**PLATFORM FOR IMPROVING SEARCHABILITY AND INTERACTIVITY OF RECORDED LECTURES**

Project ID: 19-087

Software Requirement Specification

Karunaratne D. C.

Fernando S. S. M. S.

40 mm

20 mm

20 mm

25 mm

Hettiarachchi H. A. I. S.

Epa S. S.

Bachelor of Science (Honors) in Information Technology

Specializing in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

May 2019

**PLATFORM FOR IMPROVING SEARCHABILITY AND INTERACTIVITY OF RECORDED LECTURES**

Project ID: 19-087

Software Requirement Specification

40 mm

20 mm

20 mm

25 mm

Bachelor of Science (Honors) in Information Technology Specializing in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

May 2019

**DECLARATION**

We declare that this is our own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

|  |  |  |
| --- | --- | --- |
| Name | Student ID | Signature |
| Karunaratne D. C. | IT 16037434 |  |
| Hettiarachchi H. A. I. S. | IT 15146366 |  |
| Fernando S. S. M. S. | IT 16001862 |  |
| Epa S. S. | IT 16009646 |  |

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

………………………… …………………….

Signature of Supervisor Date

TABLE OF CONTENTS

[1 Introduction 1](#_Toc8654768)

[1.1 Purpose 1](#_Toc8654769)

[1.2 Scope 1](#_Toc8654770)

[1.3 Definitions, Acronyms and Abbreviations 2](#_Toc8654771)

[1.4 Overview 3](#_Toc8654772)

[2 Overall Description 4](#_Toc8654773)

[2.1 Product perspective 4](#_Toc8654774)

[2.1.1 System interfaces 5](#_Toc8654775)

[2.1.2 User interfaces 6](#_Toc8654776)

[*2.1.3* Hardware interfaces 6](#_Toc8654777)

[2.1.4 Software interfaces 6](#_Toc8654778)

[2.1.5 Communication interfaces 7](#_Toc8654779)

[2.1.6 Memory constraints 7](#_Toc8654780)

[2.1.7 Operations 7](#_Toc8654781)

[2.1.8 Site adaptation requirements 7](#_Toc8654782)

[2.2 Product functions 8](#_Toc8654783)

[2.3 User characteristics 13](#_Toc8654784)

[2.4 Constraints 13](#_Toc8654785)

[2.5 Assumptions and dependencies 14](#_Toc8654786)

[2.6 Apportioning of requirements 14](#_Toc8654787)

[3 Specific requirements 15](#_Toc8654788)

[3.1 External interface requirements 15](#_Toc8654789)

[3.1.1 User interfaces 15](#_Toc8654790)

[3.1.2 Hardware interfaces 19](#_Toc8654791)

[3.1.3 Software interfaces 19](#_Toc8654792)

[3.1.4 Communication interfaces 19](#_Toc8654793)

[3.2 Classes/Objects 19](#_Toc8654794)

[3.3 Performance requirements 20](#_Toc8654795)

[3.4 Design constraints 20](#_Toc8654796)

[3.5 Software system attributes 20](#_Toc8654797)

[3.5.1 Reliability 20](#_Toc8654798)

[3.5.2 Availability 20](#_Toc8654799)

[3.5.3 Security 21](#_Toc8654800)

[3.5.4 Maintainability 21](#_Toc8654801)

[4 Supporting information 22](#_Toc8654802)

[4.1 References 22](#_Toc8654803)

[4.2 Appendix A: Activity Diagrams 23](#_Toc8654804)

[4.3 Appendix B: High level architecture diagrams 27](#_Toc8654805)

**LIST OF FIGURES**

[Figure 2.1: Use case diagram 8](https://mysliit.sharepoint.com/sites/cdap-2019-lecturevideos/Shared%20Documents/Documents/19-087_SRS.docx#_Toc8654806)

[Figure 3.1: User interface for home screen 15](#_Toc8654807)

[Figure 3.2: User interface for upload content 16](#_Toc8654808)

[Figure 3.3: User interface for topic segment adjusting 16](#_Toc8654809)

[Figure 3.4: User interface for video player 17](#_Toc8654810)

[Figure 3.5: User interface for navigate video using code samples 17](#_Toc8654811)

[Figure 3.6: User interface for answering question in video playback 18](#_Toc8654812)

[Figure 3.7: User interface for navigate using slides 18](#_Toc8654813)

[Figure 4.1: Activity diagram for question generation and selection 23](#_Toc8654814)

[Figure 4.2: Activity diagram for lecture video segmentation 24](https://mysliit.sharepoint.com/sites/cdap-2019-lecturevideos/Shared%20Documents/Documents/19-087_SRS.docx#_Toc8654815)

[Figure 4.3: Activity diagram for navigating from lecture slides 25](#_Toc8654816)

[Figure 4.4: Activity diagram for matching code samples 26](#_Toc8654817)

[Figure 4.5: Activity diagram for navigating using code samples 27](#_Toc8654818)

[Figure 4.6: Cloud architecture 27](#_Toc8654819)

[Figure 0.1: High-level system architecture 28](https://mysliit.sharepoint.com/sites/cdap-2019-lecturevideos/Shared%20Documents/Documents/19-087_SRS.docx#_Toc8654820)

**LIST OF TABLES**

[Table 1.1: Definitions, Acronyms and Abbreviations 2](#_Toc8654821)

[Table 2.1: Comparison of existing products 5](#_Toc8654822)

[Table 2.2: Use case scenario for upload lecture video and materials 9](#_Toc8654823)

[Table 2.3: Use case scenario for navigate video using code sample 10](#_Toc8654824)

[Table 2.4: Use case scenario for segment video into topic units 11](#_Toc8654825)

[Table 2.5: Use case scenario for navigating using lecture slides 11](#_Toc8654826)

[Table 2.6: Use case scenario for generate questions and answers 12](#_Toc8654827)

# Introduction

## Purpose

The purpose of this document is to provide a detailed overview of the proposed system, Platform for Improving Searchability and Interactivity of Recorded Lectures which will be completed at the end of the research project. This document is expected to improve the user’s understanding of the system by addressing the purpose, scope, overview, dependencies, assumptions and the constraints under which the system will give the expected performance etc. which would ultimately develop the intended audiences’ knowledge about the platform. This document acts as the key reference when developing the proposed e-learning platform to ensure that all the requirements that were identified during the background study were addressed. In addition, this document can be used for decision making in every phase of the software development life cycle, which would minimize the development cost at the end. Therefore, this document will act as the main reference for software developers, testers and the clients. Moreover, this document is written using non-technical language, so that it can be used by individuals of varying technical backgrounds. As a result, this can be used as a reference for future research in the same domain.

## Scope

This document covers the requirements for the proposed Platform for Improving Searchability and Interactivity of Recorded Lectures. The software product will follow a Software as a Service (SaaS) model. Current architectural styles and best practices in the field of cloud-computing will be considered when designing the application. The application will be hosted by an appropriate third party and delivered to the end users over the internet as a web application. There are four main components of the system are as follows,

* Code extraction and source code matching component.
* Automatic Topic Segmentation component
* Automatic Question Generation component
* Audio enhancement and lecture slide matching component

The components stated above will be combined to create a web-based platform which will enable lecturers and content creators to enhance the learning experience delivered by their material. The proposed platform will have the following objectives.

* Automatically tag specific points in the video which correspond to a given lecture slide or line of code in a given supporting code sample.
* Automatically identify main topic transitions of a given lecture video and segment the video according to the identified positions.
* Automatically generate question items related to the lecture video and present to end user.
* Automatically remove unnecessary background noises and enhance the quality of the audio.

The specific focus of the proposed platform will be the enhancement of lecture videos. The scope will be limited to that which will achieve the objectives stated above.

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| ADSL | Asymmetric Digital Subscriber Line |
| FTTH | Fiber to The Home |
| HSDPA | High-Speed Downlink Packet Access |
| SaaS | Software as a Service |
| SAST | Static Application Security Testing |

Table .: Definitions, Acronyms and Abbreviations

## Overview

The main objective of this research is to develop an automated system which can add interactivity and accessibility to course materials with minor human intervention thereby improving the overall user experience of both the learner as well as the course creator (lecturers). Primary users of this system will be lecturers, who wish to create interactive course materials, and students/learners who will utilize these materials for learning. Hence, the initial release of our product will be deployed as a web platform using some of the cloud technologies and industry best practices.

This SRS will cover in detail all the functional and non-functional requirements of the proposed system. Details will be spread across three chapters, each covering a different perspective of the system. The first chapter gives an overview of the whole system, along with a brief description about the purpose and the scope of this document.

The second chapter will present and in-depth overview of the system from the user’s perspective. Sections such as product perspective, product functions, user characteristics and constraints under which the system will operate, will be covered under this chapter. In the first sub-section ‘product perspective’, the proposed system will be compared with existing systems. It then moves on to describe details about several interfaces of the system, memory constraints and operation of users to provide a better understanding about the product to the readers. This chapter will finally conclude with apportioning of requirements.

The third and the last chapter of this SRS will focus on specific requirements of the product and is primarily written for developers. This chapter will describe about the functionalities mentioned in chapter two in a more detail and gives a technical perspective to the system. The latter part of this chapter provides details about the system attributes such as reliability, maintainability, availability and security.

Both chapters two and three describe the whole system but in two different perspectives and is intended for two different types of target audiences. The document finally concludes providing supporting information regarding the contents of the document.

# Overall Description

Recorded lectures have gained popularity as a method of delivering lecture content as this approach gives learners distinct advantages such as the ability to follow lectures without time or location constraints and consume the lectures at their own pace [1]. Although recorded lectures have many advantages, they tend to be lengthy and tedious to watch. They also prove cumbersome when specific information needs to be extracted from them. Another drawback is that the lecture videos fail to show the connection between the lecture and its supporting material such as lecture slides and questionnaires. Platforms such as LearnWorlds[[1]](#footnote-2), Echo360[[2]](#footnote-3) and Techsmith Relay [[3]](#footnote-4)allow videos to be edited to make them more interactive however the methods employed by these platforms are manual and therefore tedious and time consuming. Furthermore, at the time of writing this document there is no platform which automatically identifies relationships between different types of course material such as source code files and lecture slides and ties them into the lecture video.

The *Platform for Improving Searchability and Interactivity of Recorded Lectures* aims to address these drawbacks by introducing an automated process of identifying and combining the lecture material to create an enhanced user experience. The platform will be developed using a cloud microservices architecture which holds many benefits. Building the product as a collection of independent microservices not only facilitates an easier development process but also allows for a more robust system which has minimum downtime [2]. By utilizing the power of cloud computing, the application can be up-scaled and down-scaled with ease depending on the load.

## Product perspective

The background study and literature review carried out for the project proposal revealed several notable solutions which have a similar feature set and similar goals and objectives. Table 2.1: Comparison of existing products shows a comparison of features which are common across the platforms along with those that are improved in the proposed platform.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Features | LearnWorlds | Echo360 | TechSmith Relay | Our Solution |
| Matching lines in code samples to occurrence in recorded lectures | ✗ | ✗ | ✗ | ✔ |
| Automated segmentation of lecture video into topic units | ✗ | ✗ | ✗ | ✔ |
| Matching slides with the lecture video | ✗ | ✔ | ✗ | ✔✔ |
| Automated noise removal from the video | ✗ | ✗ | ✗ | ✔ |
| Automatic question generation | ✗ | ✗ | ✗ | ✔ |
| Embedding questions in video playback | ✔ | ✔ | ✔ | ✔✔ |
| Automated question embedding in video playback | ✗ | ✗ | ✗ | ✔ |

Table .: Comparison of existing products

✗ : Not available

✔ : Available

✔✔ : Improved

As illustrated in Table 2.1: Comparison of existing products, there is a significant research gap which needs to be addressed. Although similar existing products are available, the proposed system will improve on the features of these systems as well as introduce new features and functionality.

### System interfaces

* Python 3 sdk
* PyMongo driver
* OWL API

### User interfaces

Section 2.1.2 lists the user interfaces which will be available along with a brief description of each one. Section 3.1.1 will contain a more detailed description of each user interface.

* Platform Homepage: *list of available lecture videos with a small introduction to the platform.*
* Code finder interface: *source code is shown alongside the lecture video and clicking a line of code will highlight the timestamps where the code occurs in the video.*
* Interface for topic segmentation: *Uploaded lecture video can be submitted for topic segmentation using this interface*
* Interface for modifying segmented videos: *This interface allows lecturers to change the auto suggested topics from the system and the split points before actually segmenting the video*
* Interface for uploading course materials
* Interface for slide matching: *Electronic slides related to the lecture video is displayed and when the slide is selected the system will point to the time frames where the selected slide is discussed.*
* Interface for selecting and modifying questions and answers: *automatically generated questions and answers will be showed to the lecturer where he/she can change the content and select which of them to be embedded to the video playback*

### Hardware interfaces

No specialized hardware is required. All processing will be carried out in the cloud. A regular computer with an internet connection will be required to access the web application.

### Software interfaces

* Node.js runtime
* Ffmpeg – video and audio manipulation
* Keras machine learning library
* OpenCV computer vision library
* Amazon Web Services (AWS)
* WebStorm
* Protégé

### Communication interfaces

A high-speed internet connection is preferred (ADSL, HSDPA, 4G LTE or FTTH).

### Memory constraints

* 4GB of RAM

### Operations

* Login to the site
* Upload lecture videos and supporting material
* Segment long lecture videos into topics
* Generate questions based on the content in the video
* Enhance the lecture audio
* Search for a specific lecture.
* Use supporting material such as lecture slides and source code to jump to a specific frame in the lecture video.

### Site adaptation requirements

The product will be delivered across the web as a SaaS product. Since the end user interacts with a web-interface and all processing is carried out in the cloud, a JavaScript enabled web browser is all that is required.

## Product functions

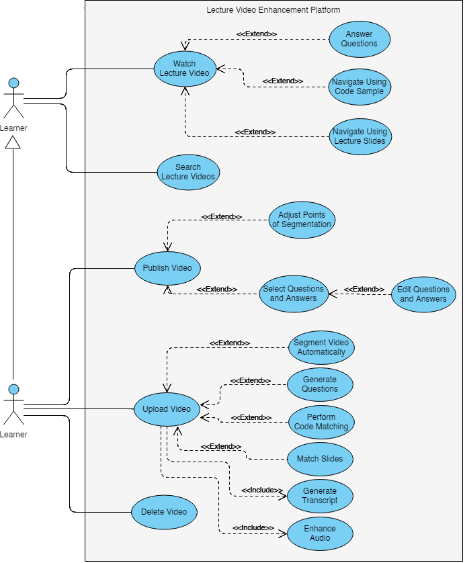
The product functions will be explained with the help of a use case diagram Figure 2.1: Use case diagram and the relevant Use case scenarios Table 2.2 - Table 2.6.

Figure 2.1: Use case diagram

|  |  |
| --- | --- |
| **Use case name** | Upload a lecture video and reference materials |
| **Pre-Conditions** | Users need to login to the system using valid credentials. |
| **Post-Conditions** | Upload video and lecture materials to the platform |
| **Actor** | Lecturer/ course creators |
| **Main success scenarios** | 1. Use case starts with user logging into the system. 2. User navigate to upload content page. 3. User clicks ‘upload video’ button 4. User select video files from the computer 5. System displays the uploading progress. 6. User clicks the ‘upload content’ button. 7. User select lecture slides from the computer. 8. System displays the uploading progress. 9. System displays uploaded video and lecture slides. |
| **Extensions** | 5a. Upload fails and system displays error message.  8a. Upload fails and system displays error message. |

Table .: Use case scenario for upload lecture video and materials

|  |  |
| --- | --- |
| **Use case name** | Navigate video using code sample |
| **Pre-Conditions** | User must be logged into the system using valid credentials. |
| **Post-Conditions** | - |
| **Actor** | Lecturer/ course creators |
| **Main success scenarios** | 1. Use case starts with user logging into the system. 2. User navigates to the search lecture videos page. 3. User selects the desired lecture video to watch 4. User goes to the ‘code finder’ tab on the video details page. 5. System displays the source code alongside the lecture video. 6. User clicks on a line in the source code which is displayed. 7. System displays the list of timestamps where the line of code is found in the video. 8. User clicks on a timestamp from the list. 9. System goes to the specific timestamp in the lecture video. |
| **Extensions** | 5a. System displays “No source code file uploaded” in the source code section.  6a. System displays “Line not found in video” message |

Table .: Use case scenario for navigate video using code sample

|  |  |
| --- | --- |
| **Use case name** | Segment video into topic units |
| **Pre-Conditions** | Users need to login to the system using valid credentials.  There should be an already uploaded lecture video |
| **Post-Conditions** | Set of topically segmented lecture videos |
| **Actor** | Lecturer/ course creators |
| **Main success scenarios** | 1. Use case starts with user logging into the system. 2. User selects the video to be segmented 3. User clicks the ‘Segment into topics’ button. 4. System displays the ‘processing.......’ progress bar. 5. System displays the segmentation points of the video. 6. User clicks the ‘Proceed’ button. 7. System displays the segmented videos |
| **Extensions** | 6a. User replays the video at segmentation points.  6b. User changes the segmentation points manually and clicks the ‘Proceed’ button. |

Table .: Use case scenario for segment video into topic units

|  |  |
| --- | --- |
| **Use case name** | Navigating using lecture slides |
| **Pre-Conditions** | User is logged in to the system with valid login credentials  Lecture videos are already uploaded to the system |
| **Actor** | Student |
| **Post Conditions** | Point to the time frame where the selected slide is discussed. |
| **Main flow** | 1. Use case starts when the user logs in to the system using valid login credentials. 2. User selects the lecture video 3. User selects the Slides tab 4. User selects a slide from the slider displayed 5. User clicks on the ‘Proceed’ button 6. The use case ends when the system points to the time frame in the lecture video where the selected slide is discussed. |
| **Extensions** | 3a. The electronic lecture slides of the selected lecture video is not uploaded to the system, the user is given a message. |

Table .: Use case scenario for navigating using lecture slides

|  |  |
| --- | --- |
| **Use case name** | Generate questions and answers from lecture content |
| **Pre-Conditions** | Users need to login to the system using valid credentials.  There should be an already uploaded lecture video and reference materials |
| **Post-Conditions** | Embed questions and answers to video playback |
| **Actor** | Lecturer/ course creators |
| **Main success scenarios** | 1. Use case starts with user logging into the system. 2. User selects the video to generate questions and answers. 3. User clicks the ‘proceed’ button. 4. System displays the ‘processing.......’ progress bar. 5. System displays the questions and answers generated. 6. User clicks the ‘Proceed’ button. 7. System displays the video with questions embedded. |
| **Extensions** | 6a. User changes the question and answers. |

Table .: Use case scenario for generate questions and answers

## User characteristics

The target audience of our system is universities and higher educational institutes who wish to provide course materials for their students online. Therefore, our users fall into 2 main categories:

* + Lecturers
  + Students

Lecturers will use this system to upload course materials, and make it more interactive for the students, whereas students will use the system for learning and studying purposes. Both the categories of users do not require any prior technical knowledge besides basic interactions with a web application.

## Constraints

* Angular.js will be used to develop frontend client application while the backend processes are developed primarily using node.js and machine learning frameworks.
* The backend of the system will follow a microservices architecture where several components of the backend will be developed independent of each other.
* The developers of each microservice will be free to choose whatever technology stack suits their purpose.
* Internet connectivity is a must to interact between the frontend and backend processes.
* Backend system must have minimum of 4GB RAM.
* After performing the necessary analysis on the uploaded video, the final optimized video shall not be stored on the system internally. All temporary files will be removed after the final video is hosted on a suitable video hosting platform.

## Assumptions and dependencies

* Initial release of platform will be targeted for desktop web browsers. Future releases shall introduce mobile friendly design.
* All browsers are assumed to be capable of running the frontend JavaScript application.
* Only screen recorded lectures (voice over presentation) will be supported for the initial release. Future releases will support more lecture video patterns such as talking head, lecture capture, etc.

## Apportioning of requirements

* Essential Requirements:

1. The content creator should be able to upload a lecture video and source code file so that it can be processed by the system.
2. The learner should be able to navigate the lecture video using the lines of code in a source code file as the reference.
3. The learner should be able to navigate the lecture video using the lecture slides.
4. The system should automatically generate and suggest suitable questions for each lecture video.
5. The system should automatically suggest segmentation points where the video can be broken down into smaller sections.

* Desirable Requirements:

1. Generate data on popularity of lecture videos.
2. Have a feedback system where students can reach the lecturer who created the videos.
3. Generate statistical report for lectures and how learners engage with them.
4. Maintain a student profile and grading system.

* Optional Requirements:

1. Create a search function to search through the indexed video file for code occurrences.

# Specific requirements

## External interface requirements

### User interfaces

The description of user interface in section 2.1.2 showed a basic description of the interfaces to provide product perspective. The interfaces mentioned under section 3.1.1 provide a more detailed representation of the required interfaces and are intended for the developers of the system.

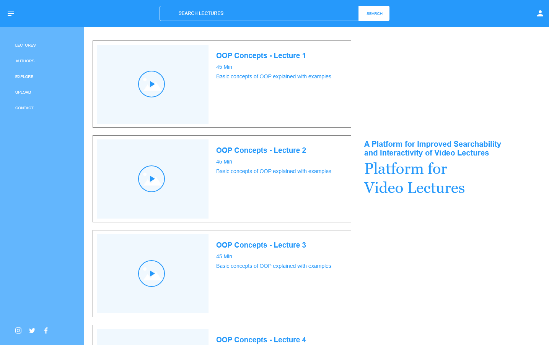


Figure 3.1: User interface for home screen

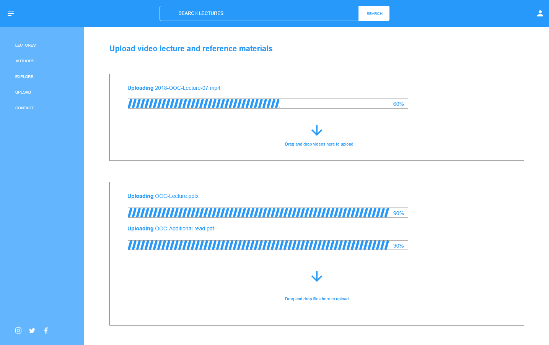


Figure 3.2: User interface for upload content

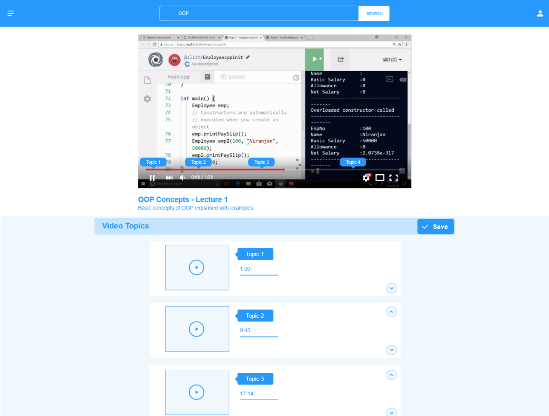


Figure 3.3: User interface for topic segment adjusting

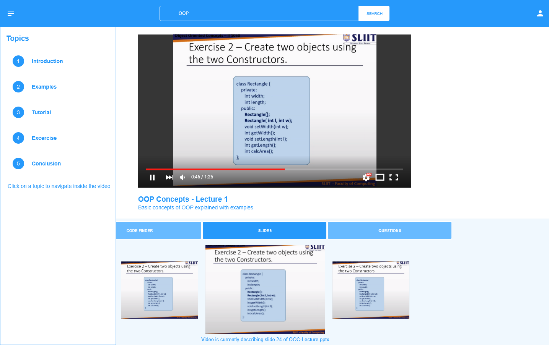


Figure 3.4: User interface for video player

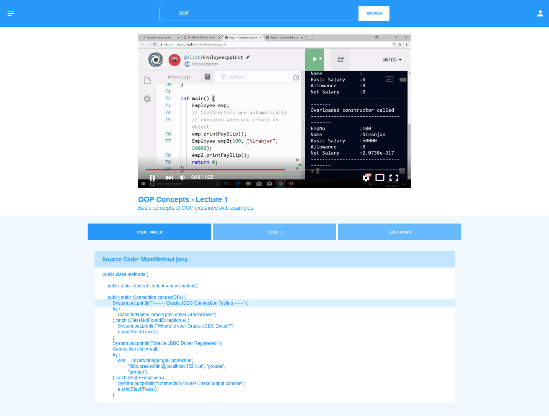


Figure 3.5: User interface for navigate video using code samples

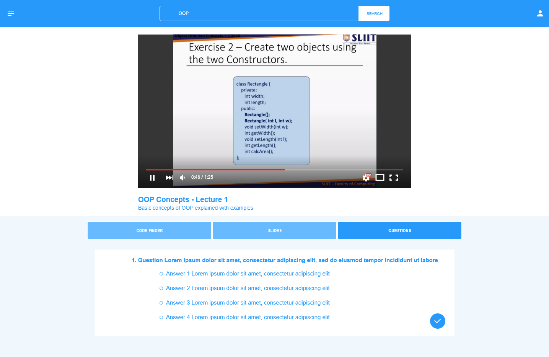


Figure 3.6: User interface for answering question in video playback

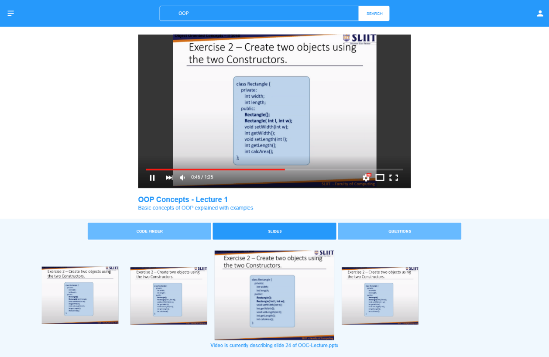


Figure 3.7: User interface for navigate using slides

### Hardware interfaces

No specialized hardware is required for the backend. A regular computer with an internet connection and web browser capable of running JavaScript will be required to access the web application.

### Software interfaces

* Node.js runtime
* Ffmpeg – video and audio manipulation
* Keras machine learning library
* OpenCV computer vision library
* Amazon Web Services (AWS)
* WebStorm
* Protégé

### Communication interfaces

A stable high-speed internet connection is required.

## Classes/Objects

* Essential
  + MediaController – To handle upload and streaming of video and other media content.
  + Authenticator – To generate authentication tokens for valid user credentials.
  + VideoPreProcessor – Handle initial processing of uploaded videos
  + ServiceController – Apply the required algorithm or service when called.
* Optional
  + DataAnalyzer – To Generate analysis reports

## Performance requirements

The proposed platform is expected to run on a standard desktop computer or a laptop computer with minimum requirements to access internet. Considering functional requirements following performance requirements are identified.

* Platform should support minimum 1000 simultaneous requests.
* Loading a webpage should not take more than 4000 milliseconds
* Once a page is loaded, each user interaction should not have a latency of more than 500 milliseconds.
* Video quality should be at least 480p

## Design constraints

* The proposed system is mainly focused on playing videos to the users. User interfaces of the platform should be focused on enabling the interactivity and searchability provided by backend processes.
* Architectural design must emphasis on parallel processing since computational heavy tasks of the platform will take significant amount of time to complete.

## Software system attributes

### Reliability

The system should be implemented using tried and tested libraries wherever possible to reduce the risk of failure. A failure will be defined as a software defect which will cause the entire system to be unavailable to end users. Each microservice should have at least 200 hours between each failure.

### Availability

Availability is one of the major system attributes that is considered in the modern world. It can be simply defined as the probability that the system is functioning properly when requested for use. Availability is also a key factor in the proposed system because the system needs to be available for use whenever requested by the lecturers and students. As the backend will be deployed in a cloud environment, the availability of the system will depend on the cloud provider’s services and performance. However, we expect the system to be available about 90% of the time.

### Security

Suitable encryption methods should be used to encrypt all data generated by the system. Special attention should be given to to maintain the confidentiality of the Users’ personal information if it is stored. The user should be authenticated using a suitable framework before utilizing the system. All passwords must be stored as hashed values. For internal system communication each call which is transmitted to a service over the internet must use token-based authentication based on the OAuth or OAuth2 standard [3]. Furthermore Static Application Security Testing (SAST) methods [4] should be employed in the continuous integration pipeline to identify security vulnerabilities in the source code.

### Maintainability

Maintainability can be defined as the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. The system is divided into functionally cohesive modules to improve maintainability. Also, we practice software engineering best practices throughout this research, which will improve the overall maintainability of the system. The architecture of the system will be deployed in a manner that will support high maintainability in the cloud environment in which it is deployed.

# Supporting information

## References

|  |  |
| --- | --- |
| [1] | "Importance and Effectiveness of E-learning," 1 December 2015. [Online]. Available: https://higheredrevolution.com/importance-and-effectiveness-of-e-learning-9513046ed46c. [Accessed 6 March 2019]. |
| [2] | "Benefits of Microservices - Microservices on AWS," Docs.aws.amazon.com, 2019. [Online]. Available: https://docs.aws.amazon.com/aws-technical-content/latest/microservices-on-aws/benefits-of-microservices.html. [Accessed 20 April 2019]. |
| [3] | "OAuth 2.0 — OAuth," Oauth.net, 2019. [Online]. Available: https://oauth.net/2/. [Accessed 13 May 2019]. |
| [4] | "Static Application Security Testing (SAST) - Gartner IT Glossary," Gartner IT Glossary, 2019. [Online]. Available: https://www.gartner.com/it-glossary/static-application-security-testing-sast. [Accessed 13 May 2019]. |

## Appendix A: Activity Diagrams

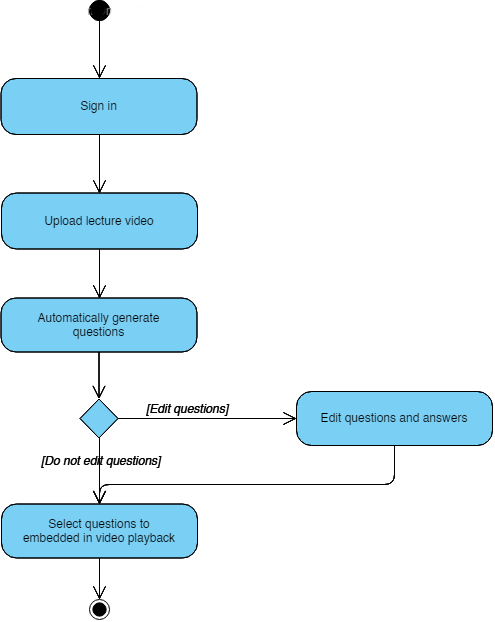


Figure 4.1: Activity diagram for question generation and selection

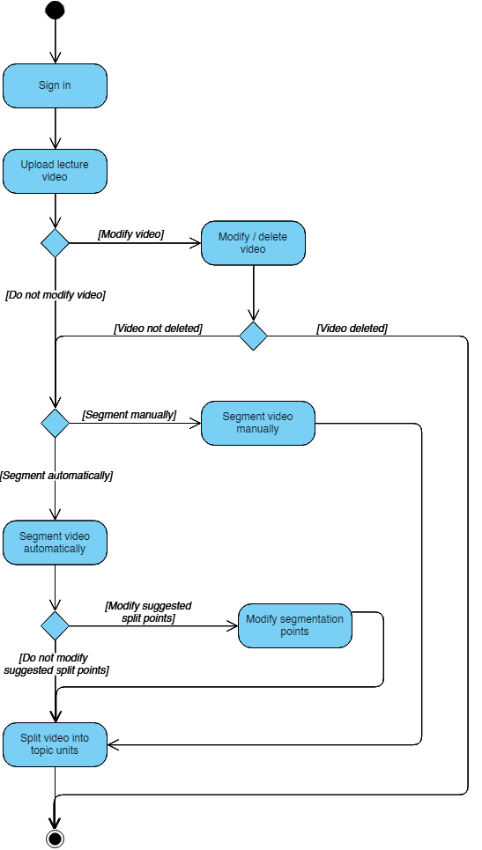


Figure 4.2: Activity diagram for lecture video segmentation

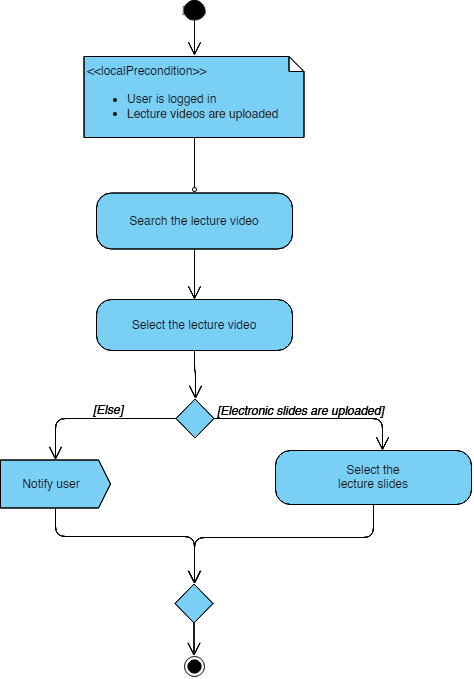


Figure 4.3: Activity diagram for navigating from lecture slides

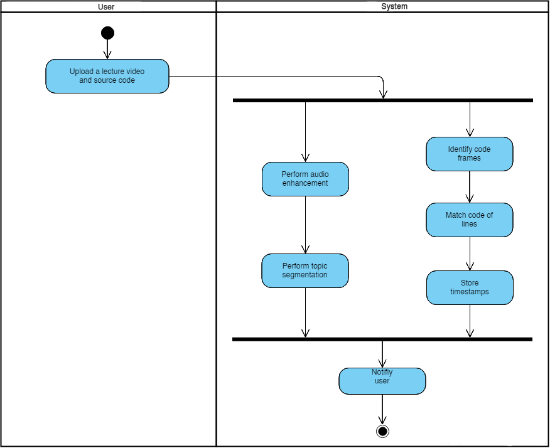


Figure 4.4: Activity diagram for matching code samples

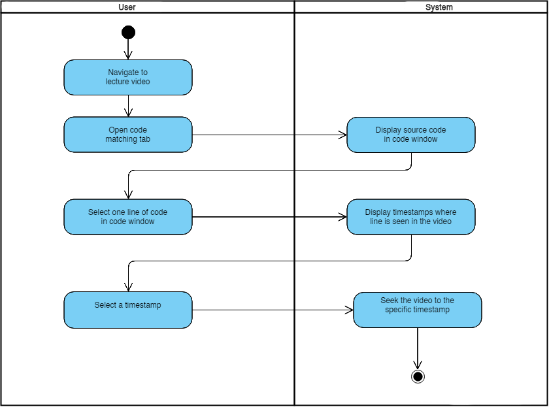


Figure 4.5: Activity diagram for navigating using code samples

## Appendix B: High level architecture diagrams

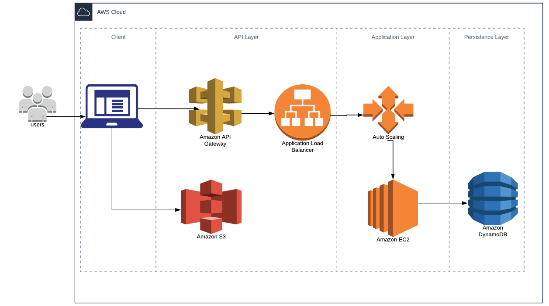


Figure 4.6: Cloud architecture



Unedited lecture video, slides and reference materials

Noise removal and optimizing

Optimized Video

Transcript

Code Matching

Question Generation

Slide Matching

Topic Segmentation

Metadata



Question Items

Proposed System

Processes

Database

Web-Platform

Figure 0.1: High-level system architecture

1. https://www.learnworlds.com/ [↑](#footnote-ref-2)
2. https://www.echo360.com/about/ [↑](#footnote-ref-3)
3. https://www.techsmith.com [↑](#footnote-ref-4)