

**CLOUD-BASED LECTURE CAPTURING SYSTEM:
A CASE STUDY**

L. C. Tennakoon

IT15010322

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**

L. C. Tennakoon

IT15010322

**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 log in 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 advice. 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

DECLARATION	i
ABSTRACT	ii
ACKNOWLEDGEMENT	iii
LIST OF TABLES	v
LIST OF FIGURES	v
LIST OF ABBREVIATIONS	vi
1. INTRODUCTION	1
1.1 Problem to be addressed	1
1.2 Background context	2
1.3 Research gap	3
1.4 Research questions	5
2. BODY OF THE REPORT	7
2.1 Addressing the Literature	7
2.1.1 Existing Applications	13
2.2 Use Cases	16
2.3 Methodology	18
2.3.1 Open Broadcaster Software (OBS) Studio plugin	19
2.3.2 Video Thumbnails/Chapters Creation	20
2.3.3 Quota Management	27
2.4 Research Findings	27
3. RESULTS AND DISCUSSION	29
3.1 Evidence	29
3.2 Discussion	30
4. CONCLUSION	33
5. REFERENCES	34

6. APPENDICES	38
---------------------	----

LIST OF TABLES

Table 1: List of Abbreviations	vi
Table 2: Comparison with existing systems.....	3
Table 3: Use case scenario for offline screen capture.....	16
Table 4: Use case scenario for video thumbnails creation	16
Table 5: Use case scenario for quota management	17
Table 6: Input Parameters for PySceneDetect	22

LIST OF FIGURES

Figure 1: System Architecture	18
Figure 2: OBS Studio Plugin Main Interface	19
Figure 3: Video Uploading Process	20
Figure 4: Generated Statistics File	23
Figure 5: Detecting a Scene Change from Statistics	24
Figure 6: List of Video Chapters (Scenes/Thumbnails).....	25
Figure 7: Core part of the Algorithm	26
Figure 8: Interface to create or view video chapters	29
Figure 9: Created video chapters and their text	30
Figure 10: Video Chapters Creation Statistics	31
Figure 11: Speech-To-Text Statistics.....	32
Figure 12: Main Method	38
Figure 13: Initializing the Scene Detector	39
Figure 14: Core Part of the Video Splitting Algorithm.....	40

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 of 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 of 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 the 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 of 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 types of research have been done to find the best way

to use 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 the 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 $1/4$ th 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 the 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 the SQLite database. This method of data storage doesn't require the installation of a 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 the Spn 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 a 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 in 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, the human tracking system is built on a specific area that has lots of room or building such as a theme park or sports club. The 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 the 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 supported 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 the object motion vector, the relevant information such as time, background and geographic parameters can be recorded in the database as well [12]. In the experiment, the change of a person's position is recorded by the PTZ camera in real time. The PCTS provides a feasible solution for position analysis and security surveillance services in the 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 works 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 captures an image of the entire class at a 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 grayscale 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 are the proposed face detection solution in [15] which has reported relatively well than LBP method. In terms of recognition, the 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 introduces 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

The main 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	<ol style="list-style-type: none">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	<p>2a. The lecturer enters an invalid username/password to connect to the remote server.</p> <p>2a1. The plugin displays an error saying "Invalid username/password. Try again."</p> <p>3a. The internet connection is interrupted during the video transfer.</p> <p>3a1. The plugin displays an error "Internet connection disrupted. Please upload the video again".</p> <p>3a2. The lecturer uploads the video again.</p>

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	<ol style="list-style-type: none"> 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 are 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	<p>2a. The video is corrupted so the system is unable to process the file.</p> <p>2a1. The system displays an error saying “Corrupted video file. Please try again.”</p>

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	<ol style="list-style-type: none"> 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	<p>4a. The administrator enters an invalid value (e.g. letters) in the monthly quota field.</p> <p>4a1. The system displays an error saying “Invalid value. Please enter a number.”</p>

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.

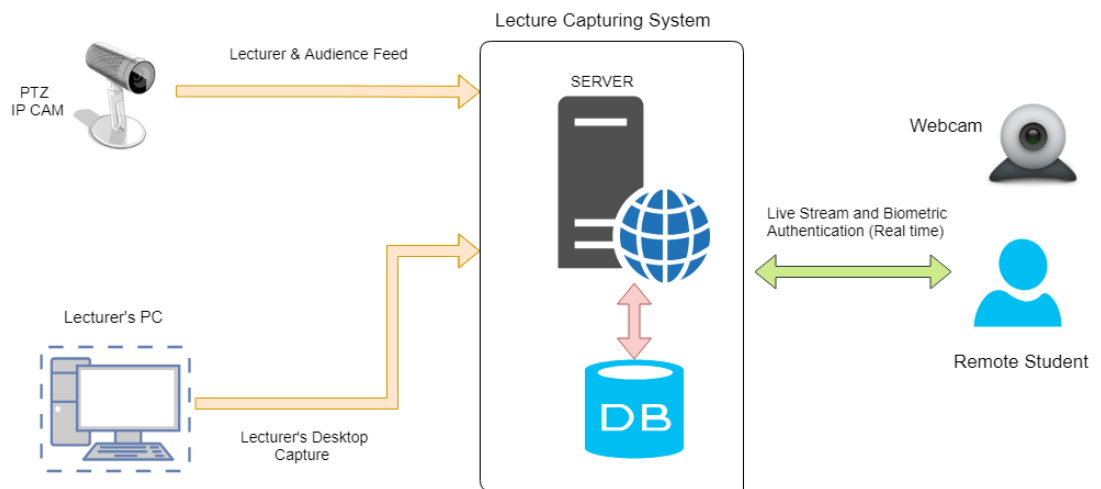


Figure 1: System Architecture

2.3.1 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 2 for the OBS Studio software which allows the lecturer to upload the recorded video directly to the server for further processing.

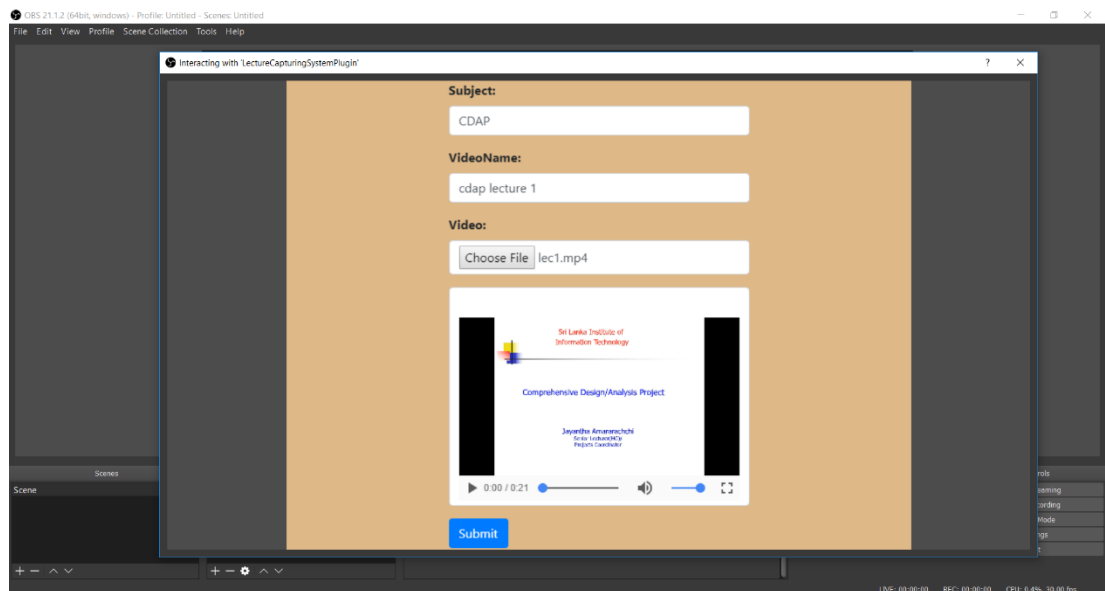


Figure 2: OBS Studio Plugin Main Interface

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 [25]. 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 [26]. 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 [27]. 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 3 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.

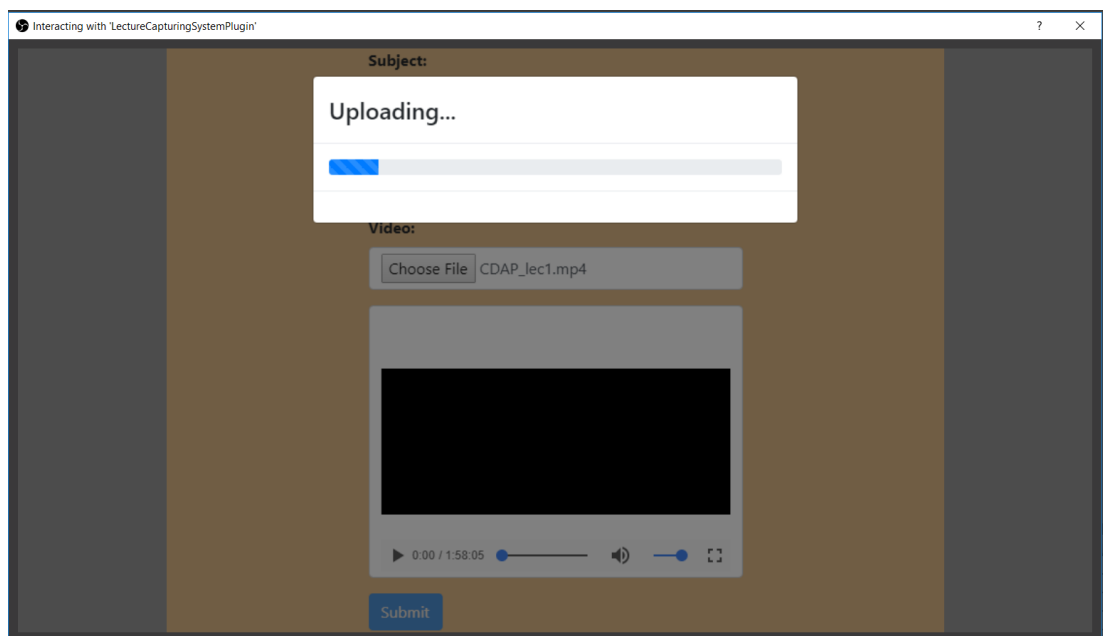


Figure 3: Video Uploading Process

2.3.2 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 [28]. 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 [29].

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 [30]. 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 [31].

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 the video to begin detecting scenes.
--duration	Maximum time in video to process.
--end	Time in the 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 4.

	A	B	C	D	E	F	G
1	Frame Number	Timecode	delta_hue	delta_lum	delta_hsv_avg	delta_sat	
2	1	00:00.0	0.0033322	0.0001215	0.001382017	0.000692274	
3	2	00:00.1	0.0018001	0.0001172	0.000804036	0.000494792	
4	3	00:00.1	0.0021756	9.44E-05	0.000913628	0.00047092	
5	4	00:00.1	0.0042122	0.0001714	0.001632306	0.000513238	
6	5	00:00.2	0.0275271	0.0003429	0.00958912	0.000897352	
7	6	00:00.2	0.0009798	0.000191	0.000530237	0.000419922	
8	7	00:00.2	0.0015592	0.0001845	0.000715784	0.000403646	
9	8	00:00.3	0.0136372	0.00037	0.005014829	0.001037326	
10	9	00:00.3	0.1371593	0.0032878	0.048893953	0.006234809	
11	10	00:00.3	0.0166189	0.0006337	0.006148003	0.001191406	
12	11	00:00.4	0.0161664	0.0006847	0.006041667	0.001273872	
13	12	00:00.4	0.2159201	0.0030273	0.076304615	0.009966363	
14	13	00:00.4	0.3310124	0.007168	0.118004196	0.015832248	
15	14	00:00.5	0.0455914	0.0032064	0.017816479	0.004651693	
16	15	00:00.5	0.0542741	0.0037337	0.02109158	0.005266927	
17	16	00:00.5	0.2654253	0.0065321	0.09572555	0.015219184	
18	17	00:00.6	0.5485189	0.0145497	0.198399884	0.032131076	
19	18	00:00.6	0.1790169	0.0063867	0.065477431	0.011028646	
20	19	00:00.6	0.1960069	0.0072504	0.071977358	0.012674696	
21	20	00:00.7	0.5560872	0.0121766	0.197677951	0.024769965	
22	21	00:00.7	0.7946267	0.0206641	0.288415075	0.049954427	
23	22	00:00.7	0.2929709	0.0068837	0.105362052	0.016231554	
24	23	00:00.8	0.3048958	0.0079557	0.110304543	0.018062066	
25	24	00:00.8	1.1098416	0.0157368	0.387459491	0.03680013	

Figure 4: 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.

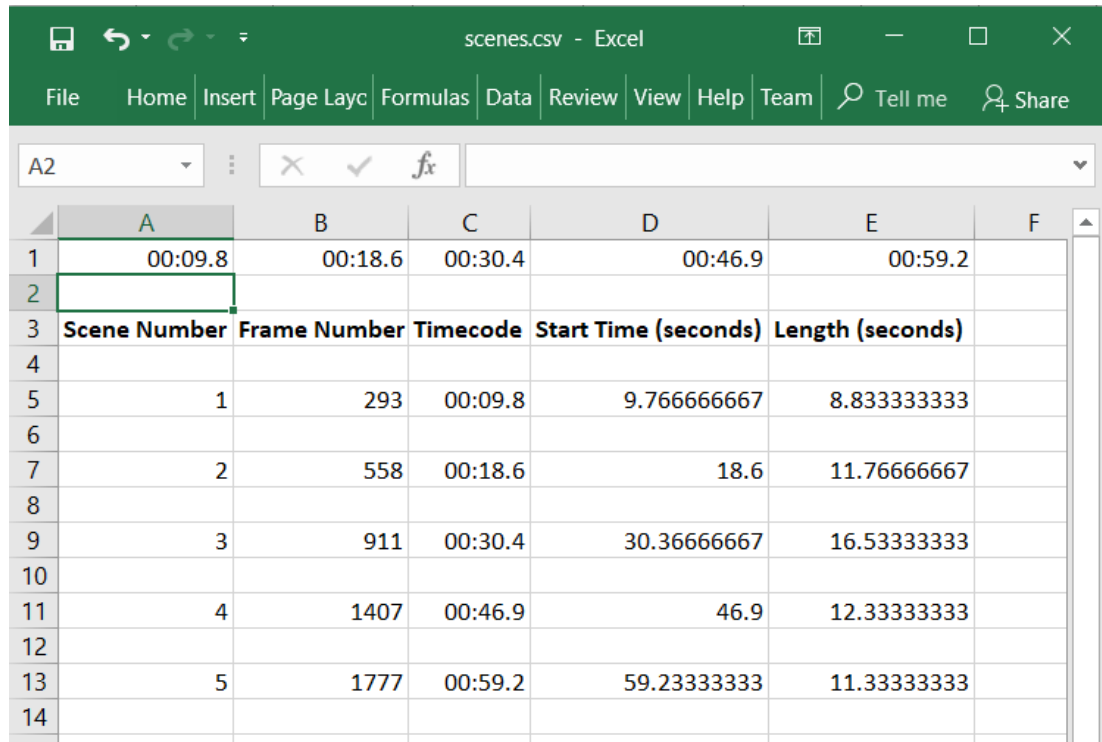
Due to the better performance of the content-aware detection method, it was used for the video splitting process. It compares the difference in content between adjacent frames against a set threshold/score, which if exceeded, triggers a scene cut. It checks

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 [32]. 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 5.

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 5: Detecting a Scene Change from Statistics

Therefore, the video is split at this time frame. For Figure 5, 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 6. 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	11.766666667	
8						
9	3	911	00:30.4	30.366666667	16.533333333	
10						
11	4	1407	00:46.9	46.9	12.333333333	
12						
13	5	1777	00:59.2	59.233333333	11.333333333	
14						

Figure 6: List of Video Chapters (Scenes/Thumbnails)

The core part of the algorithm responsible for detecting scenes and creating the video chapter is shown in Figure 7. More details of the source code are available in the APPENDICES section.

```

if self.last_frame is not None:
    # Change in average of HSV (hsv), (h)ue only, (s)aturation only, (l)uminance only.
    delta_hsv_avg, delta_h, delta_s, delta_v = 0.0, 0.0, 0.0, 0.0

    if (self.stats_manager is not None and
        self.stats_manager.metrics_exist(frame_num, metric_keys)):
        delta_hsv_avg, delta_h, delta_s, delta_v = self.stats_manager.get_metrics(
            frame_num, metric_keys)

    else:
        num_pixels = frame_img.shape[0] * frame_img.shape[1]
        curr_hsv = cv2.split(cv2.cvtColor(frame_img, cv2.COLOR_BGR2HSV))
        last_hsv = self.last_hsv
        if not last_hsv:
            last_hsv = cv2.split(cv2.cvtColor(self.last_frame, cv2.COLOR_BGR2HSV))

        delta_hsv = [0, 0, 0, 0]
        for i in range(3):
            num_pixels = curr_hsv[i].shape[0] * curr_hsv[i].shape[1]
            curr_hsv[i] = curr_hsv[i].astype(numpy.int32)
            last_hsv[i] = last_hsv[i].astype(numpy.int32)
            delta_hsv[i] = numpy.sum(
                numpy.abs(curr_hsv[i] - last_hsv[i])) / float(num_pixels)
        delta_hsv[3] = sum(delta_hsv[0:3]) / 3.0
        delta_h, delta_s, delta_v, delta_hsv_avg = delta_hsv

        if self.stats_manager is not None:
            self.stats_manager.set_metrics(frame_num, {
                metric_keys[0]: delta_hsv_avg,
                metric_keys[1]: delta_h,
                metric_keys[2]: delta_s,
                metric_keys[3]: delta_v})

        self.last_hsv = curr_hsv

    if delta_hsv_avg >= self.threshold:
        if self.last_scene_cut is None or (
            (frame_num - self.last_scene_cut) >= self.min_scene_len):
            cut_list.append(frame_num)
            self.last_scene_cut = frame_num

    if self.last_frame is not None and self.last_frame is not _unused:
        del self.last_frame

# If we have the next frame computed, don't copy the current frame
# into last_frame since we won't use it on the next call anyways.
if (self.stats_manager is not None and
    self.stats_manager.metrics_exist(frame_num+1, metric_keys)):
    self.last_frame = _unused
else:
    self.last_frame = frame_img.copy()

return cut_list

```

Figure 7: Core part of the Algorithm

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 [33]. 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.

2.3.3 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).

2.4 Research Findings

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 8 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.

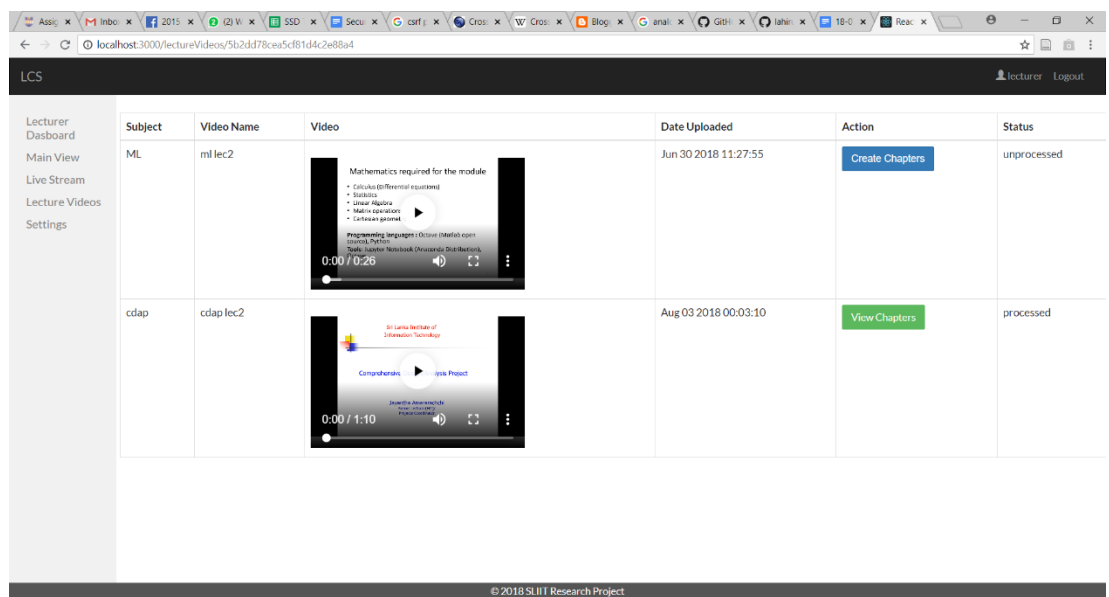


Figure 8: Interface to create or view video chapters

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 9.

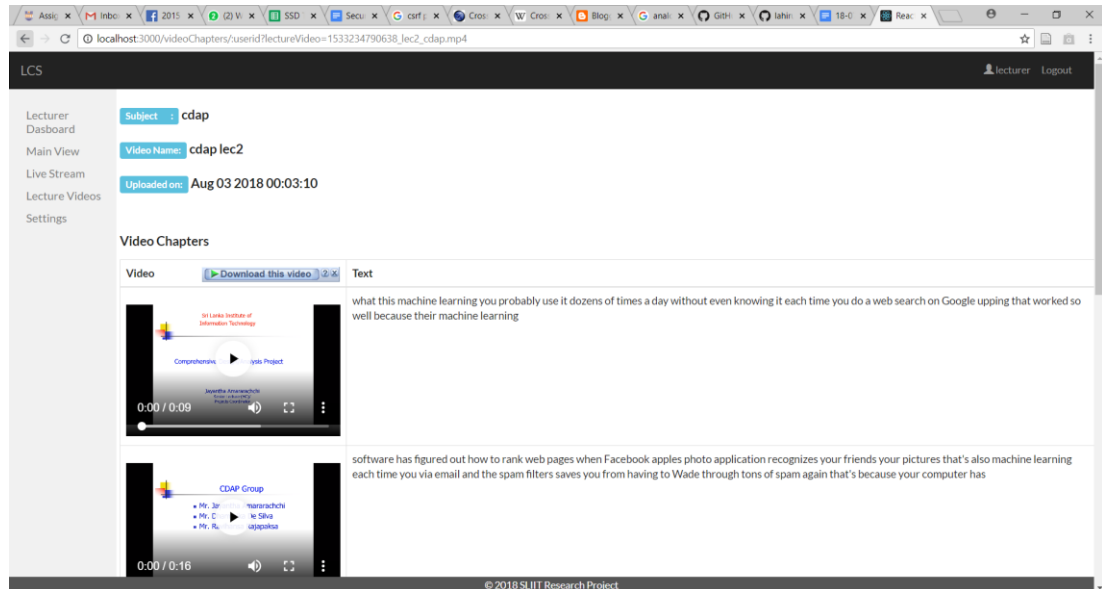


Figure 9: Created video chapters and their text

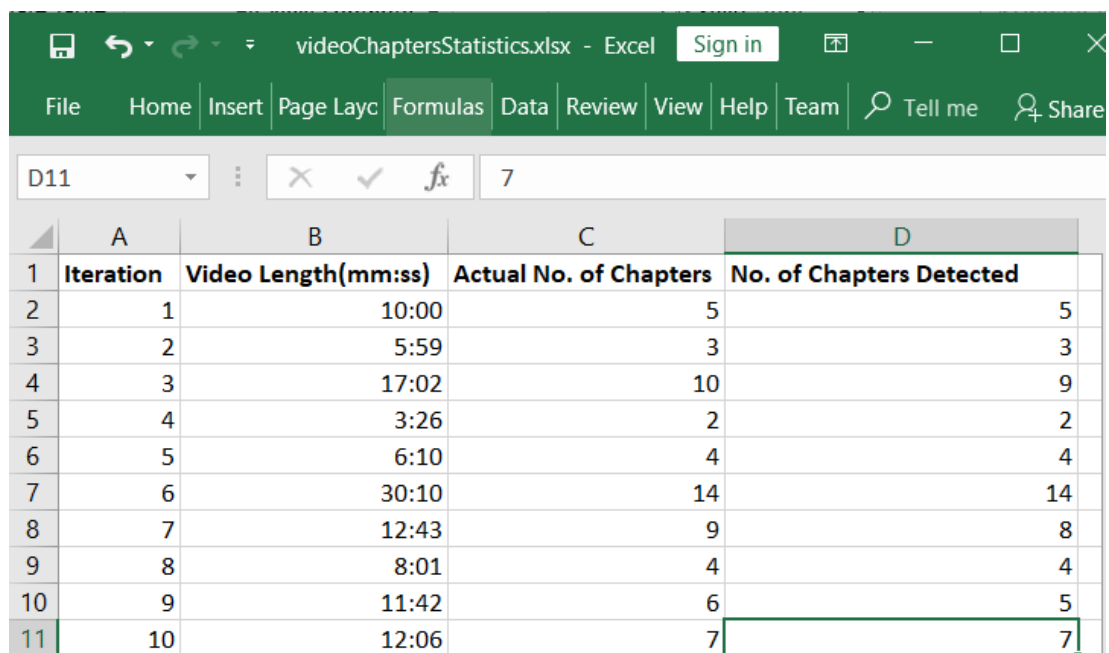
3.2 Discussion

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 were 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 [34]. This ensured that any new changes made did not lead to any bugs in the completed system.

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 10.



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

Figure 10: 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 11.

```
{ "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 11: 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 conferences. In the next stage, in one hand, the research team will be focusing on improving the accuracy of the face recognition and gesture detection models by testing other algorithms effectively. Also, the 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’cal, (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] “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].
- [26] “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].
- [27] “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].
- [28] “About - OpenCV library.” [Online]. Available: <https://opencv.org/about.html>. [Accessed: 07-Oct-2018].
- [29] “ffmpeg Documentation.” [Online]. Available: <https://www.ffmpeg.org/ffmpeg.html>. [Accessed: 07-Oct-2018].
- [30] B. Castellano, *:movie_camera: A Python/OpenCV-based scene detection program, using threshold/content analysis on a given video.: Breakthrough/PySceneDetect*. 2018.
- [31] “Command Reference — PySceneDetect v0.5 documentation.” [Online]. Available: <https://pyscenedetect-manual.readthedocs.io/en/latest/cli/commands.html>. [Accessed: 09-Oct-2018].

- [32] “Introduction - PySceneDetect.” [Online]. Available: <https://pyscenedetect.readthedocs.io/en/latest/>. [Accessed: 08-Aug-2018].
- [33] “Watson Speech to Text,” 28-Nov-2016. [Online]. Available: <https://www.ibm.com/watson/services/speech-to-text/>. [Accessed: 08-Aug-2018].
- [34] “What is Regression Testing? Test Cases, Tools & Examples.” [Online]. Available: <https://www.guru99.com/regression-testing.html>. [Accessed: 10-Oct-2018].

6. APPENDICES

```
#
""" PySceneDetect scenedetect.__main__ Module

Provides entry point for PySceneDetect's command-line interface (CLI)
functionality (in addition to using in other scripts via import scenedetect)
by installing the module and running the scenedetect command, or by calling:

    > python -m scenedetect

This module provides a high-level main() function, utilizing the scenedetect.cli
module, itself based on the click library, to provide command-line interface (CLI)
parsing functionality. Also note that a convenience script scenedetect.py is also
included for development purposes (allows ./scenedetect.py vs python -m scenedetect)

Installing PySceneDetect (using python setup.py install in the parent directory)
will also add the scenedetect command to %PATH% be used from anywhere.
"""

# PySceneDetect Library Imports
import ...

def main():
    """ Main: PySceneDetect command-line interface (CLI) entry point.

    Passes control flow to the CLI parser (using the click library), whose
    entry point is the decorated scenedetect.cli.scenedetect_cli function.
    """

    cli_ctx = CliContext() # CliContext object passed between CLI commands.
    try:
        # pylint: disable=unexpected-keyword-arg, no-value-for-parameter
        cli.main(obj=cli_ctx) # Parse CLI arguments with registered callbacks.
    finally:
        cli_ctx.cleanup()

if __name__ == '__main__':
    main()
```

Figure 12: Main Method

```

#
""" PySceneDetect 'scenedetect.scene_detector' Module

This module implements the base SceneDetector class, from which all scene
detectors in the scenedetect.detectors module are derived from.

The SceneDetector class represents the interface which detection algorithms
are expected to provide in order to be compatible with PySceneDetect.
"""

# pylint: disable=unused-argument, no-self-use

class SceneDetector(object):
    """ Base class to inherit from when implementing a scene detection algorithm.

    Also see the implemented scene detectors in the scenedetect.detectors module
    to get an idea of how a particular detector can be created.
    """

    stats_manager = None
    """ Optional :py:class:`StateManager` <scenedetect.stats_manager.StateManager> to
    use for caching frame metrics to and from."""

    _metric_keys = []
    """ List of frame metric keys to be registered with the :py:attr:`stats_manager`,
    if available. """

    cli_name = 'detect-none'
    """ Name of detector to use in command-line interface description. """

    def is_processing_required(self, frame_num):
        # type: (int) -> bool
        """ Is Processing Required: Test if all calculations for a given frame are already
        done.

        Returns:
            bool: False if the SceneDetector has assigned _metric_keys, and the
            stats_manager property is set to a valid StateManager object containing
            the required frame metrics/calculations for the given frame - thus, not
            needing the frame to perform scene detection.

            True otherwise (i.e. the frame_img passed to process_frame is required
            to be passed to process_frame for the given frame_num).
        """
        return not self._metric_keys or not (
            self.stats_manager is not None and
            self.stats_manager.metrics_exist(frame_num, self._metric_keys))

    def get_metrics(self):
        # type: () -> List[str]
        """ Get Metrics: Get a list of all metric names/keys used by the detector.

        Returns:
            List[str]: A list of strings of frame metric key names that will be used by
            the detector when a StateManager is passed to process_frame.
        """
        return self._metric_keys

    def process_frame(self, frame_num, frame_img):
        # type: (int, numpy.ndarray) -> Tuple[bool, Union[None, List[int]]]
        """ Process Frame: Computes/stores metrics and detects any scene changes.

        Prototype method, no actual detection.

        Returns:
            List[int]: List of frame numbers of cuts to be added to the cutting list.
        """
        return []

    def post_process(self, frame_num):
        # type: (int) -> List[int]
        """ Post Process: Performs any processing after the last frame has been read.

        Prototype method, no actual detection.

        Returns:
            List[int]: List of frame numbers of cuts to be added to the cutting list.
        """
        return []

```

Figure 13: Initializing the Scene Detector

```

if self.last_frame is not None:
    # Change in average of HSV (hsv), (h)ue only, (s)aturation only, (l)uminance only.
    delta_hsv_avg, delta_h, delta_s, delta_v = 0.0, 0.0, 0.0, 0.0

    if (self.stats_manager is not None and
        self.stats_manager.metrics_exist(frame_num, metric_keys)):
        delta_hsv_avg, delta_h, delta_s, delta_v = self.stats_manager.get_metrics(
            frame_num, metric_keys)

    else:
        num_pixels = frame_img.shape[0] * frame_img.shape[1]
        curr_hsv = cv2.split(cv2.cvtColor(frame_img, cv2.COLOR_BGR2HSV))
        last_hsv = self.last_hsv
        if not last_hsv:
            last_hsv = cv2.split(cv2.cvtColor(self.last_frame, cv2.COLOR_BGR2HSV))

        delta_hsv = [0, 0, 0, 0]
        for i in range(3):
            num_pixels = curr_hsv[i].shape[0] * curr_hsv[i].shape[1]
            curr_hsv[i] = curr_hsv[i].astype(numpy.int32)
            last_hsv[i] = last_hsv[i].astype(numpy.int32)
            delta_hsv[i] = numpy.sum(
                numpy.abs(curr_hsv[i] - last_hsv[i])) / float(num_pixels)
        delta_hsv[3] = sum(delta_hsv[0:3]) / 3.0
        delta_h, delta_s, delta_v, delta_hsv_avg = delta_hsv

        if self.stats_manager is not None:
            self.stats_manager.set_metrics(frame_num, {
                metric_keys[0]: delta_hsv_avg,
                metric_keys[1]: delta_h,
                metric_keys[2]: delta_s,
                metric_keys[3]: delta_v})

        self.last_hsv = curr_hsv

    if delta_hsv_avg >= self.threshold:
        if self.last_scene_cut is None or (
            (frame_num - self.last_scene_cut) >= self.min_scene_len):
            cut_list.append(frame_num)
            self.last_scene_cut = frame_num

    if self.last_frame is not None and self.last_frame is not _unused:
        del self.last_frame

# If we have the next frame computed, don't copy the current frame
# into last_frame since we won't use it on the next call anyways.
if (self.stats_manager is not None and
    self.stats_manager.metrics_exist(frame_num+1, metric_keys)):
    self.last_frame = _unused
else:
    self.last_frame = frame_img.copy()

return cut_list

```

Figure 14: Core Part of the Video Splitting Algorithm