	A success message will be displayed to the user if the registration is successful, if not an error message will be displayed.

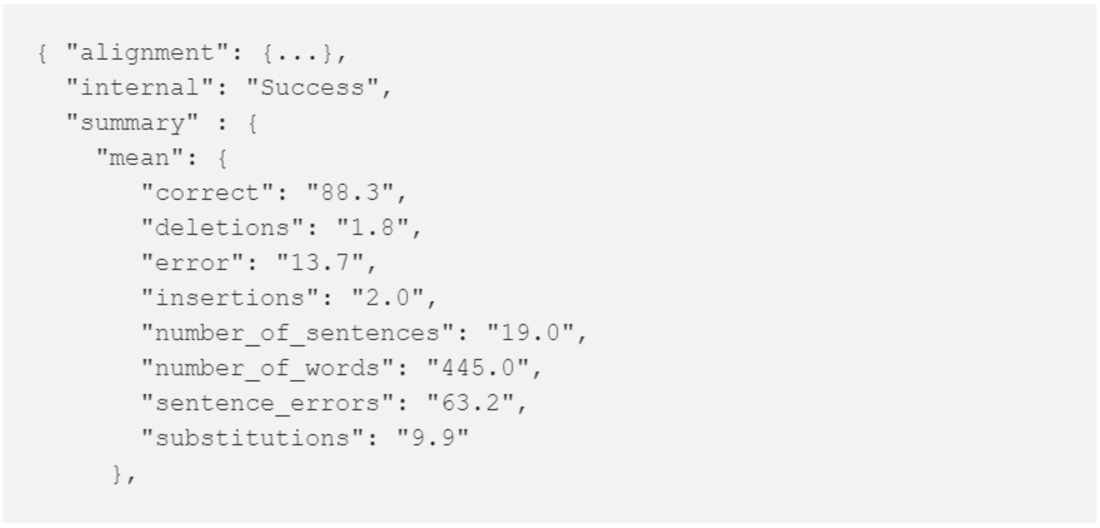
With regards to the video chapter creation feature, each frame in the video is analyzed by an algorithm for changes in color and intensity - namely the average HSV color space difference (difference in hue, saturation, and luminance of the frame). If this calculated value is very high than the preceding and following values, it means that there has been a scene change. Therefore, the video is split at this time frame. This process is repeated for the entire length of the video clip until the entire video clip is analyzed and all the video chapters are created. This process was run over repeated 100-iteration cycles which produced an average accuracy of 95% which is a very satisfactory value. Some of the obtained results are shown in Figure 44.



Iteration	Video Length(mm:ss)	Actual No. of Chapters	No. of Chapters Detected
1	10:00	5	5
2	5:59	3	3
3	17:02	10	9
4	3:26	2	2
5	6:10	4	4
6	30:10	14	14
7	12:43	9	8
8	8:01	4	4
9	11:42	6	5
10	12:06	7	7

Figure 44: Video Chapters Creation Statistics

Following this process is the real-time speech transcription (audio-to-text conversion) of each video chapter. This is achieved via the Watson Speech-To-Text API. This service leverages machine intelligence to transcribe the human voice accurately. The service combines information about grammar and language structure with knowledge of the composition of the audio signal resulting in a remarkable accuracy level of 88.3% as shown in Figure 45.



```
{ "alignment": {...},
  "internal": "Success",
  "summary" : {
    "mean": {
       "correct": "88.3",
       "deletions": "1.8",
       "error": "13.7",
       "insertions": "2.0",
       "number_of_sentences": "19.0",
       "number_of_words": "445.0",
       "sentence_errors": "63.2",
       "substitutions": "9.9"
    },
```

Figure 45: Speech-To-Text Statistics

4. CONCLUSION

This report examines an innovative approach that is best suited to develop a lecture capturing system that provides a complete classroom experience and much more to remotely logged in students. This system stands unique from other existing products as being a comprehensive product that includes biometric authentication, gesture detection, live streaming of lectures, automated attendance marking, offline recording of lectures, bandwidth management and desktop screen capturing all in one.

This research work has been developed mainly for addressing the problems in Sri Lankan universities, specifically addressing the lack of interactivity between the lecturer and the students. Even though this research focuses on universities, it definitely has the potential to be used in other fields such as business conferencing. In next stage, in one hand, research team will be focusing on improving the accuracy of the face recognition and gesture detection models by testing other algorithms effectively. Also, research team will focus on minimizing bandwidth costs by testing out bandwidth optimization techniques. It is hoped that for any person who expects to build a similar system or any other real-time system, results of this research will be an aid and will provide insight on the performance, accuracy and reliability level that can be expected with the combination of tools, technologies, programming approach considered in this paper.

5. REFERENCES

[1] W., *"Use of E-Learning," Universiti Teknologi*. Malaysia, Johor, Malaysia, 2018.

[2] 3] M. A. Mahmod, "E-learning in Iraqi Universities," in *International Conference on Computing, Engineering, and Design (ICCED), Selangor*, Malaysia, 2017.

[3] 4] P. D. Z. Varcheie, "Online Body Tracking by a PTZ Camera in IP Surveillance System," Department of Computer Engineering and Software Engineering, Station Centre-ville, Montr´eal, (Qu´ebec), Canada, 2009.

[4] T. G. Dries Hulens, "Autonomous lecture recording with a PTZ camera," presented at the Canadian Conference on Computer and Robot Vision, Belgium, 2014.

[5] 5] M. M. M. H. R. Jacko, "Remote control of the PTZ camera system for lecture rooms," *Department of Computers and Informatics*, 2015.

[6] 7] B. Wulff, "OpenTrack - Automated Camera Control for Lecture Recordings," *IEEE International Symposium on Multimedia*, 2011.

[7] C.-F. C. a. P.-C. S. Yong-Quan CHEN, "A Tabletop Lecture Recording System," in *International Conference on Consumer Electronics-Taiwan*, Taiwan, 2015.

[8] Y.-T. T. S. C. a. S.-W. C, "Chiung-Yao Fang, 'Chiung-Yao Fang, You-Ting Tsai, Shuan Chu, and Sei-Wang Chen,'" Department of Computer Science and Information Engineering, Taiwan, 2015.

[9] K. Kumar and T. S. Sheng, "Real Time Target Tracking with Pan Tilt Zoom Camera," presented at the Digital Image Computing, Adelaide, 2009.

[10] D. P. Hutabarat, D. Patria, S. Budijono, and R. Saleh, "Human Tracking Application in a Certain Closed Area Using RFID Sensors and IP," presented at the Information Tech, Jakarta, 2016.

[11] C. C. Phan, B. T. Nguyen, and S. T. Chung, “Design and Implementation of ONVIF-based Event Service for DM 814x Camera,” presented at the IEEE, DaNang, 2015.

[12] C. S. Yang, R. H. Chen, C. Y. Lee, and S. J. Lin, “PTZ Camera Based Position Tracking in IP - Surveillance System,” presented at the IEEE, Tainan, 2008.

[13] “Automated Student Attendance Management System Using Face Recognition | Ise A Orobor and Ofualagba Godswill - Academia.edu.” [Online]. Available: http://www.academia.edu/37437099/Automated_Student_Attendance_Management_System_Using_Face_Recognition. [Accessed: 10-Oct-2018].

[14] V. Mankar and S. G Bhele, “A Review Paper on Face Recognition Techniques,” *International Journal of Advanced Research in Computer Engineering & Technology*, vol. 1, pp. 339–346, Oct. 2012.

[15] F. Ahmad, “Image-based Face Detection and Recognition.” [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1302/1302.6379.pdf.

[16] “WebRTC 1.0: Real-time Communication Between Browsers.” [Online]. Available: https://www.w3.org/TR/webrtc/. [Accessed: 10-Oct-2018].

[17] P. Braun, M. Sipos, P. Ekler, and F. Fitzek, “On the Performance Boost for Peer To Peer WebRTC-based Video Streaming with Network Coding,” 2017.

[18] M. Y. Abbass and K. HyungWon, “Blind image separation using pyramid technique,” Dec. 2017.

[19] C. Jiandong, “Urban Intelligent Traffic Monitoring System Based on Video Image Processing.”

[20] “Lecture Capture & Recording Software For Education | Panopto.” [Online]. Available: https://www.panopto.com/panopto-for-education/lecture-capture/. [Accessed: 10-Oct-2018].

[21] “Home - BigBlueButton.” [Online]. Available: https://bigbluebutton.org.

[22] "Echo360 - The Smarter Video Platform for Higher Ed and Continuing Ed." [Online]. Available: https://echo360.com/. [Accessed: 10-Oct-2018].

[23] "Kaltura Video Platform - Powering Any Video Experience." [Online]. Available: https://corp.kaltura.com/. [Accessed: 10-Oct-2018].

[24] "Open Broadcaster Software | Home." [Online]. Available: https://obsproject.com/. [Accessed: 26-Mar-2018].

[25] "About Kurento and WebRTC." [Online]. Available: https://doc-kurento.readthedocs.io/en/stable/user/about.html.

[26] "Interoperating WebRTC and IP Cameras." [Online]. Available: http://www.kurento.org/blog/interoperating-webrtc-and-ip-cameras.

[27] "OpenCV library." [Online]. Available: https://opencv.org.

[28] "OpenCV: Face Detection using Haar Cascades." [Online]. Available: https://docs.opencv.org/3.4.2/d7/d8b/tutorial_py_face_detection.html.

[29] "YOLO: Real-Time Object Detection." [Online]. Available: https://pjreddie.com/darknet/yolo.

[30] "Onvif." [Online]. Available: https://www.onvif.org/onvif/ver20/util/operationIndex.html.

[31] "WebRTC Home | WebRTC." [Online]. Available: https://webrtc.org.

[32] P. Garg, N. Aggarwal, and S. Sofat, "Vision Based Hand Gesture Recognition," *World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 3, no. 1, 2009.

[33] "Face Detection using OpenCV and Python: A Beginner's Guide." .

[34] "An Overview of H.264 Advanced Video Coding," *Vcodex*. [Online]. Available: http://www.vcodex.com/an-overview-of-h264-advanced-video-coding/. [Accessed: 07-Oct-2018].

[35] "What is Advanced Audio Coding (AAC)? - Definition from Techopedia," *Techopedia.com*. [Online]. Available: https://www.techopedia.com/definition/218/advanced-audio-coding-aac. [Accessed: 07-Oct-2018].

[36] "How to convert FLVs to MP4 fast without re-encoding," *Open Broadcaster Software*. [Online]. Available: https://obsproject.com/forum/resources/how-to-convert-flvs-to-mp4-fast-without-re-encoding.78/. [Accessed: 26-Mar-2018].

[37] "About - OpenCV library." [Online]. Available: https://opencv.org/about.html. [Accessed: 07-Oct-2018].

[38] "ffmpeg Documentation." [Online]. Available: https://www.ffmpeg.org/ffmpeg.html. [Accessed: 07-Oct-2018].

[39] B. Castellano, :*movie_camera: A Python/OpenCV-based scene detection program, using threshold/content analysis on a given video.: Breakthrough/PySceneDetect*. 2018.

[40] "Command Reference — PySceneDetect v0.5 documentation." [Online]. Available: https://pyscenedetect-manual.readthedocs.io/en/latest/cli/commands.html. [Accessed: 09-Oct-2018].

[41] "Introduction - PySceneDetect." [Online]. Available: https://pyscenedetect.readthedocs.io/en/latest/. [Accessed: 08-Aug-2018].

[42] "Watson Speech to Text," 28-Nov-2016. [Online]. Available: https://www.ibm.com/watson/services/speech-to-text/. [Accessed: 08-Aug-2018].

[43] Kurento Technologies, "Kurento Documentation," 2014.

[44] "What is Regression Testing? Test Cases, Tools & Examples." [Online]. Available: https://www.guru99.com/regression-testing.html. [Accessed: 10-Oct-2018].