**MAKERERE UNIVERSITY**

**COLLEGE OF COMPUTING AND INFORMATION SCIENCES**

**DEPARTMENT OF NETWORKS**

**BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING (YEAR 2)**

**RECESS TERM 2**

**COURSE CODE: BSE 2301**

**PROPOSAL FOR YOUTUBE TRENDING STATISTICS**

**GROUP NUMBER: G-09**

|  |  |  |
| --- | --- | --- |
| NAME | REG.NO | STUDENT NO. |
| SEMPA ETHAN IAN | 16/U/11403/PS | 216004900 |
| KWESIGA PETER | 16/U/6411/EVE | 216002578 |
| NABAGIDDE GRACE DIANA | 16/U/8005/PS | 216012180 |
| BWIRE IVAN | 16/U/4522/PS |  |

**SUPERVISOR: NOAH KANGE**

**PROJECT LEADER: KWESIGA PETER**

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE SOFTWARE ENGINEERING RECESS PROJECT BSE 2301

***SOFTWARE DESIGN DOCUMENT***

***YOUTUBE TRENDING STATISTICS SYSTEM***

TABLE OF CONTENTS

[**1.** **INTRODUCTION** 4](#_Toc520167241)

[**1.1** **PURPOSE** 4](#_Toc520167242)

[**1.2** **SCOPE** 4](#_Toc520167243)

[**1.3** **DEFINITIONS, ACRONYMS AND ABBREVIATIONS** 4](#_Toc520167244)

[**1.4** **OVERVIEW** 4](#_Toc520167245)

[**2.** **SYSTEM OVERVIEW** 5](#_Toc520167246)

[**2.1** **SYSTEM CHARACTERISTICS** 5](#_Toc520167247)

[**2.2** **SYSTEM ARCHITECTURE** 5](#_Toc520167248)

[**3.** **SYSTEM CONTEXT** 6](#_Toc520167249)

[**4.** **SYSTEM DESIGN** 8](#_Toc520167250)

[**4.1** **DESIGN METHOD AND STANDARDS** 8](#_Toc520167251)

[**4.2** **NAMING CONVENTIONS** 8](#_Toc520167252)

[**4.3** **PROGRAMMING STANDARDS** 8](#_Toc520167253)

[**4.4** **SOFTWARE DEVELOPMENT TOOLS** 8](#_Toc520167254)

[**4.5** **OUTSTANDING ISSUES** 9](#_Toc520167255)

[**4.6** **DECOMPOSITION DESCRIPTION** 10](#_Toc520167256)

[**5.** **COMPONENT DESCRIPTION** 11](#_Toc520167257)

[**5.1** **Module 1: Popularity check** 11](#_Toc520167258)

[**5.2** **Module 2: Recommended publish time in order to achieve trending status** 11](#_Toc520167259)

[**5.3** **Module 3: Sentimentality check** 12](#_Toc520167260)

[**5.4** **Module 4: Statistical and relation visualization** 12](#_Toc520167261)

[**6.** **SOFTWARE REQUIREMENTS TRACEABILITY MATRIX** 12](#_Toc520167262)

LIST OF FIGURES

[Figure 1: Architectural Diagram 5](#_Toc520167207)

[Figure 2: Context diagram 7](#_Toc520167208)

[Figure 3: Functional Decomposition Diagram 10](#_Toc520167209)

[Figure 4: Module 1 determines which videos are popular based on likes. 11](#_Toc520167210)

[Figure 5: Module 2 shows what time is favorable to post a video in order for it to trend. 12](#_Toc520167211)

LIST OF TABLES

[Table 1: Definitions, Acronyms and Abbreviation 4](#_Toc520167199)

[Table 2: Conventions 8](#_Toc520167200)

[Table 3: RTM 13](#_Toc520167201)

# **INTRODUCTION**

The Software Design Document is one that provides documentation which will be used to aid in software development by providing details for how the software should be built. [1] Within the SDD are narrative and graphical documentation of the software design for the project including context diagram, flow charts, architectural diagram and a functional decomposition diagram.

## **PURPOSE**

The purpose of the SDD is to provide a description of the design of the YouTube Trending Statistics System fully enough to allow for software development to proceed with an understanding of what is to be built and how it is expected to be built.

## **SCOPE**

The complete SDD will contain the general definition and features of the project, design constraints and overall system architecture. With the help of decomposition diagrams, design of the system and sub-systems will be explained visually in order to help the programmers to understand the information stated in this document correctly and easily.

## **DEFINITIONS, ACRONYMS AND ABBREVIATIONS**

|  |  |
| --- | --- |
| **DEFINITION, ACRONYMS AND ABBREVIATIONS** | **INTERPRETATION** |
| SDD | Software Design Document |
| RTM | Requirement Traceable Matrix: Captures all requirements proposed by the client or development team. It generally maps and traces user requirement with test cases. |
| URL | Uniform Resource Locator |
| FDD | Functional Decomposition Diagram |
|  |  |

Table 1: Definitions, Acronyms and Abbreviation

## **1.4 REFERENCES**

[1] R.McElrath (2007,04.20) XML Legal Document Utility Software Design Document

[2] (2014, 04.09) [OMG Unified Modeling Language (OMG UML), Superstructure. Version 2.4.1"](http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF).  Object Management Group.

## **OVERVIEW**

The SDD is divided into 6 sections with various subsections. these sections include:

1. Introduction
2. System overview
3. System context
4. System design
5. Component description
6. Software Requirements Traceability Matrix

# **SYSTEM OVERVIEW**

The YouTube Trending Statistical System is data analysis system that provides meaningful feedback to YouTube management as well as the channel owners using data collected over a period of time. Using the data set collected, the system is able to identify the different patterns, correlations and factors that make particular videos successful. This information is presented in form of graphs, statistical numbers and charts.

The system is designed based on a two tier client server architecture. The server contains the core functionality and data. It handles the major services provided by the system using the data set provided. The client side provides the requests, inputs and provides the user interface to ease use of the system. Due to the simplicity of the system and the little need much, the client server architecture would be easy to implement and develop.

.

## **SYSTEM CHARACTERISTICS**

* + The system is to function using a data set provided which will be read and organized using the server logic. This will create the visual representations which will be seen by the client with the help of the provided interface.
  + The data sets will be added or provided by the designated software developers which will be handled by the server logic. Only the software developers will have the access to the server and will change logic according to the data set provided.
  + The system is easily scalable as it can support an increase in the number of users as the server provides the same data to multiple users that is to say clients simply request for the page. Therefore, the only hindering factor would be traffic on the internet.

## **SYSTEM ARCHITECTURE**

SERVER

CLIENT

Figure 1: Architectural Diagram

Client:

The client will be responsible for handling the requests of the user. Using the controls provided, the user will be able to control the output. For example, the user can be able to use the select input to select which region the graphs should be focused on. This provides an easier way to communicate with the system.

Server:

The server is responsible for providing the logic necessary to provide the output requested by the client. Using the data set the server logic will be able to plot the graphs charts and statistical numbers to be displayed to the user on the provided interface.

# **SYSTEM CONTEXT**

External interfaces

1. Browser

It enables communication between the shiny application and the shiny server as the shiny application sends data in its raw form to the shiny server and requests for visualization of that data for example in form of a heat map and the shiny server sends the appropriate response to the shiny application. This wouldn’t be possible without a browser. On top of this, it enables interaction between the entities in this case, Channel Owners and YouTube Administration and the Shiny Application. As entities, they request for particular statistical reports and these reports are displayed to them.

1. Operating System

It enables communication between the Shiny Application and the dataset as the operating system will look for the dataset in memory and send the raw data to the application.

Data formats

The Shiny Application and Shiny Server communicate with the browser in the form of HTTP message format in the form of request and response messages.

The Shiny Application communicates with the operating systems through the use of system calls.

*Context Diagram*

The diagram below explains how the main system (Shiny Application) interacts with various systems and entities to produce desired output.

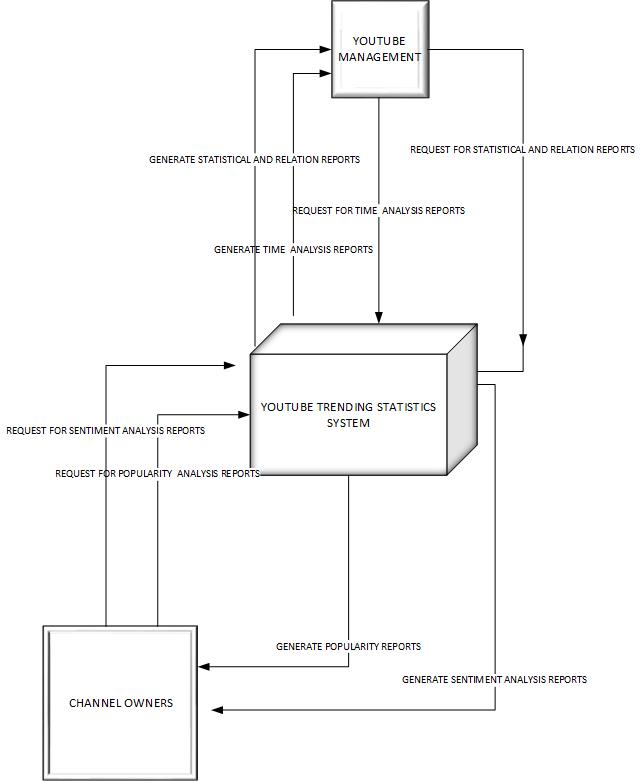


Figure 2: Context diagram

# **SYSTEM DESIGN**

## **DESIGN METHOD AND STANDARDS**

Unified Modelling Language(**UML**) will be used to visualize the design of the system. The latest version of UML that is to say UML 2.4.1 will be used in this case. [2] UML visualizes the architectural blueprints in a diagram that is to say structural and behavioral views of the system for instance any activities done by the system, individual components of the system and how they interact with other software components, how the system will run, how entities will interact with others and external user interfaces.

## **NAMING CONVENTIONS**

* All programs should be written in the format **underscore\_separate** format for instance *display\_chart.*
* All modules should be written in the format **UpperCamelCase** for easy understanding that is to say that words are capitalized both when the name consists of a single word for instance *Capture*, or multiple words, as in *ExtractData.*
* All variables are to written in **alllowercase** format for example *sum*.
* All files are to written in alllowercase format as well.

|  |  |
| --- | --- |
| File type | Extension |
| Comma separated values file | .csv |
| R script | .r |
| File | .txt |

Table 2: Conventions

## **PROGRAMMING STANDARDS**

* File names should have meaning e.g. display\_graph
* When indenting code, use two spaces and **never** mix tabs and spaces.
* Place a space around binary operators and do not place a space before a comma but always place one after a comma.
* Use < - for assignment instead of =
* Do not terminate lines with semicolons.
* Comment code as much as possible especially essential modules by use of # and a space.
* Function definitions should first list arguments without default values, followed by those with default values.
* Syntax lines should not be long that is to say 80 characters maximum
* An opening curly brace should never go on its own line; a closing curly brace should always go on its own line.

## **SOFTWARE DEVELOPMENT TOOLS**

* RStudio will be used to write, compile and run scripts as well as designing Shiny applications.
* Microsoft Word will be used to write the necessary documentation of the project.
* Pencil and Microsoft Visio will be used for drawing diagrams required.
* The ***shinyloadtes****t* package and the accompanying ***shinycannon*** software enable load testing deployed shiny applications.
* GitHub will be used for source code management.

## **OUTSTANDING ISSUES**

There are no outstanding issues that have arisen per now.

## **4.6 DECOMPOSITION DESCRIPTION**

*Functional Decomposition Diagram*

This diagram shows the hierarchy, control flow, data flow between the components.

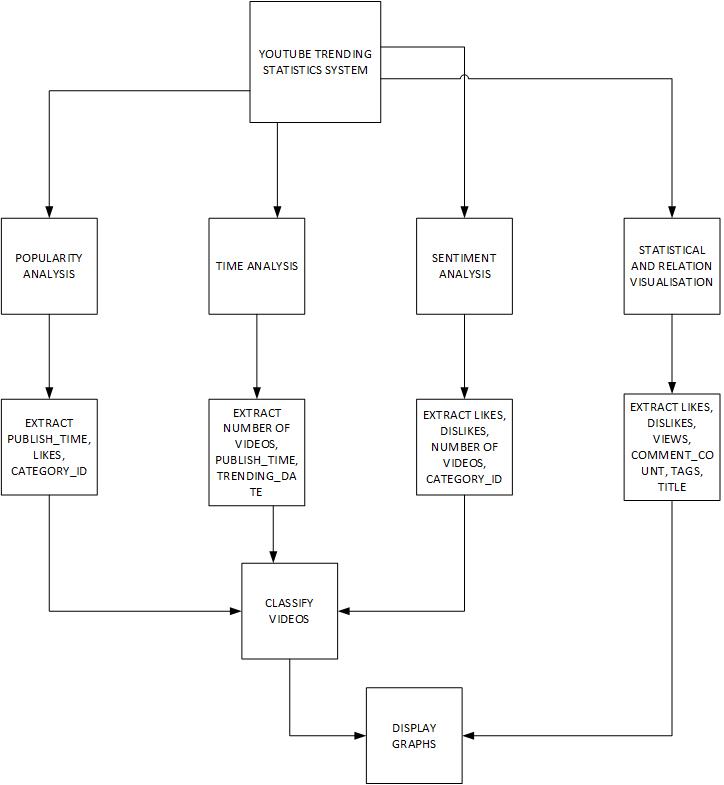


Figure 3: Functional Decomposition Diagram

# **COMPONENT DESCRIPTION**

## **Module 1: Popularity check**

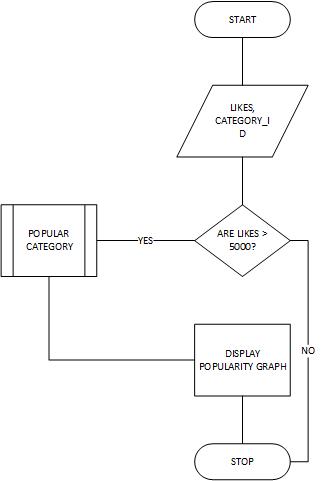


Figure 4: Module 1 determines which videos are popular based on likes.

## **Module 2: Recommended publish time in order to achieve trending status**

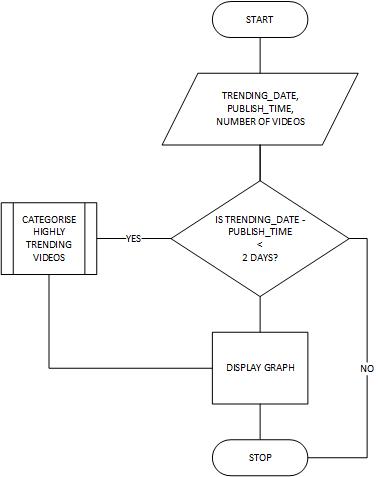


Figure 5: Module 2 shows what time is favorable to post a video in order for it to trend.

## **Module 3: Sentimentality check**

In this case, we get the most liked and most disliked videos through finding average likes or dislikes per category. Then we find the ratio of total likes to total dislikes irrespective of total likes to further determine if a video was truly positively/ negatively reacted to.

## **Module 4: Statistical and relation visualization**

Here, we extract the likes, dislikes, views and comment count from the data set and show their various relationships i.e. how they correlate per category id through scatter plots. We also extract tags and titles and perform a word cloud that determines the most used word in the given data per region.

# **SOFTWARE REQUIREMENTS TRACEABILITY MATRIX**

|  |  |
| --- | --- |
| REQUIREMENT | REFERENCE SECTION |
| External interface requirements | 3 in the SRS |
| Functional requirement | 4 in the SRS |
| Non-functional requirement | 5 in the SRS |

Table : RTM