

**University Institute of Information Technology,**

**PMAS-Arid Agriculture University,**

**Rawalpindi Pakistan**

**Video Classification And Indexing**

***By***

**Ahsan Yousaf 17-ARID-1284**

**Hina Shabbir 17-ARID-1324**

**Anum Bibi 17-ARID-1292**

***Supervisor***

**Mr. Zeeshan Javed**

***Bachelor of Science in Computer Science (2017-2021)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software documentation and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Ahsan Yousaf Hina Shabbir Anum Bibi

--------------------------- --------------------------- ---------------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS)“**video classification and indexing** ” was developed by“ **Ahsan Yousaf, Registration 17-ARID-1284”**, “**Hina Shabbir, Registration 17-ARID-1324”**and “**Anum Bibi, Registration 17-ARID-1292”**under the supervision of“ **Mr. Zeeshan Javed** ” and that in their opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Science.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner (If any)**

---------------------------------------

**Administrator UIIT**

**Executive Summary**

In public places, there is often a need for monitoring people and different activities going on, which can be referred later for many reasons including security. Appointing humans for this task involves many problems such as increased employee hiring, accuracy problem, trust, no proof for later use, and also the fact that a human can remember things till a certain time limit. Talking about the current security system, they use dumb still cameras with a continuous recording facility ir-respective of the fact that any event may happen or not. Moreover they are usually pointing at a specific user defined locations so more than one cameras are required to cover the entire region.

To prevent all these problems from prevailing, the CSCS is developed. It is a surveillance system, which provides solution to many of these problems. It is a stand-alone application which doesn’t require any computer to operate. It monitors different situations using a camera which is able to rotate intelligently based on sensor messages and captures the scene in the form of video or photos later reference as well.

**C**ustomizable **S**urveillance **C**ontrol **S**ystem**(CSCS)** is a surveillance system that can be assigned a sensor type as in our case a heat sensor is used, it works accordingly, rotates the camera upon event detection and perform user defined actions like capturing video and stores them, for the future use.

It is an embedded system consisting of Linux fox kit with embedded a running server application also a camera, USB storage device and a sensor node base station is attached with fox kit. LAN communication is used by user to download the videos and to operate the system manually.

**Acknowledgement**

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “Mr. Zeeshan Javed” for personal supervision, advice, valuable guidance and completion of this project. We are deeply indebted to him for encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Ahsan Yousaf Hina Shabbir Anum Bibi

--------------------------- --------------------------- ---------------------------

**Table of Contents**

[Chapter 1: Introduction 1](#_Toc77482127)

[1.1 Brief 1](#_Toc77482128)

[1.2 Relevance to Course Modules 1](#_Toc77482129)

[1.3 Project Background 1](#_Toc77482130)

[1.4 Literature Review 1](#_Toc77482131)

[1.5 Analysis from Literature Review 3](#_Toc77482132)

[1.6 Methodology and Software Lifecycle for This Project 3](#_Toc77482133)

[1.6.1 Rationale behind Selected Methodology 3](#_Toc77482134)

[Chapter 2: Problem Definition 4](#_Toc77482135)

[2.1 Problem Statement 4](#_Toc77482136)

[2.2 Purpose 4](#_Toc77482137)

[2.3 Product Function 4](#_Toc77482138)

[2.4 Purposed Architecture 4](#_Toc77482139)

[2.5 Project Deliverables 5](#_Toc77482140)

[2.6 Operating Environment 6](#_Toc77482141)

[2.6 Assumptions And Dependencies 6](#_Toc77482142)

[Chapter 3: Requirement Analysis 7](#_Toc77482143)

[3.1 Use Cases 7](#_Toc77482144)

[3.2 Functional Requirements 16](#_Toc77482145)

[3.3 Non-Functional Requirements 20](#_Toc77482146)

[3.3.1 Usability 20](#_Toc77482147)

[3.3.2 Reliability 20](#_Toc77482148)

[3.3.3 Performance 20](#_Toc77482149)

[Chapter 4: Design and Architecture 22](#_Toc77482150)

[4.1 UML Structural Diagrams 22](#_Toc77482151)

[4.1.1 Component Diagram 22](#_Toc77482152)

[4.1.2 Deployment Diagram 23](#_Toc77482153)

[4.1.3 Class Diagram: 24](#_Toc77482154)

[4.2 UML Behavioral Diagrams 25](#_Toc77482155)

[4.2.1 Activity Diagrams 25](#_Toc77482156)

[4.2.2 State Machine Diagrams 30](#_Toc77482157)

[4.3 UML Interaction Diagrams 31](#_Toc77482158)

[4.3.1 Sequence Diagrams 31](#_Toc77482159)

[Chapter 5: Implementation 36](#_Toc77482160)

[5.1 Language Python 3.7 36](#_Toc77482161)

[5.2 Libraries 36](#_Toc77482162)

[5.3 Tool 37](#_Toc77482163)

[IDE (PyCharm 2020.2.3) 37](#_Toc77482164)

[5.4 User Interface 37](#_Toc77482165)

[Chapter 6: Testing and Evaluation 41](#_Toc77482166)

[6.1 Module / Unit Testing 41](#_Toc77482167)

[6.1.1 Test Case Name: User Signup 41](#_Toc77482168)

[6.1.2 Test Case Name: User Signup 42](#_Toc77482169)

[6.1.3 Test Case Name: User Signup 43](#_Toc77482170)

[6.1.4 Test Case Name: User Login 44](#_Toc77482171)

[6.1.5 Test Case Name: User Login 45](#_Toc77482172)

[6.1.6 Test Case Name: User Logout 46](#_Toc77482173)

[6.1.7 Test Case Name: Upload Video 47](#_Toc77482174)

[6.1.8 Test Case Name: Upload Video 48](#_Toc77482175)

[6.1.9 Test Case Name: Video Splitting 49](#_Toc77482176)

[6.1.10 Test Case Name: Video Splitting 50](#_Toc77482177)

[6.1.11 Test Case Name: Search Video 51](#_Toc77482178)

[6.1.12 Test Case Name: Search Video 52](#_Toc77482179)

[6.1.13 Test Case Name: Video Classification 53](#_Toc77482180)

[6.1.14 Test Case Name: Video Classification 54](#_Toc77482181)

[6.1.15 Test Case Name: Audio Classification 55](#_Toc77482182)

[6.1.16 Test Case Name: Audio Classification 56](#_Toc77482183)

[6.2 Integration Testing 57](#_Toc77482184)

[Chapter 7: Conclusion and Future Work 58](#_Toc77482185)

[7.1 Conclusion 58](#_Toc77482186)

[7.2 Future Work 58](#_Toc77482187)

[References 59](#_Toc77482188)

**List of Figures**

[Figure 2. 1 5](#_Toc77630302)

[Figure 3. 1 21](#_Toc77630303)

[Figure 4. 1 22](#_Toc77630304)

[Figure 4. 2 23](#_Toc77630305)

[Figure 4. 3 24](#_Toc77630306)

[Figure 4. 4 25](#_Toc77630307)

[Figure 4. 5 26](#_Toc77630308)

[Figure 4. 6 27](#_Toc77630309)

[Figure 4. 7 28](#_Toc77630310)

[Figure 4. 8 29](#_Toc77630311)

[Figure 4. 9 30](#_Toc77630312)

[Figure 4. 10 30](#_Toc77630313)

[Figure 4. 11 30](#_Toc77630314)

[Figure 4. 12 31](#_Toc77630315)

[Figure 4. 13 32](#_Toc77630316)

[Figure 4. 14 33](#_Toc77630317)

[Figure 4. 15 34](#_Toc77630318)

[Figure 4. 16 35](#_Toc77630319)

[Table 6. 1 42](#_Toc77630320)

[Table 6. 2 43](#_Toc77630321)

[Table 6. 3 43](#_Toc77630322)

[Table 6. 4 44](#_Toc77630323)

[Table 6. 5 45](#_Toc77630324)

[Table 6. 6 46](#_Toc77630325)

[Table 6. 7 47](#_Toc77630326)

[Table 6. 8 48](#_Toc77630327)

[Table 6. 9 49](#_Toc77630328)

[Table 6. 10 50](#_Toc77630329)

[Table 6. 11 51](#_Toc77630330)

[Table 6. 12 52](#_Toc77630331)

[Table 6. 13 53](#_Toc77630332)

[Table 6. 14 54](#_Toc77630333)

[Table 6. 15 55](#_Toc77630334)

[Table 6. 16 56](#_Toc77630335)

# 

# Chapter 1: Introduction

This chapter is all about introduction of our project which include a brief introduction of project, literature background of our project and methodology we will use in our system.

## 1.1 Brief

We are creating a platform that will implement video classification and video indexing system. User would be able to upload a video of maximum 1 minutes duration. System allows user to upload and search a video. Proposed system will provide more accurate result by implementing new technologies and algorithms of image processing to perform video classification. Outcomes of the system are video classification and indexing. OpenCV, TensorFlow, PyCharm, Python are the tools, and agile methodology we will use in the development of our system.

## 1.2 Relevance to Course Modules

Following Courses are related to our project that we learnt in our degree.

* Software Engineering-I and Software Engineering-II help us to design UML diagrams and Use case model for our system.
* Database systems and Distributed Database system help us in designing and maintaining database for our system.

## 1.3 Project Background

Approaches to classify video are: using text modality, using audio, video and combination of these. The majority of the literature describes approaches that utilize features from a single modality. [1]

Our main idea is to develop a desktop application with the implementation of video classification and indexing. We will classify videos using two modality of the video one is audio and second is visual. Video Classification using two modality will give us better result rather than single modality.

## 1.4 Literature Review

**video classification**

According to a review paper of video classification, to classify a video, feature are drawn from three modalities: text, audio and visual. Approaches to classify video are: using text modality, using audio, video and combination of these. The majority of the literature describes approaches that utilize features from a single modality. [1]

**Machine learning techniques used for video classification**

Proposed work for video classification of a review paper is “The methods reviewed so far in this paper uses unsupervised learning to develop video classification methods. This work uses the strength of unsupervised learning method and fuzziness to attain better classifications. In this proposed work the video data is first divided into key clips and then we extract the visual features like color, texture and motion vectors. These features are then subjected to supervised deep learning models and the then results then undergo Fuzzy C means clustering. With the objective of further improving the results we apply an optimization algorithm which then categorizes the video under a label. The proposed system is shown in figure.” [2]

Video Data

Key Clips

Visual Features

(Color, texture,

Motion vectors)

Categories

Supervised deep learning model

Optimizing algorithm

Fuzzy C-means Clustering

**Video indexing technology**

**“**To make searching video easier, FX Palo Alto Laboratory, Inc., located in Silicon Valley, has developed a video indexing technology that is able to find specific scenes in videos by the text (character strings) appearing in them. With this technology, by simply entering a search term, users can search for lecture videos and scenes in which that term appears.”[3].

**Audio Emotion Classification**

“This project presents a deep learning classifier able to predict the emotions of a human speaker encoded in an audio file. The classifier is trained using 2 different datasets, RAVDESS and TESS, and has an overall F1 score of 80% on 8 classes (neutral, calm, happy, sad, angry, fearful, disgust and surprised)”.[4]

## 1.5 Analysis from Literature Review

“Approaches to classify video are: using text modality, using audio, video and combination of these. The majority of the literature describes approaches that utilize features from a single modality. [1]”

Our proposed system will work with video. It will take a video as an input and classify the video one the following category, Basketball, Formula1, Swimming, Cricket, Boxing, Kabaddi, Table tennis, weight lifting. Our overall accuracy rate 95%. We download datasets from GitHub. [6]

Audio emotion classifier classify the audio of a human into following category, neutral, calm, happy, sad, angry, fearful, disgust and surprised using 2 different dataset with overall score of 80%. [4]

Our proposed system will classify the audio of the video into two different category which is speech and music. We use two datasets one for speech and second for music. And our overall score is 98%. We download dataset from Kaggle. [5]

## 1.6 Methodology and Software Lifecycle for This Project

Video Classification and indexing system will provide more accurate result by implementing new technologies and algorithms of image processing to perform video classification. For implementation of this system we will use agile methodology. The agile approach is focused on giving high priority from the very beginning of the development cycle to customer involvement. The aim to keep the customer engaged at very stage so that at the end they have a product for which they are satisfied.

### 1.6.1 Rationale behind Selected Methodology

We use Agile methodology because Agile helps teams to deliver a prototype and improve it in each cycle and this strategy saves the customer money and time because at each point of production the customer checks and approves the product. If there are problems then modification can be made during production cycles to fix the issue.

# Chapter 2: Problem Definition

Problem statement of our project and the solution to that problem is discuss in this chapter.

Our project deliverables, project environment and assumption and dependencies of our project is also part of this chapter.

## 2.1 Problem Statement

Number of platform are there in the market having video content but they are lacking of video classification and indexing process which leads towards massive amount of content on the platform without any classification which cause irrelevant suggestions to users.

## 2.2 Purpose

Purpose of our system is to classify videos into various categories and perform indexing on the videos so that it is easier and faster to search the videos category wise due to classification and indexing performed on the videos.

## 2.3 Product Function

Our product function will be to perform video classification on the uploaded video and it will place it in a specific category to which it belongs and it will further perform indexing on the videos so they are easy and quick to search.

## 2.4 Purposed Architecture

Our proposed system will work with video classification and indexing. Video classification and indexing process would be done, when a user tried to upload any video.

Video classification help system to detect the type of video content and make decision, what type of content is in the video. And indexing would help to organize all data so that when user try to find any type of video, system will just fetch video that fall into that search tag.

The diagram 2.1 shows flow of control in the system

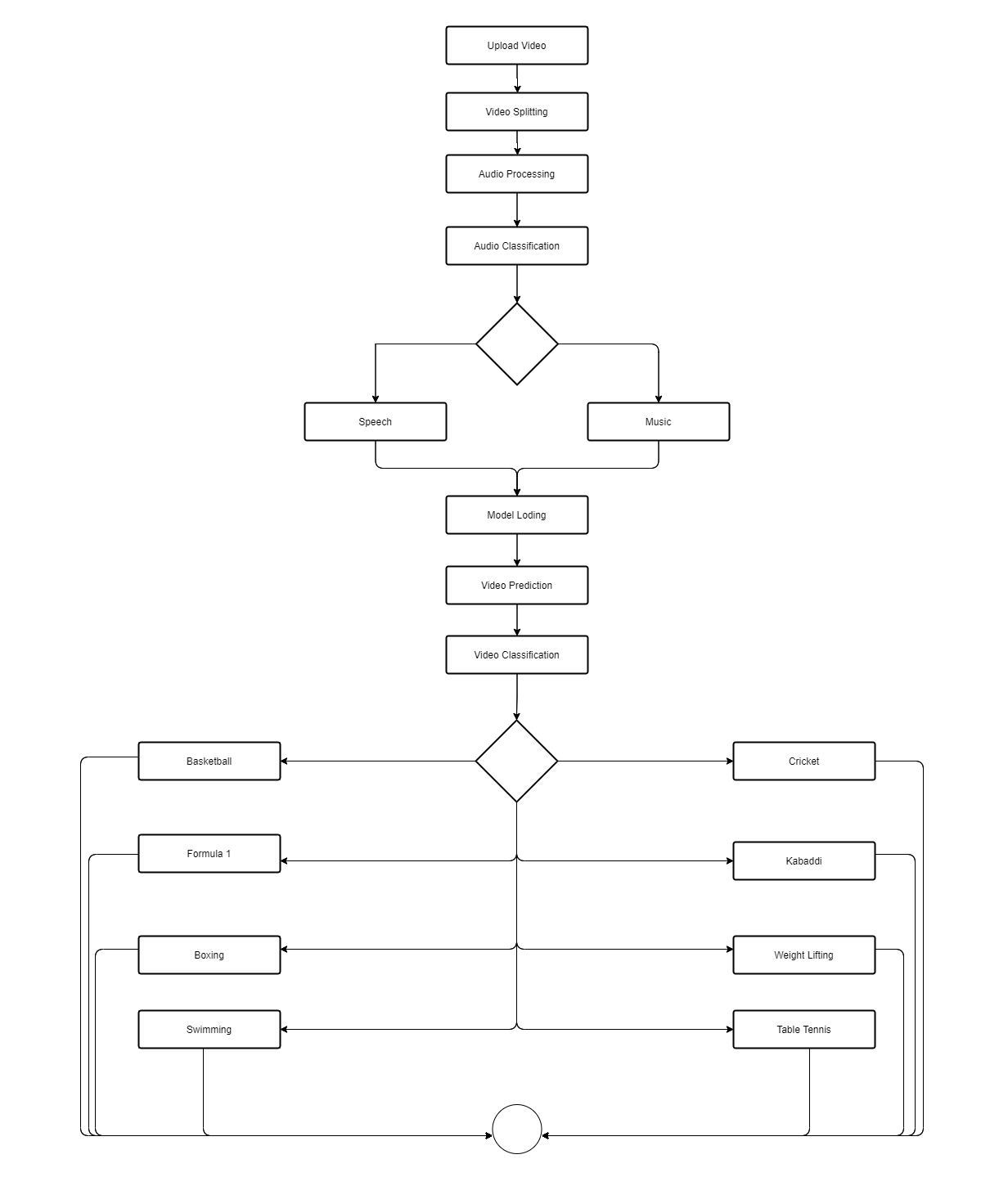


Figure 2.

## 2.5 Project Deliverables

First deliverable of the project will be a brief proposal of the project which will highlight all the major requirements and all the important information about the system. A system with the implementation of video classification and indexing is final deliverable of our project. Complete system development report or documentation will also be our deliverable.

## 2.6 Operating Environment

We are building our system in python so it will be cross platform and will run on the latest versions of Operating Systems and machines.

But as we are developing it on window 10 so our system will work on windows with python 3.7 installed on it having at least 4GRAM and I5 processor

## 2.6 Assumptions And Dependencies

Our assumption is that our system should be cross platform and there should be no dependency of the machine that it will run on so we are making our system in python so it can work on all type of operating systems and machines with suitable hardware and software.

# Chapter 3: Requirement Analysis

All the requirements related to our project and use cases of this project are discussed in this chapter.

## 3.1 Use Cases

Use cases are a widely used and highly regarded format for capturing requirements. Before writing functional requirement use cases can help you to understand the requirements in the way user expect. Following table presents you not only the template to write use case(s) as well as guides you to write each section with example.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-1 |
| **Use Case Name:** | User Signup |
| **Actors:** | User |
| **Description:** | User create an account and become a registered user. User enter their personal details. |
| **Trigger:** | New user want to create an account. |
| **Preconditions:** | Username should be unique. |
| **Postconditions:** | User entered data is stored in user account and user get username and password to login to the system. |
| **Normal Flow:** | When the user accesses the system feature to create an account, this use case starts   1. The user enters the required values for the User Account  * Name * Gender * User Name * Password * Confirm Password   And then will press the Submit button.   1. The system validates the entered User Account information. 2. The User Account information values are stored in the User Account. The system notifies the user that the account has been created. |
| **Alternative Flows:** | 1a: User entered invalid user account information, the following occurs:   1. The system warn the user name is already exist in the system. 2. The system warn the user password and confirm password should be same. 3. The user reenters the data and it is re-validated by the system. 4. The User Account Information is processed if valid information is entered. |
| **Exceptions:** |  |
| **Includes:** |  |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-2 |
| **Use Case Name:** | User Login |
| **Actors:** | User |
| **Description:** | User enter their user name and password for login. |
| **Trigger:** | User want to login the system. |
| **Preconditions:** | Should have an account and user name and password should correct. |
| **Postconditions:** | The User is authenticated and user get complete access of the System. |
| **Normal Flow:** | The use case starts when the user accesses the system's Log in feature.   1. User enter user username and password. 2. The system validates the entered information, after pressing login button, making sure that the entered email and password are valid. 3. The User is login to the system. |
| **Alternative Flows:** | 1: If User entered invalid user name and password system display the alert when user press login button. |
| **Exceptions:** |  |
| **Includes:** | User have an account. |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-3 |
| **Use Case Name:** | User Logout |
| **Actors:** | User |
| **Description:** | This use case allow the user to logout the system. |
| **Trigger:** | User want to logout from the system. |
| **Preconditions:** | Login to the system. |
| **Postconditions:** | Divert to login screen. |
| **Normal Flow:** | 1: Press logout button.  2: System returns the user to the login screen. |
| **Alternative Flows:** |  |
| **Exceptions:** | The user is not Login. |
| **Includes:** | User must be login. |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-4 |
| **Use Case Name:** | Upload video |
| **Actors:** | User |
| **Description:** | User upload the video using upload video Button. |
| **Trigger:** | User want to upload the video. |
| **Preconditions:** | User should be login and have video. |
| **Postconditions:** | Video should be uploaded and displayed in the system. |
| **Normal Flow:** | 1: Press “Upload Video” button.  2: Choose video from computer to upload to the system.  3: Press “Open” button  4: System will upload the video. |
| **Alternative Flows:** |  |
| **Exceptions:** | 1: Video file format is miss match.  2: Video size is too large to upload. |
| **Includes:** |  |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-5 |
| **Use Case Name:** | Search Video |
| **Actors:** | User |
| **Description:** | User can search any type of video using search bar. |
| **Trigger:** | User wants to search video. |
| **Preconditions:** | User must be Login and must have title or keywords that relate to the video. |
| **Postconditions:** | List of video should be displayed. |
| **Normal Flow:** | 1: Go to search bar.  2: Type title of the video or keyword related to the video.  3: Press Enter button or search icon next to search bar.  4: The system will display the related results of video. |
| **Alternative Flows:** | 2a: If Title of the video or keyword not match then system give a message, video not found. |
| **Exceptions:** |  |
| **Includes:** |  |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-6 |
| **Use Case Name:** | Video splitting |
| **Actors:** | User |
| **Description:** | Using an OpenCV library here system will perform video splitting video will split into audio and visuals. |
| **Trigger:** |  |
| **Preconditions:** | Video should be type of MP4.  Video should be of maximum 1 mint. |
| **Postconditions:** | Video will split into audio and visual and save in different folders. |
| **Normal Flow:** | 1: Take the video that user upload to the system.  2: System perform splitting using OpenCV library of python. |
| **Alternative Flows:** |  |
| **Exceptions:** | 1: Video file format is miss match. |
| **Includes:** | User have to upload a video. |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-7 |
| **Use Case Name:** | Video Classification |
| **Actors:** | User |
| **Description:** | System start loading the model and will predict the video category then in will classify the type of video. Type of video could be one of the following. Basketball, Formula1, Swimming, Cricket, Boxing, Kabaddi, Table tennis, weight lifting. |
| **Trigger:** |  |
| **Preconditions:** | Type of the video should be mp4. |
| **Postconditions:** | Video will be classify and its type should be define in system. |
| **Normal Flow:** | 1: Get the video.  2: predict video type.  3: Classify the video. |
| **Alternative Flows:** |  |
| **Exceptions:** | If uploaded video is not from any classified classes system prediction will be wrong. |
| **Includes:** | User have to upload a video and video should be in the format of mp4 then system load model and start prediction. |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

|  |  |
| --- | --- |
| **Use Case ID:** | UC-8 |
| **Use Case Name:** | Audio Classification |
| **Actors:** | System |
| **Description:** | When system perform splitting, later audio will be process, using audio classification model system will predict the audio type. It could be a speech type or music. |
| **Trigger:** | User want to upload a video. |
| **Preconditions:** | Audio should be of type wav.. |
| **Postconditions:** | Classification of audio either speech or music, audio will save in separate folder. And will show a message with classified class. |
| **Normal Flow:** | 1: Get audio that split from video before.  2: Using Kapre and models, analysis the audio and predict the type of audio. |
| **Alternative Flows:** |  |
| **Exceptions:** | If audio is not a type of wav classification will not perform. |
| **Includes:** | User have to upload a video and video should be in the format of mp4. |
| **Special Requirements:** |  |
| **Assumptions:** |  |
| **Notes and Issues:** |  |

## 3.2 Functional Requirements

**FOR USER MODULE:**

**1**

**NAME:** User Signup

**DESCRIPTION**: User enter their personal details.

**INPUT:** User enter Name, Gender , User Name, Password and Confirm password.

**OUTPUT:** Confirmation dialogue box to display message “Successfully Signup” or “Failed Signup”.

**SOURCE:** Mouse and Keyboard.

**DESTINATION:** on screen.

**PRECONDITION:** Username should be unique.

**POSTCONDITION:** User get username and password to login to the system.

**ACTION:**

Step 1: Enter all required details.

Step 2: press “sign up” button.

**2**

**NAME:**  User Login.

**DESCRIPTION**: User enter their username and password for login.

**INPUT:** User enter username and password.

**OUTPUT:** Show dashboard to the user.

**SOURCE:** Mouse and Keyboard.

**DESTINATION:** on screen.

**PRECONDITION:** Should have an account and username and password should correct.

**POSTCONDITION:** User get complete access of the System.

**ACTION:**

Step 1: Enter username and password.

Step 2: press “login” button

**3**

**NAME:**  User Logout

**DESCRIPTION**: user is login to the system and when it press logout button, system will terminate its access to the system.

**INPUT:** user click on logout button.

**OUTPUT:** Display the Login interface.

**SOURCE:** Mouse

**DESTINATION:** Computer screen

**PRECONDITION:** user must be login

**POSTCONDITION:** User must be logout.

**ACTION:**

Step 1: press “ Logout ” button

**4**

**NAME:** Upload video

**DESCRIPTION**: User will upload the video using a button “upload video”.

**INPUT:** Video

**OUTPUT:** Video should be uploaded in the system.

**SOURCE:** Mouse

**DESTINATION:** Computer Screen

**PRECONDITION:**  User should be login and have video.

**POSTCONDITION:** Video must be loaded.

**ACTION:**

Step 1: Press “Upload Video” button.

Step 2: Choose video from computer to upload to the system.

Step 3: Press “Open” button

**5**

**NAME:** Search video

**DESCRIPTION**: User can search any type of video using search bar.

**INPUT:** title or keywords that relate to video

**OUTPUT:** view the list of video.

**SOURCE:** mouse and keyboard

**DESTINATION:** Computer Screen

**PRECONDITION:**  User must be Login and must have title or keywords that relate to the video

**POSTCONDITION:** List of video should be displayed

**ACTION:**

Step 1: Go to search bar.

Step 2: Type title of the video or keyword related to the video.

Step 2: Press Enter button or search icon next to search bar.

**FOR SYSTEM MODULE:**

**6**

**NAME:** Video splitting

**DESCRIPTION**: Using an OpenCV library here system will perform video splitting Video will split into audio and visuals.

**INPUT:** Video

**OUTPUT:** Audio and video.

**SOURCE:** OpenCV and system

**DESTINATION:** System

**PRECONDITION:**  video should be type of MP4.

**POSTCONDITION:** Video will split into audio and video.

**ACTION:**

Step 1: Take the video that user upload to the system.

Step 2: Perform splitting using OpenCV library of python.

**7**

**NAME:**  Video Classification

**DESCRIPTION**: System start loading the model and will predict the video category then in will classify the type of video. Type of video could be one of the following. Basketball, Formula1, Swimming, Cricket, Boxing, Kabaddi, Table tennis, weight lifting.

**INPUT:** video

**OUTPUT:** Classification would be done and system return the type of video one of the following, Basketball, Formula1, Swimming, Cricket, Boxing, Kabaddi, Table tennis, weight lifting.

**SOURCE:** OpenCV, Keras, resnet50 model and system.

**DESTINATION:** System

**PRECONDITION:**  video of type mp4

**POSTCONDITION:** Video will be classify and its type should be return through message box.

**ACTION:**

Step 1: Get the video.

Step 2: load model.

Step 3: predict the video

Step 4: classify the video

**8**

**NAME:** Audio classification

**DESCRIPTION**: When system perform splitting, later audio will be process, using audio classification model system will predict the audio type.

**INPUT:** Audio

**OUTPUT:** Classification of audio.

**SOURCE:** Kapre, Tensorflow, Conv1d, Conv2d, LSTM model and System

**DESTINATION:** System

**PRECONDITION:**  Audio should be of type wav.

**POSTCONDITION:** Classification of audio, audio will save in separate folder

**ACTION:**

Step 1: Get audio that split from video before.

Step 2: Using Kapre and models analysis the audio and predict the type of audio.

## 3.3 Non-Functional Requirements

### 3.3.1 Usability

* The interface of our system is user friendly and easy to get familiar with
* There is no need to learn anything new to use this system.
* There is no hard and fast rule or regulation that need to memorize to use the system.

### 3.3.2 Reliability

* The system is available 24/7.
* The system never crash are hanged. Other than as a result of an operating error.
* The system is able to recover from hardware and power failure and other natural disaster.

### 3.3.3 Performance

* The overall system should be fast and active responsive.
* The system should be able to handle large amount of data comfortably.
* Searching result would be more accurate due to indexing.
* Response time of the system is between 100ms-200ms.
* A one minute video may take 2-3 minute for uploading and classification.
* Over project is specifically designed to classify video and indexing and it will provide more accurate result.

**Use Case Diagram:**

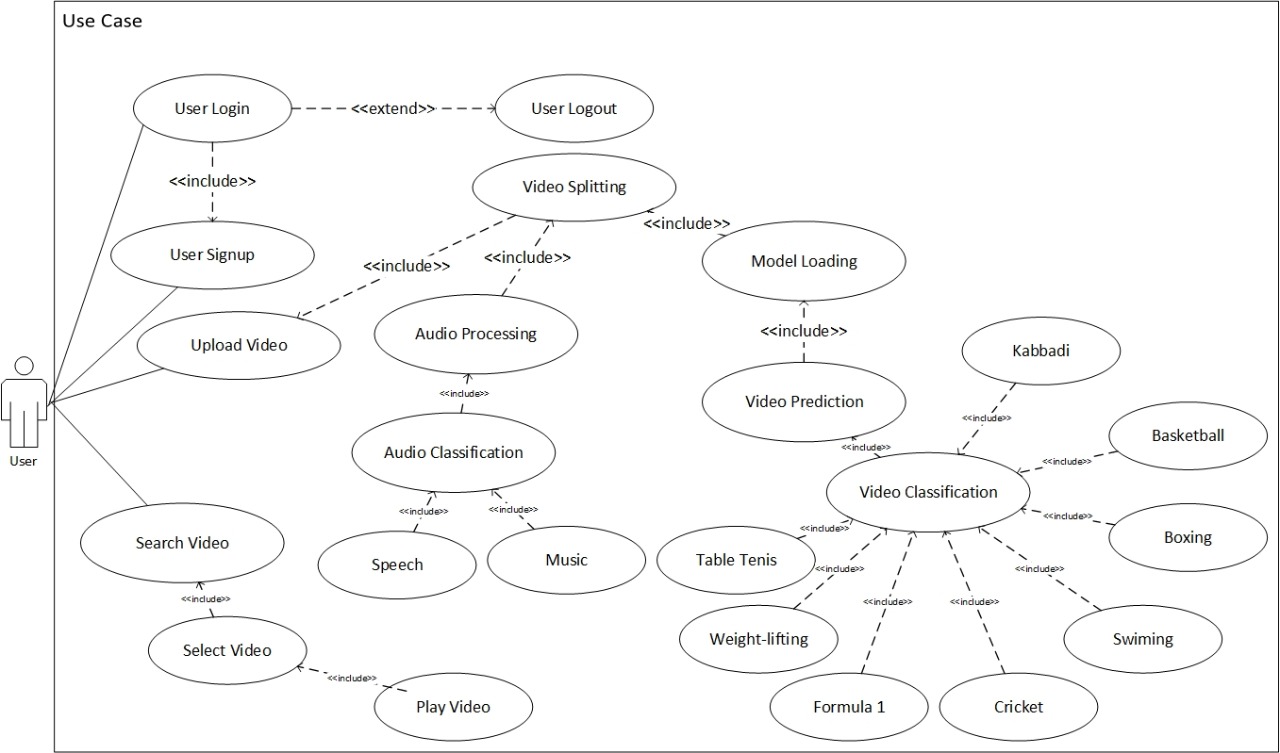


Figure 3.

# Chapter 4: Design and Architecture

This chapter will discuss the design and architecture of our system.

## [4.1 UML Structural Diagrams](#_Toc268523830)

### [4.1.1 Component Diagram](#_Toc268523787)

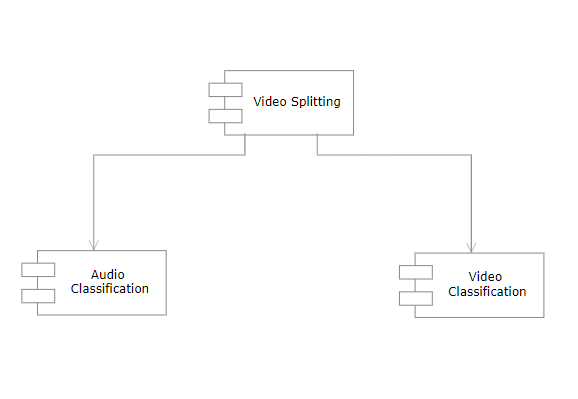


Figure 4.

### [4.1.2 Deployment Diagram](#_Toc268523787)

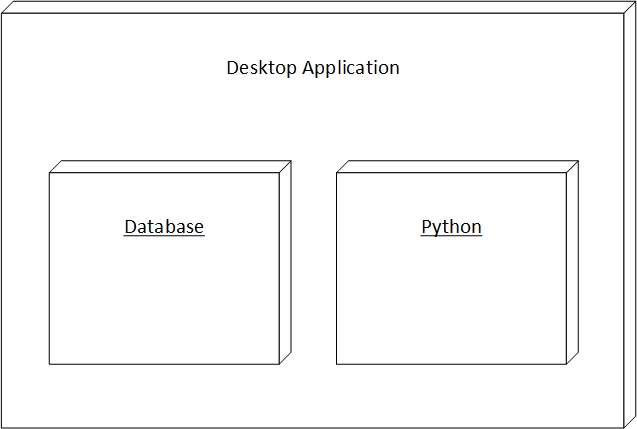


Figure 4.

### 4.1.3 Class Diagram:

Diagram

Description automatically generated

Figure 4.

## [4.2 UML Behavioral Diagrams](#_Toc268523830)

### [4.2.1 Activity Diagram](#_Toc268523787)s

#### [4.2.1.1. Signup](#_Toc268523787)

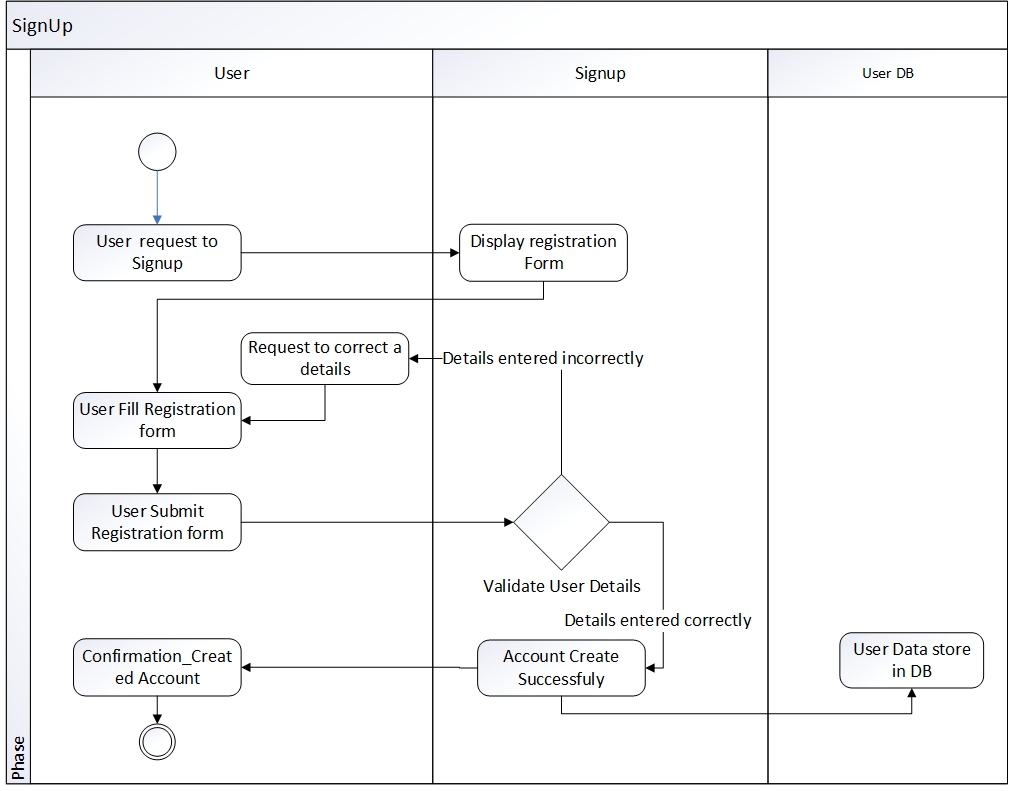


Figure 4.

#### [4.2.1.2.](#_Toc268523787) Login

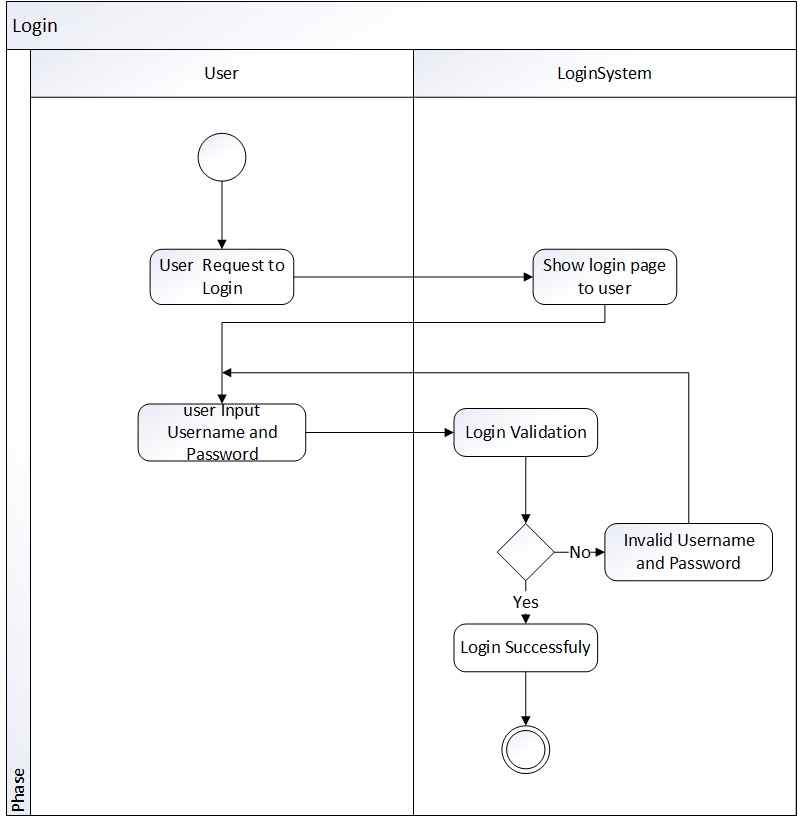


Figure 4.

#### [4.2.1.3. Logout](#_Toc268523787)

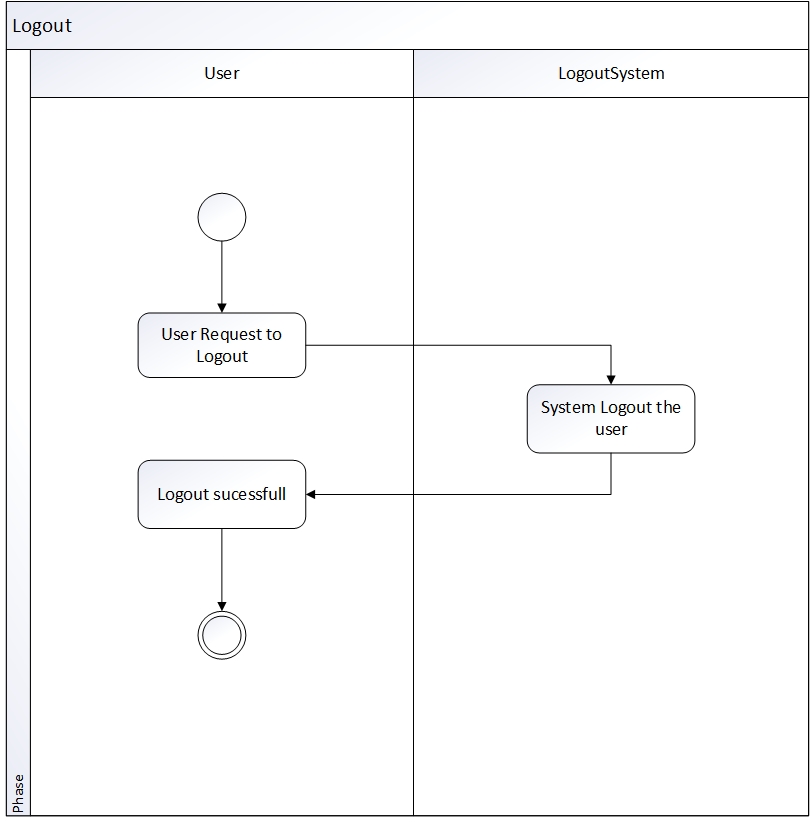


Figure 4.

#### [4.2.1.4.](#_Toc268523787) Search Video

Diagram

Description automatically generated

Figure 4.

#### [4.2.1.5.](#_Toc268523787) Upload Video

Diagram

Description automatically generated

Figure 4.

### [4.2.2 State Machine Diagrams](#_Toc268523787)

#### [4.2.2.1.](#_Toc268523787) Account



Figure 4.

#### [4.2.2.2.](#_Toc268523787) Video



Figure 4.

#### [4.2.2.3.](#_Toc268523787) Audio



Figure 4. 11

## [4.3 UML Interaction Diagrams](#_Toc268523830)

### [4.3.1 Sequence Diagrams](#_Toc268523787)

#### [4.3.1.1. Signup](#_Toc268523787)

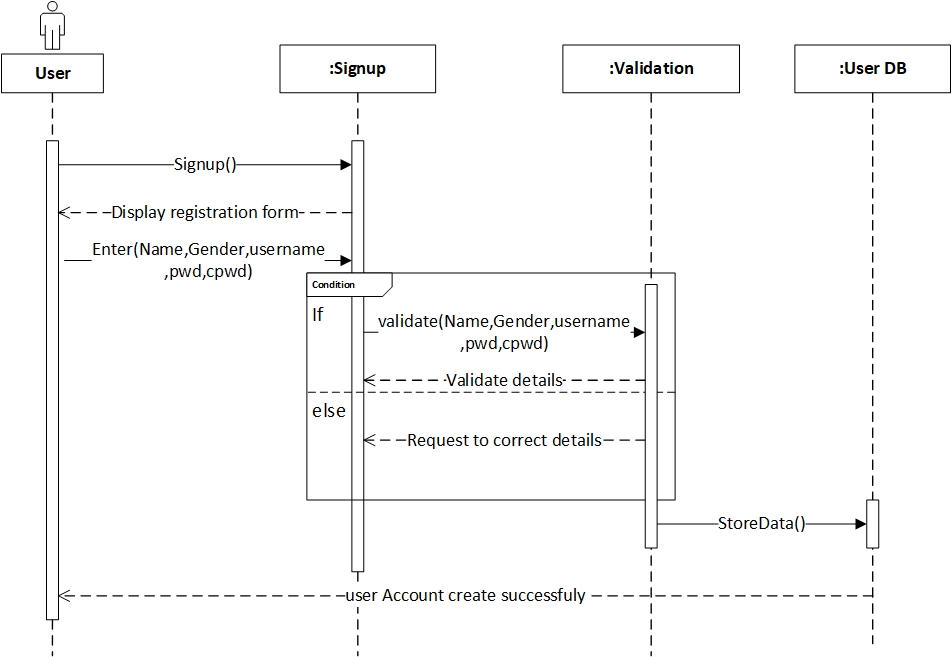


Figure 4. 12

#### [4.3.1.2. Login](#_Toc268523787)

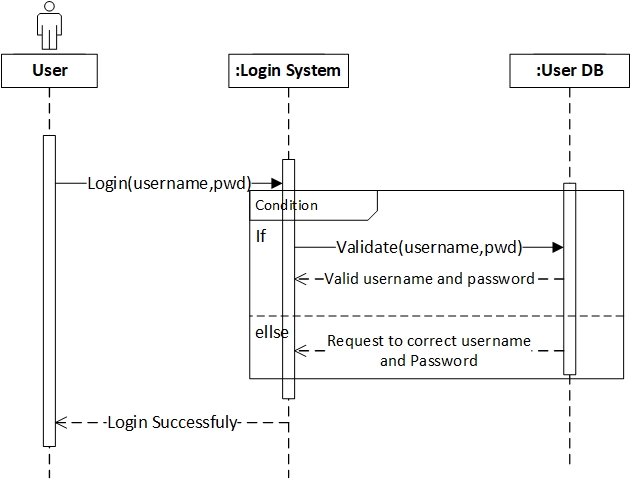


Figure 4. 13

#### [4.3.1.3. Logout](#_Toc268523787)

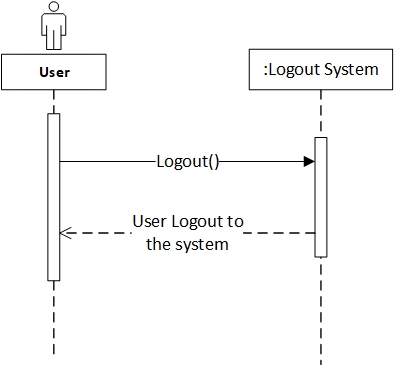


Figure 4. 14

#### [4.3.1.4 Search Video](#_Toc268523787)

Diagram

Description automatically generated

Figure 4. 15

#### [4.3.1.5. Upload Video](#_Toc268523787)

Diagram

Description automatically generated

Figure 4. 16

# Chapter 5: Implementation

In this chapter , we explain all the implementation of our project. Why we are using the IDE PyCharm, the libraries and the language Python.

## 5.1 Language Python 3.7

* Python language is in the high demand in the market
* Python language is best choice for machine learning project because of its own libraries that make development very easy
* Helping material of python in form of documentation and video tutorial is widely available over internet.
* Python 3.7 support more libraries than latest version of python

## 5.2 Libraries

* Tkinter
  + Version 2.3.2
  + Easy to configure with database
  + It is best for design simple and user friendly interface
  + Provide 15 types of widgets that help in designing GUI

* Pymysql
* Version 1.0.2
* For database connection
* Pymysqls is best suitable with python
* OpenCV
* Version 4.5.1.48
* For video classification we need to analyze and classify the video, for that we use OpenCV.
* Tensorflow
* Version 2.0.5
* For classification and prediction of audio and video
* Kapre
* Version 0.3.5
* Preprocessor for audio data

## 5.3 Tool

# IDE (PyCharm 2020.2.3)

* PyCharm is smart code editor for python
* Readability of the code is good due to the different color of the keyword, classes and function etc.
* Free available on internet
* Compatible with all operating system
* Helps to write code in python quickly and efficiently

**WAMP**

* WAMP server version 3.2.3
* For database implementation
* WAMP provide support for MySQL
* We are developing a desktop app that will run without using any external resources, for that, WAMP is best choice

## 5.4 User Interface

Graphical user interface, application

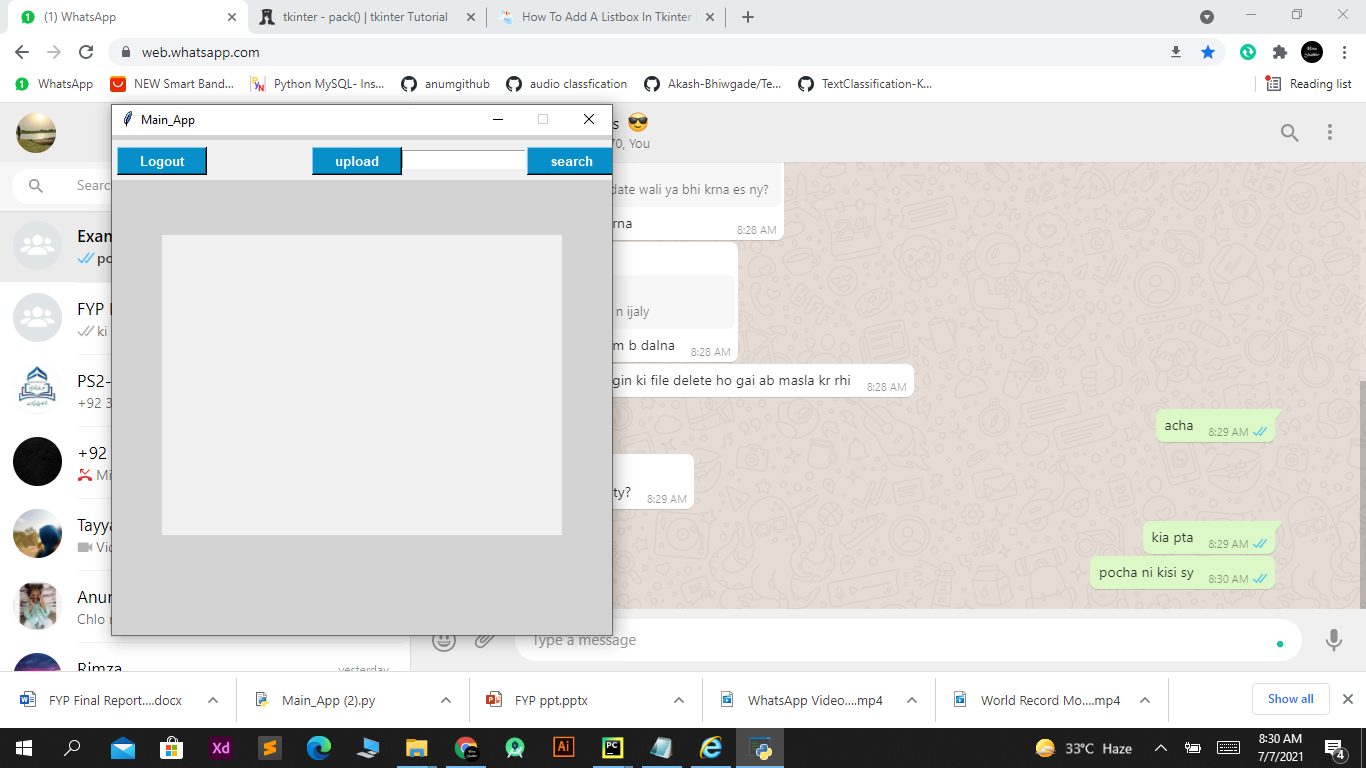
Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated



Graphical user interface, application

Description automatically generated

# Chapter 6: Testing and Evaluation

This Chapter includes unit testing detail of the system

## 6.1 Module / Unit Testing

### 6.1.1 Test Case Name: User Signup

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 1** | | | | | | | |
| **Test Case ID:** T\_01 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Signup | | | | | | |
| **Input:** Data for variable Name, gander, username, password and conform password. | | | | | | |
| **Test Title:** Entering all the Values that must be correct. | | | | | | |
| **Description:** User press the Signup button on splash screen, a new window open with following requirement, Name, gander, username(must be unique), password and conform password, user enter values against them and press Signup button. | | | | | | |
| **Pre-Condition:** Username should be unique | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Signup Button Clicked | Enter Name, gander, username, password and confirm password. | All data entered by user will store in database named userdb and show message to user Successfully signup. | All data entered by user will store in database named userdb and show message to user Successfully signup. | Pass | User get username and password to login to the system. |

Table 6.

### 6.1.2 Test Case Name: User Signup

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 2** | | | | | | | |
| **Test Case ID:** T\_02 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Signup | | | | | | |
| **Input:** Data for variable Name, gander, username, password and conform password. | | | | | | |
| **Test Title:** Entering all the Values that must be correct. | | | | | | |
| **Description:** User press the Signup button on splash screen, a new window open with following requirement, Name, gander, username, password and conform password, user enter values against them and press Signup button. | | | | | | |
| **Pre-Condition:** User should enter all the required detail correctly. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Signup Button Clicked | User Enter Name, gander, username, password and confirm password(Conform password not same, user enter wrong password). | All data entered by user will store in database named userdb. | Show error message Password and conform password should be same. | Fail |  |

Table 6.

### 6.1.3 Test Case Name: User Signup

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 3** | | | | | | | |
| **Test Case ID:** T\_03 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Signup | | | | | | |
| **Input:** Data for variable Name, gander, username, password and conform password. | | | | | | |
| **Test Title:** Entering all the Values that must be correct. | | | | | | |
| **Description:** User press the Signup button on splash screen, a new window open with following requirement, Name, gander, username, password and conform password, user enter values against them and press Signup button. | | | | | | |
| **Pre-Condition:** User should enter all the required detail correctly. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Signup Button Clicked | User Enter Name, gander, username(username not unique), password and confirm password. | All data entered by user will store in database named userdb. | Show warning Username is already exist | Fail |  |

Table 6.

### 6.1.4 Test Case Name: User Login

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 4** | | | | | | | |
| **Test Case ID: T\_04** | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Login | | | | | | |
| **Input:** Values against username and password | | | | | | |
| **Test Title:** Entering all the Values that must be correct. | | | | | | |
| **Description:** Test the login page User enter username and password. | | | | | | |
| **Pre-Condition:** User should Have an account and user name and password should correct. | | | | | | |
| **Dependencies:** User must be registered first. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Login Button Clicked | Enter username and password. | User enter username and password correctly, System granted access to user. | User enter username and password correctly, System granted access to user. | Pass |  |

Table 6.

### 6.1.5 Test Case Name: User Login

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 5** | | | | | | | |
| **Test Case ID:** T\_05 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Login | | | | | | |
| **Input:** Values against username and password | | | | | | |
| **Test Title:** Entering all the Values that must be correct. | | | | | | |
| **Description:** Test the login page User enter username and password. | | | | | | |
| **Pre-Condition:** User should Have an account and user name and password should correct. | | | | | | |
| **Dependencies:** User must be registered first. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Login Button Clicked | Enter username and password(user enter wrong password). | User enter username and password correctly, System granted access to user | Invalid Credential | Fail |  |

Table 6.

### 6.1.6 Test Case Name: User Logout

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 6** | | | | | | | |
| **Test Case ID:** T\_06 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** User Logout | | | | | | |
| **Input:** Press logout button | | | | | | |
| **Test Title:** Logout user from system | | | | | | |
| **Description:** User press logout button. | | | | | | |
| **Pre-Condition:** User login to the system | | | | | | |
| **Dependencies:** User must be login. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Logout Button Clicked |  | System logout the user and return the user to login screen. | System logout the user and return the user to login screen. | Pass |  |

Table 6.

### 6.1.7 Test Case Name: Upload Video

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 7** | | | | | | | |
| **Test Case ID:** T\_07 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Upload Video | | | | | | |
| **Input:** A Video of type mp4 | | | | | | |
| **Test Title:** Upload a video | | | | | | |
| **Description:** User click on **‘Upload’** button then choose video from computer to upload to the system and press open button. | | | | | | |
| **Pre-Condition:** User should be login and have video. | | | | | | |
| **Dependencies:** User must be login. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | User click on upload button then choose video from computer to upload to the system and press open button. | Video.mp4 | User select a video for upload to the system and then video store in the database. | User select a video for upload to the system and then video store in the database. | Pass |  |

Table 6.

### 6.1.8 Test Case Name: Upload Video

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 8** | | | | | | | |
| **Test Case ID:** T\_08 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Upload Video | | | | | | |
| **Input:** A Video of type mp4 | | | | | | |
| **Test Title:** Upload a video | | | | | | |
| **Description:** User click on **‘Upload’** button then choose video from computer to upload to the system and press open button. | | | | | | |
| **Pre-Condition:** User should be login and have video. | | | | | | |
| **Dependencies:** User must be login. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | User click on upload button then choose video from computer to upload to the system and press open button. | Video.wmv (video format is not supported) | User select a video for upload to the system and then video store in the database. | Video file format is mis match. | Fail |  |

Table 6.

### 

### 6.1.9 Test Case Name: Video Splitting

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 9** | | | | | | | |
| **Test Case ID:** T\_09 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Video Splitting | | | | | | |
| **Input:** A Video of type mp4 | | | | | | |
| **Test Title:** Split a video | | | | | | |
| **Description:** When user upload a video, System performs splitting using OpenCV library and split video in audio and visual. | | | | | | |
| **Pre-Condition:** Video should be type of mp4. | | | | | | |
| **Dependencies:** User must be login and have a video. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | User upload a video and then system perform splitting, video split into visual. | Video.mp4 | Video split into visual and store in different folder. | Video split into visual and store in different folder. | Pass |  |

Table 6.

### 

### 6.1.10 Test Case Name: Video Splitting

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 10** | | | | | | | |
| **Test Case ID:** T\_10 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Video Splitting | | | | | | |
| **Input:** A Video of type mp4 | | | | | | |
| **Test Title:** Split a video | | | | | | |
| **Description:** When user upload a video, System performs splitting using OpenCV library and split video in audio and visual. | | | | | | |
| **Pre-Condition:** Video should be type of mp4. | | | | | | |
| **Dependencies:** User must be login and have a video. | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | User upload a video and then system perform splitting, video split into visual. | Video.wmv (video format is not supported) | Video in visuals and store in different folder. | Video file format is mis match. | Fail |  |

Table 6.

### 6.1.11 Test Case Name: Search Video

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 11** | | | | | | | |
| **Test Case ID:** T\_011 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Search Video | | | | | | |
| **Input:** Title of video. | | | | | | |
| **Test Title:** Enter the title of video. | | | | | | |
| **Description:** User can search any type of video using search bar. | | | | | | |
| **Pre-Condition:** User must be Login and must have title or keywords that relate to the video. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Click on the Search bar | Enter the title of the video | List of video should be displayed. | List of video should be displayed. | Pass |  |

Table 6.

### 6.1.12 Test Case Name: Search Video

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 12** | | | | | | | |
| **Test Case ID:** T\_012 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Search Video | | | | | | |
| **Input:** Title of video. | | | | | | |
| **Test Title:** Enter the title of video. | | | | | | |
| **Description:** User can search any type of video using search bar. | | | | | | |
| **Pre-Condition:** User must be Login and must have title or keywords that relate to the video. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Click on the Search bar | Enter the wrong title (Title not related to sports video) | List of video should be displayed. | System give a message, video not found. | Fail |  |

Table 6.

### 6.1.13 Test Case Name: Video Classification

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 13** | | | | | | | |
| **Test Case ID:** T\_013 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Video Classification | | | | | | |
| **Input:** Type of visual content that detect from video | | | | | | |
| **Test Title:** Video classification through visual content | | | | | | |
| **Description:** Type of content in the visual is already detected so here output of that process and through this classification of video will be perform using OpenCV, Pandas, Keras and Deep learning algorithms. | | | | | | |
| **Pre-Condition:** All visual should be preprocessed and in the form of jpeg. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Video Model | Type of visual content that detect from video. | Video will be classify and its type should be define in system. | Video will be classify and its type should be define in system. | pass |  |

Table 6. 13

### 6.1.14 Test Case Name: Video Classification

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 14** | | | | | | | |
| **Test Case ID:** T\_014 | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Video Classification | | | | | | |
| **Input:** Type of visual content that detect from video | | | | | | |
| **Test Title:** Video classification through visual content | | | | | | |
| **Description:** Type of content in the visual is already detected so here output of that process and through this classification of video will be perform using OpenCV, Pandas, Keras and Deep learning algorithms. | | | | | | |
| **Pre-Condition:** All visual should be preprocessed and in the form of jpeg. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Video Model | Visual not preprocessed | Video will be classify and its type should be define in system. | System will not allow to classify the video because visual will be not preprocessed. | fail |  |

Table 6. 14

### 6.1.15 Test Case Name: Audio Classification

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 15** | | | | | | | |
| **Test Case ID: T\_015** | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Audio classification | | | | | | |
| **Input:** Audio in the type of wav. | | | | | | |
| **Test Title:** Enter audio in the form of wav. | | | | | | |
| **Description:** When system perform splitting, later audio will be process, using audio classification model system will predict the audio type. | | | | | | |
| **Pre-Condition:** Audio should be of type wav. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Audio Model | Audio in the type of wav. | Classification of audio, audio will save in separate folder | Classification of audio, audio will save in separate folder | Pass |  |

Table 6. 15

### 6.1.16 Test Case Name: Audio Classification

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Project Name:** Video Classification and Indexing | | **Test Case: 16** | | | | | | | |
| **Test Case ID: T\_016** | | | | | | |
| **Test Priority (Low/Medium/High):** High | | | | | | |
| **Module Name:** Audio classification | | | | | | |
| **Input:** Audio in the type of wav. | | | | | | |
| **Test Title:** Enter audio in the form of wav. | | | | | | |
| **Description:** When system perform splitting, later audio will be process, using audio classification model system will predict the audio type. | | | | | | |
| **Pre-Condition:** Audio should be of type wav. | | | | | | |
| **Dependencies:** | | | | | | |
| **Steps** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 01: | Audio Model | Audio not in the type of wav. | Classification of audio, audio will save in separate folder | Audio is not a type of wav classification will not perform. | Fail |  |

Table 6. 16

## 6.2 Integration Testing

Integration testing is the technique for testing the interfaces of software components. Each software components in unit tested, and all the components are integrated and evaluated as a group. The tests are conducted to ensure that the components are working properly after interfacing.

All modules are integrated by an incremental approach, and integration testing of the system is performed as follows:

* Integrate login module and test.
* Integrate Video Classification and Indexing modules and test.

# Chapter 7: Conclusion and Future Work

In this chapter we are concluding our FYP work by analyzing what we were aim to do what we have done till now and what are improvements that still needs in this project or what kind of future work can anyone perform.

## 7.1 Conclusion

Our FYP is about video classification and indexing which is about classifying video in different category and index them.

Till now we have trained our system on a sports dataset and have created 8 classes to classify videos like basketball, boxing, cricket, kabaddi, formula1, swimming, table-tennis and weight lifting. We have also trained our system to classify audio which will classify the audio in two categories music and speech for that we have used a dataset of wav files.

What we have to do is to upload a video from the GUI we have created and the system will split the video into frames and audio than it will classify the audio whether it is speech or music than it will classify each frame from the data provided by the model we have trained and then it will write that category name on the video and place it in that category folder and the database. We have further perform indexing the all the videos that are uploaded so it is easier to search the videos from the GUI of the system.

Our aim was to make video classifying and indexing system and we have made system that can classify video up-to 8 categories and audio from the video is also classified into 2 categories and our system videos that are uploaded are indexed so it is easy to search. More categories can also be added by training the system on more categories through more datasets.

## 7.2 Future Work

We were simply focused on classify video on bases of content in the video and analyzing audio of the video into limited categories. We developed a desktop app for it. Also system can be trained on more and different categories for different platforms.

Future work related to this project is, you can develop a video platform like mobile app or website by implementing video classification which will improve the backend working of your system.

## References

[1]

“ ResearchGate ” research.net [online]. Available:

<https://www.researchgate.net/publication/3421929> [ Accessed : 22-oct-2020]

[2]

“ TRO India ” troindia.in [online]. Available:

<http://troindia.in/journal/ijcesr/vol4iss12/64-69.pdf> . [Accessed : 22-oct-2020]

[3]

“ fuji xerox” fujixerox.com [online]. Available:

[https://www.fujixerox.com/eng/company/technology/production/multimedia/talkminer.html/](https://www.w3schools.com/). [Accessed : 22-oct-2020]

[4]

“ GitHub” github.com [online]. Available:

<https://github.com/marcogdepinto/emotion-classification-from-audio-files>

[Accessed : 18-july-2021]

[5]

“ Kaggle” kaggle.com [online]. Available:

<https://www.kaggle.com/lnicalo/gtzan-musicspeech-collection>

[Accessed : 29-May-2021]

[6]

“ GitHub” github.com [online]. Available:

<https://github.com/jurjsorinliviu/Sports-Type-Classifier/tree/master/data>

[Accessed : 15-May-2021]