Bansilal Ramnath Agarwal Charitable Trust’s

**Vishwakarma University**

Pune – 411 048

**Design Document**

**On**

**VIDEO SUMMARIZATION**

###### **Submitted By**

**Pritesh Deepak Singhvi 201700101 (1780030053)**

**Paras Satish Karnawat 201700064 (1780030016)**

**Rinki Rajendra Ray 201700092 (1780030044)**

**Ritu Rajendra Ray 201700093 (1780030045)**

**Sumit Kishor Jeswani 201700060 (1780030012)**

**Bachelor of Technology (Computer Engineering)**

**School of Science and Technology**

**Year 2017 – 2021**

#### BRACT’s

Vishwakarma University

Survey No. 2/3/4 Kondhwa(Bk.) , Pune – 411048

Department of Computer Engineering

C E R T I F I C A T E

This is to certify that the project entitled

**VIDEO SUMMARIZATION**

Submitted by

Pritesh Deepak Singhvi

Paras Satish Karnawat

Rinki Rajendra Ray

Ritu Rajendra Ray

Sumit Kishor Jeswani

is a bonafide work carried out by them under the supervision of Prof. Prasad Gokhale and it is approved for the partial fulfillment of the requirement of Vishwakarma University for the award of the degree of Bachelor of Engineering (Computer Engineering) in the academic year 2017 - 2021.

Prof. Prasad Gokhale Head of Department

(Project Guide) (Computer Engineering)

**Table of Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| Table of Contents | | | |
| 1 |  |  | **Introduction** |
|  | 1.1 |  | Purpose |
|  | 1.2 |  | Scope |
|  | 1.3 |  | Definitions, Acronyms, and Abbreviations |
| 2 |  |  | **Literature Survey** |
| 3 |  |  | **System Design** |
|  | 3.1 |  | System Architecture |
|  | 3.2 |  | System Modules |
|  | 3.3 |  | User Requirements |
|  | 3.4 |  | Functional Requirements |
|  |  | 3.4.1 | General Requirements |
|  |  | 3.4.2 | Video to Summary Requirements |
|  |  | 3.4.3 | Presentation Requirements |
|  | 3.5 |  | Non-functional Requirements |
|  | 3.6 |  | Constraints and Assumptions |
|  | 3.7 |  | Data Flow Diagram |
| 4 |  |  | **Detail Design** |
|  | 4.1 |  | UML Diagrams |
|  |  | 4.1.1 | Use Case Diagram |
|  |  | 4.1.2 | Class Diagram |
|  |  | 4.1.3 | Component Diagram |
|  |  | 4.1.4 | Sequence Diagram |
| 5 |  |  | **Testing** |
|  | 5.1 |  | Test Objective |
|  | 5.2 |  | Detailed Testing Strategy |
|  |  | 5.2.1 | Unit Testing |
|  |  | 5.2.2 | White Box Testing |
|  |  | 5.2.3 | Black Box Testing |
|  |  | 5.2.4 | Pass/Fail Criteria |
|  | 5.3 |  | Test Cases |
|  | 5.4 |  | Test Tools |
| 6 |  |  | Planning |
| 7 |  |  | Conclusion |
| 8 |  |  | References |

List of Figures

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Figure Name** | **Page No.** |
| 1 | System Architecture | 9 |
| 2 | Data Flow Diagram | 12 |
| 5 | Use Case Diagram | 13 |
| 6 | Class Diagram | 14 |
| 7 | Component Diagram | 17 |
| 8 | Sequence Diagram | 18 |

List of Tables

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Table Title** | **Page No.** |
| 1 | Use Case Diagram Table | 13 |
| 2 | Project Planning | 21 |

# 

# **1. Introduction**

## **1.1. Purpose**

Video Summarization is a process of creating & presenting a meaningful abstract view of the entire video within a short period of time.The Purpose of video summarization is to speed up browsing of a large collection of video data, and achieve efficient access and representation of the video content. By watching the summary, users can make quick decisions on the usefulness of the video.

Summarizing news videos automatically allows us to quickly look out for the important patterns shown in the news. Generating a trailer of a movie. Moreover, the need for surveillance has increased significantly due to increase in the demand of security and highlights of sports video recordings automatically are some of the engrossing applications of video summarization.

It helps in efficient storage, quick browsing, and retrieval of large collections of video data without losing important aspects.

## **1.2. Scope**

For movie summarization, the system is extracting each frame and comparing the image intensity using image histogram, when it will detect any change in the intensity, it merges the frames to get the desired summarized video. For highlights of sports, a threshold has been set based on audience volume, then the system will extract the frames in every five seconds and merge the frames which have crossed the threshold to get the desired highlight of any sport.

We have not used machine learning for video summarization because training the sheer volume of video data is time-consuming, which is not efficient for users.

End users include:

1. Sport enthusiasts who would like to see a game within a short amount of time looking for the key moments.
2. Movie enthusiasts who would like to take a glimpse of what the movie contains to decide for themselves whether to watch the complete movie or not.
3. Detecting key scenes helps the editors to ease out their work of highlights or trailer generation.

## **1.3. Definitions, Acronyms, and Abbreviations**

1. GUI: Graphical User Interface

# 

# **2. Literature Survey**

Taru et al., 2017 propose a new method of video summarization into text. This approach takes an input of consumed video. The aim of this research is for a user to easily understand the summary in text format. Srinivas et al., 2016 proposed an improved method for video summarization techniques. The aim is to get a summary content of a video which is interesting to the viewer and represents the whole video.

Yair Shemer, Daniel Rotman, and Nahum Shimkin proposed ILS-SUMM: Iterated Local Search For Unsupervised Video Summarization, 2020. Its objective is to automatically create a short summary of the whole contents. Moreover, to indicate the high scalability of ILS-SUMM, the authors introduce a new dataset consisting of videos of various lengths. Zhou et al., 2018 develop a deep summarization network (DSN) to summarize videos for predicting each video frame probabilities. The training is an end to end reinforcement learning so the result is better than that of supervised approaches. Cai et al., 2018 proposed a generative modeling framework to learn representation with a variational autoencoder. Encoder-decoder attention for saliency estimation of raw video for generating the summary.

Song et al., 2015 present TvSum an unsupervised summarization framework. Introduce a new benchmarks dataset. This approach produces superior quality summaries compared with other approaches. Yuan et al.,2017 present a novel Deep Side Semantic Embedding (DSSE) model to generate video summaries. In semantic relevance can be more effectively measured. Fu et al., 2019 proposed a GAN-based training framework through an unsupervised and supervised video summarization approach. The generator is focused on Ptr-Net that generates the cutting points of summarized fragments.

Chu et al., 2015 developed a Maximal Biclique Finding (MBF) algorithm that is optimized to find sparsely co- occurring patterns. The results suggest that summaries generated by visual co-occurrence tend to match more closely with human-generated summaries. Agyeman et al.,2019 presents a deep learning approach to summarizing long soccer videos which are three-dimensional Convolutional Neural Network (3D-CNN) and Long Short-term Memory (LSTM) Recurrent Neural Network (RNN). Fajtl et al. 2019 propose a novel method for supervised bi- directional recurrent networks such as BiLSTM combined with attention. Elfeki et al. 2019 conduct extensive experiments on the compiled dataset in addition to three other standard benchmarks.

However, all of the methods except that of ILS-SUMM tend to be resource heavy and are slow. ILS-SUMM also tends to work slow on larger videos.

# 

# 

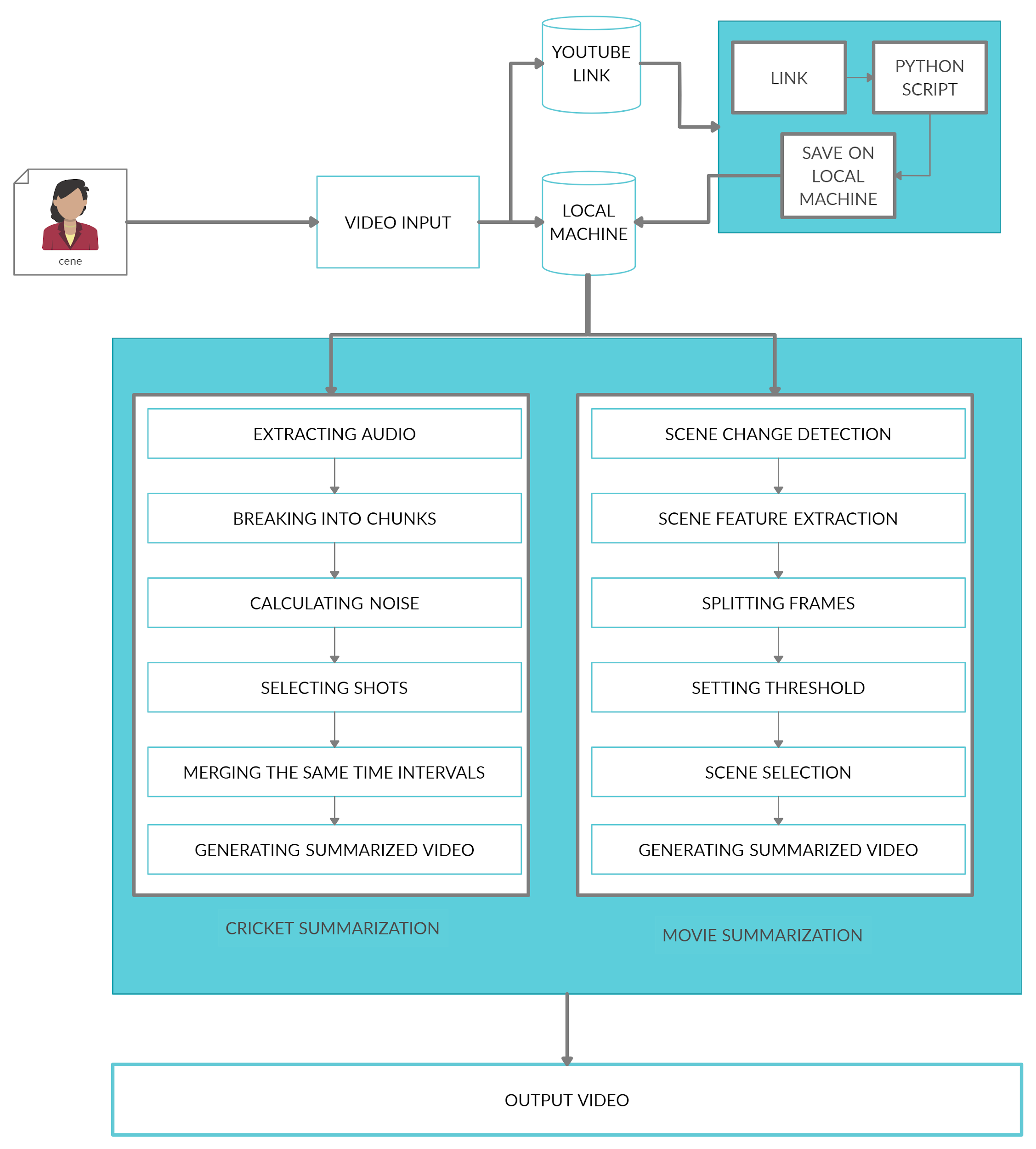
# 

# 

# 

# **3. System Design**

## **3.1. System Architecture**



Our system follows the two-tier architecture. First tier consists of GUI and the second tier consists of linguistic components.

1. GUI: This component in our project deals with the interface for the user where the user can choose the video input from the local storage or can provide a youtube link.
2. Linguistic components: The linguistic component is the block where the actual processing of our project is done. This module is represented for creating summarized video. It has two parts, cricket based summarization and movie based summarization. In the cricket part it generates highlights of that match whereas in the movie part it generates trailer of that movie.

## **3.2. System Modules**

There are 4 modules in this project. They are -

1. User Interface: This module contains a user interface in which the user select the input type of the video and type of summarization (movie/cricket).
2. Input Type: This module contains how the input file will be uploaded. It can be used from a local machine or from a youtube video link which the video can be downloaded by running a python script and saving on the local machine.
3. Summarization Process:
   1. Movie Summarization: This module will generate trailer of that movie video by using feature extraction and key frame algorithm.
   2. Cricket Summarization: This module will generate highlights of that video by using the short time energy.
4. Formation of Output: This module will save the output video on the local machine.

## **3.3. User Requirement**

1. Presentation: The user should be able to select the category for which they want the summary and should be able to copy the URL from YouTube to watch the summary of the video of their choice.
2. Clarity: The user should be able to understand the interface and perform the tasks above easily and completely.
3. Accessibility: User can access the system anytime.

## **3.4. Functional Requirements**

### **3.4.1. General Requirements**

1. The system will be able to extract key frames from the video and present the most informative or interesting materials for potential users from the FlagFile and delete it after its use.
2. The system will allow users different actions according to their choice, that is, Movies,Cricket Highlights or YouTube video.
3. The shortening process should be consistent and transparent to users.

### **3.4.2. Video to Summary Requirements**

1. The system will receive the choice of video from the user and transform it into a Summary.
2. The system will provide options which include Movies,Cricket, Youtube URL and browse from Computer as input.
3. The system will provide segment selection and fast-forwarding.
4. The output summary is usually composed of a set of keyframes or video clips extracted from the original video with some editing process.

### **3.4.3. Presentation requirements**

1. The system will present the output of the video by selecting and presenting the most informative materials without losing any important aspects.

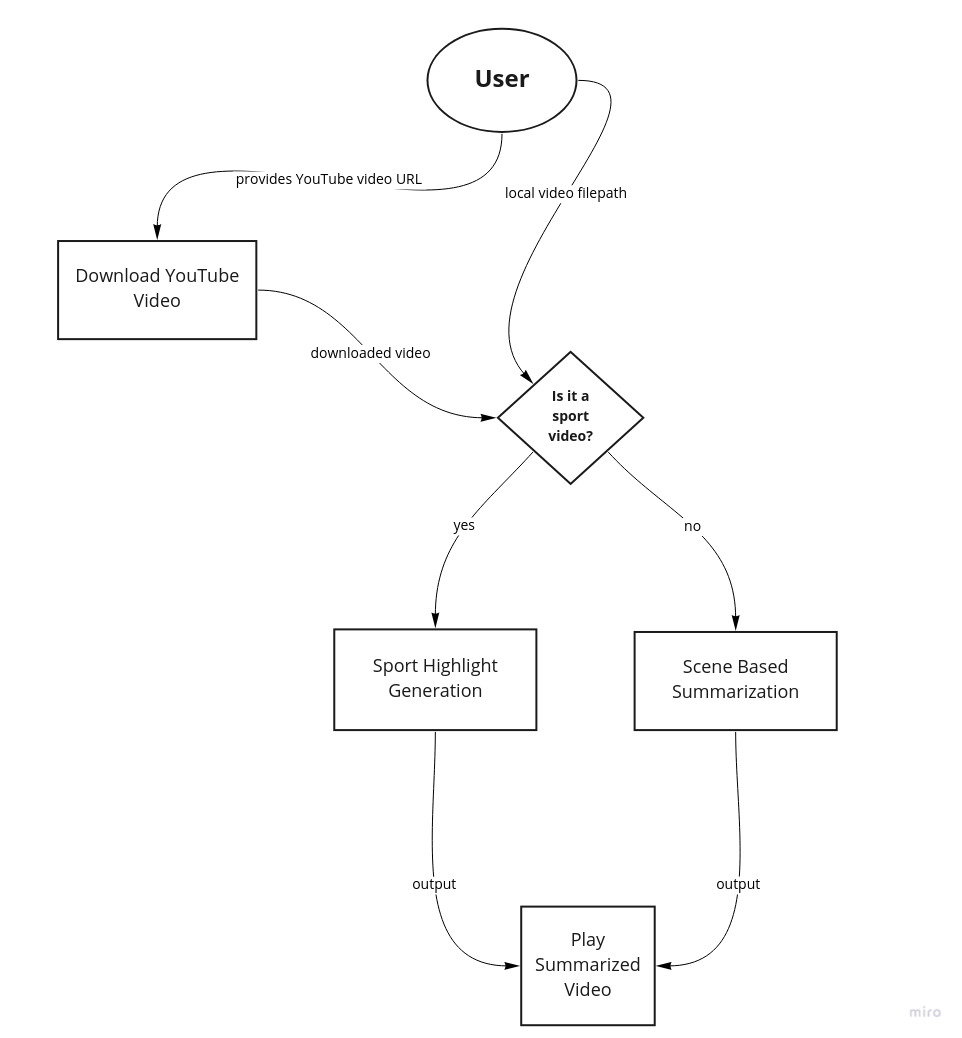
## **3.5. Non-Functional Requirements**

1. Speed: Speed depends upon the length of the video file provided by the user, more the duration, more frames to process.
2. Scalability: The system should provide efficient storage, and retrieval of video summary.
3. Availability: The system should be available 24 hours.

## **3.6. Constraints and Assumptions**

1. Constraints
   1. The speed of the system is dependent on the number of cores of the hardware the system is running on.
2. Assumptions
   1. Users must have Python 3.8 installed on their computer in order for the system to run.

## **3.7. Data Flow Diagram**

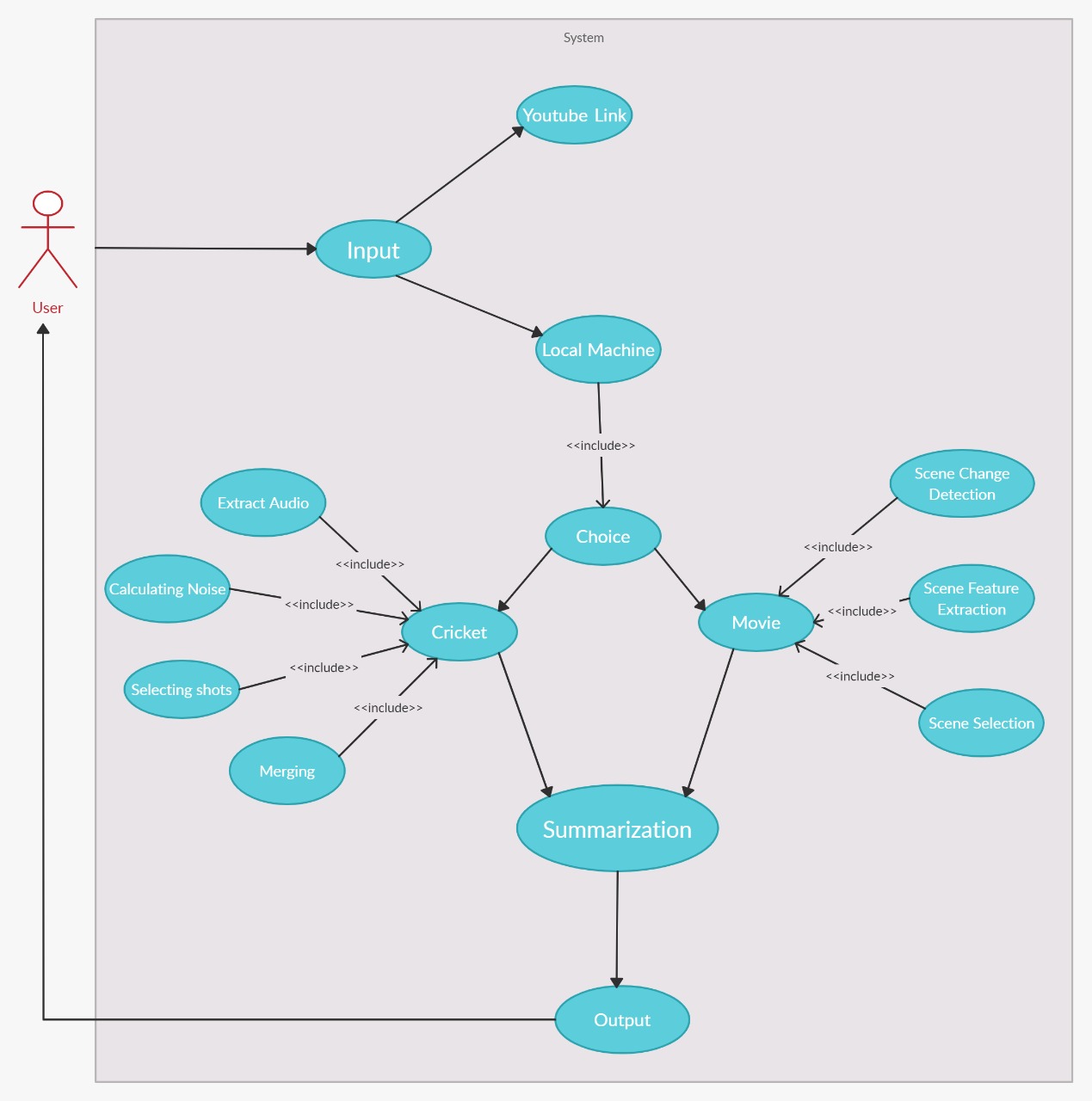


# 

# **4. Detail Design**

## **4.1. UML Diagram**

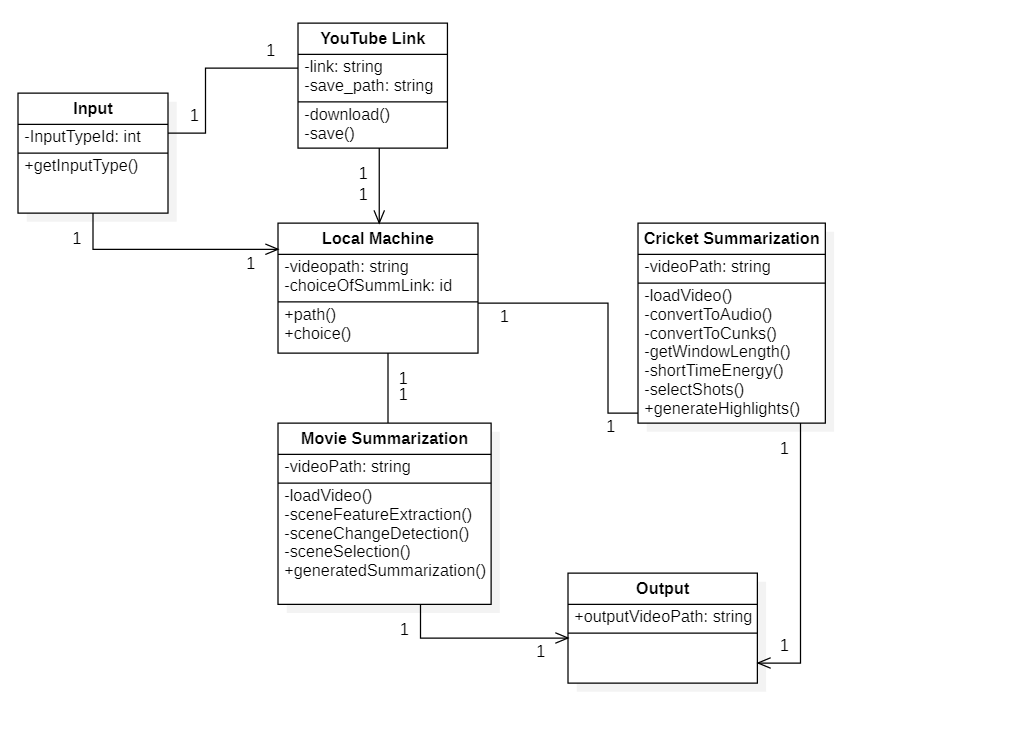
### **4.1.1. Use Case Diagram**



|  |  |
| --- | --- |
| Title | Video Summarization |
| Description | Generate Summary of the video given as an input by the user. |
| Primary Actor | User |
| Preconditions | User should be able to browse from Local machine and should be able to copy Youtube URL. |
| Post conditions | User will provide the video for summarization either from the local machine or Youtube link. |
| Main Success Scenario | * User provide their choice of video for summarization. * For Cricket, the script performs Audio Extraction, Noise Calculation, shots selection, and then merging. * For Movie, the script performs Scene Change Detection, Scene Feature Extraction, and Scene Selection. * Then Summarization of the video takes place and displayed to the user. |
| Frequency of Use | User can retrieve the Summary of video any number of times. |

## 

### **4.1.2. Class Diagram**



#### CRC 1

|  |  |
| --- | --- |
| **Class Name** | Input |
| **Class Type** | USER |
| **Characteristics** | Provide Input management |
| **Superclass** | None |
| **Subclass** | * Youtube Link * Local Machine |
| **Variables** | * InputTypeID |
| **Services** | Provide what should be input type |
| **Responsibilities** | **Collaborators** |
| ● getInputType() | System |

#### CRC 2

|  |  |
| --- | --- |
| **Class Name** | Local Machine |
| **Class Type** | System |
| **Characteristics** | Load the file. |
| **Superclass** | Input |
| **Subclass** | * Cricket Summarization * Movie Summarization |
| **Variables** | * VideoPath |
| **Services** | Choice of the Summarization User wants to do. |
| **Responsibilities** | **Collaborators** |
| * path() * choice() | User |

#### CRC 3

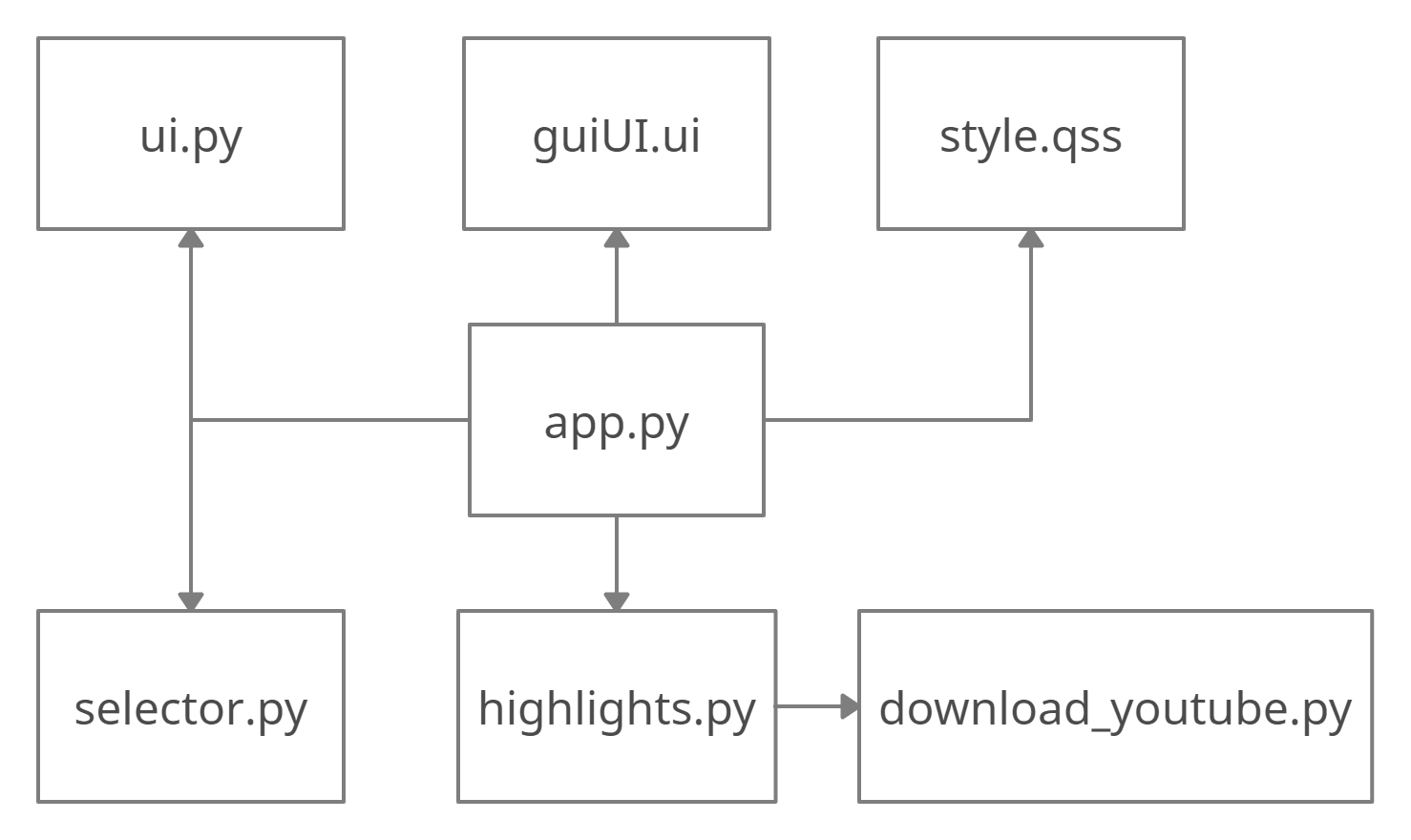
|  |  |
| --- | --- |
| **Class Name** | Cricket Summarization |
| **Class Type** | System |
| **Characteristics** | Load the file. |
| **Superclass** | Local Machine |
| **Subclass** | * Output |
| **Variables** | * VideoPath |
| **Services** | * Extract the Audio. * Calculate the noise. * Break into chunks. * Calculate the short time energy. * Selection of shots. * Generation video summarization. |
| **Responsibilities** | **Collaborators** |
| * loadVideo() * convertToAudio() * convertToChunks() * getWindowLength() * shortTimeEnergy() * selectShots() * generateHighlights() | System |

#### CRC 4

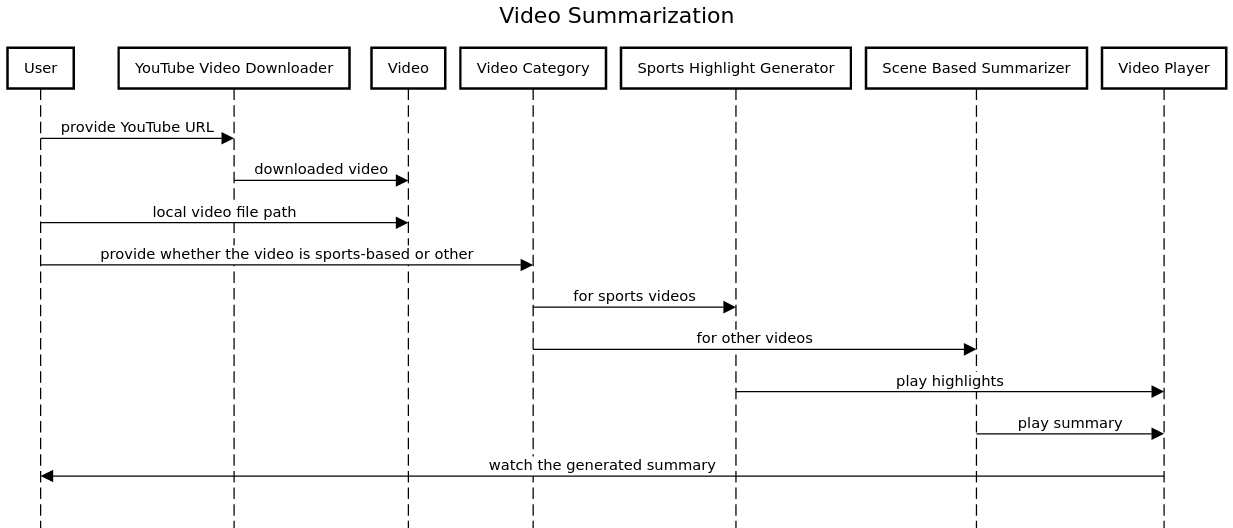
|  |  |
| --- | --- |
| **Class Name** | Movie Summarization |
| **Class Type** | System |
| **Characteristics** | Load the file. |
| **Superclass** | Local Machine |
| **Subclass** | * Output |
| **Variables** | * VideoPath |
| **Services** | * Load the video. * Extract the feature from scenes. * Detect the change of scenes. * Select the scenes. * Generated Video Summarization. |
| **Responsibilities** | **Collaborators** |
| * loadVideo() * sceneFeatureExtraction() * sceneChangeDetection() * sceneSelection() * generateSummarization() | System |

## 

### **4.1.3. Component Diagram**



### **4.1.4. Sequence Diagram**



# 

# **5. Testing**

## **5.1. Test Objective**

The objective of this document is to expand on the test plan and provide specific information needed to actually perform the necessary tests. By providing detailed test information, we hope to reduce the probability of overlooking items and improve test coverage. Testers will be able to use each test case provided in this document to move forward and begin testing. Test results will be logged in a database and a complete bug report generated for each test failure.

## **5.2. Detailed Testing Strategy**

### **5.2.1. Unit Testing**

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the programs correctness.

### **5.2.2. White Box Testing**

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. The test cases that have been generated shall cause each condition to be executed at least once. To ensure this happens, we are applying Basis Path Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply.

### **5.2.3. Black Box Testing**

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We have decided to perform Equivalence Partitioning and Boundary Value Analysis testing on our application. The Equivalent Partitioning will be performed at both the unit test level and the system test level. Boundary Value analysis will only be done at the system test level. In considering the inputs for our equivalence testing, the following types will be used:

* Legal input values – Test values within boundaries of the specification equivalence classes. This shall be input data the program expects and is programmed to transform into usable values.
* Illegal input values – Test equivalence classes outside the boundaries of the specification. This shall be input data the program may be presented, but that will not produce any meaningful output.

The equivalence partitioning technique is a test case selection technique in which the test designer examines the input space defined for the unit under test and seeks to find sets of input that are, or should be, processed identically. Black box testing will be performed by the test team. All procedural steps have been included to assist the team in executing the various tests.

### **5.2.4. Pass/Fail Criteria**

This section will include the master list of both white box and black box tests which will be used to track the progress of the testing. A test will be considered a failure if the expected result or output is not achieved. A bug report will be filled out for each failure and will be submitted to the development team for correction. After the bug has been fixed, the test case will be repeated.

## **5.3. Test Cases**

### **5.3.1.** **Install desktop application**

|  |  |
| --- | --- |
| Name | Install desktop application |
| Summary | The desktop application is installed on device. |
| Users | All users |
| Pre-conditions | The device has python and pip installed. |
| Basic Course of Events | 1. The user accesses the installation file.  2. The mobile application is installed on the device. |
| Input | N/A |
| Expected output | The app is ready to be used. |

### **5.3.2. Desktop Application**

|  |  |
| --- | --- |
| Name | Desktop application |
| Summary | The user accesses the desktop app to do summarization. |
| Users | All users |
| Pre-conditions | The app is installed on the desktop. |
| Basic Course of Events | 1. The user enters the required information shown on the screen and clicks on the generate button.  2. The video file gets generated and saved on a local machine. |
| Input | N/A |
| Expected output | Video file is saved on the local machine. |

### 

### **5.3.3. Youtube Video Download**

|  |  |
| --- | --- |
| Name | Youtube video download |
| Summary | The video is downloaded on the machine. |
| Users | All users |
| Pre-conditions | The device has internet connection. |
| Basic Course of Events | 1. Select proper fields. 2. Give proper URL. |
| Input | https://www.youtube.com/watch?v=QhGXjJSg2jA |
| Expected output | The video is downloaded. |

### **5.3.4. Summarization**

|  |  |
| --- | --- |
| Name | Summarization |
| Summary | Summary of videos. |
| Users | All users |
| Pre-conditions | The app is installed on the desktop. |
| Basic Course of Events | 1. Select the required fields. 2. Enter the proper path. |
| Input | c://video//test.mp4 |
| Expected output | The summary of the video has been generated. |

## **5.4. Test tools**

1. Pyqt5 for desktop application
2. Moviepy for video formatting
3. Pytube for youtube video downloading.

# **6. Planning**

|  |  |  |
| --- | --- | --- |
| Time Period | Activity | Comments |
| Week 1 | Requirement Gathering | * Requirement gathering was to be done through searching on inter- net and taking the ideas, sharing the views among group members. |
| Week 2 | Planning | * Planning was done by reviewing IEEE papers. |
| Week 3 | Design | * Designing UML diagrams. |
| Week 4 | Frontend User Interface | * Designing application from UML diagrams. |
| Week 5 | Frontend Coding | * Coding for frontend in Python |
| Week 6 - 10 | Backend Coding | * Coding for Backend in Python |
| Week 11 | Test Cases | * Creating test cases. |
| Week 12 | Unit and Functional Testing | * Carrying out unit and functional testing. |
| Week 13-17 | Implementation of Additional Features | * Implementing additional features. |
| Week 18 | Debugging | * Debugging the code. |
| Week 19 -20 | Final Testing | * Final Testing. |

# 

# **7. Conclusion**

Video summarization is a very important tool where people can use it to get the main idea and the important scenes without watching the full original video.

The system is designed to help the users to just glance at the content of the video quickly, and decide whether to watch the video or not. For sports, looking at the highlights of the match, users can decide whether the match is interesting or not, many sports fans will be able to benefit from a summarized version of sports video which is available from anywhere.

# 

# **8. References**

[1] ‘**Video summarization using deep semantic features**,’ M. Otani, Y. Nakashima, E. Rahtu, J. Heikkila, and N. Yokoya, in Proc. Asian Conference on Computer Vision, 2016.

[2] **‘Structure Analysis of Sports Video Using Domain Models.’** Zhong, D. and S.-F. Chang.. in IEEE ICME 2001. 2001. Tokyo, Japan.

[3] **‘Event detection and summarization in sports video’.** Li, B. and M. Ibrahim Sezan. in IEEE Workshop on Content-Based Access of Image and Video Libraries, 2001

[4] **Analytics Vidhya.**

[www.analyticsvidhya.com/blog/2019/09/guide-automatic-highlight-generation-python-without-machine-learning/](http://www.analyticsvidhya.com/blog/2019/09/guide-automatic-highlight-generation-python-without-machine-learning/)

[5] **GitHub.** <https://github.com/aravindpai/Cricket-Highlights-Generation>

[6] **‘Scene change detection using multiple histograms for motion-compensated frame rate up-conversion,’.** S.-J. Kang, S. I. Cho, S. Yoo, and Y. H. Kim in J. Display Technol., vol. 8, no. 3, pp. 121–126, Mar. 2012.

[7] **‘Fast scene change detection for personal video recorder.’** J.-R. Kim, S. Suh, and S. Sull,

IEEE Trans. Consum. Electron., vol. 49, no. 3, pp. 683–688, Aug. 2003.

[8] **GitHub.** <https://github.com/Breakthrough/PySceneDetect>