ABSTRACT

Albumizer is comprehensive software that converts raw photographs into unique designer photo books in a smart, fast easy way. It is a combination of Photoshop and the skills of a professional designer; a mix that makes it one of the most advanced and intelligent album designing software available today. Complex algorithms and creative enhancements give you real power to create amazing album designs. The themes, frames and backgrounds which are being stored using relative database is not efficient. It becomes difficult to store and manage the database and the scalability reduces due to inefficient method of storage and accessibility becomes low. with the increase in the number of users, the requirement for themes will increase. To manage the storage and retrieval of huge volume of themes, we propose theme builder application which can manage large data base using Hadoop technology. Here the themes are stored using file systems in hadoop for user application. Map-reduce algorithm is used to map the themes created. CBIR is used at the client side to select themes of interest based on Euclidean distance of user selected image and the set of stored images. Once the user selects the themes they will be provided access to the cloud, if they are the authorized users of Albumizer.

keywords: Hadoop, Mapreduce.

ACKNOWLEDGEMENT

We take this opportunity to express profound sense of gratitude and respect to all those who helped us through the duration of this mini project work. First of all, we thank our parents for being our strength and the whole college that gave us the opportunity to carry out the project on 'Theme Builder' application.

We would like to thank our project coordinator Mr.Amit Gundad, Lecturer, Department of Information Science and Engineering, BVBCET Hubli, whose guidance and knowledge was great use to us during this project work.

We would like to extend our thanks to our guide, Mrs. Namrata Hiremath for the guidance and support throughout this project and for their constant source of inspiration to us.

We would like to express our sincere thanks to Dr.Meena S.M, Professor and Head of Department, Information Science and Engineering, BVBCET Hubli, for providing us an opportunity to carry out our major project work successfully.

We find our acknowledgement incomplete without thanking our Principal Dr.Ashok Shettar, BVBCET Hubli for the inspiration and co-operation.

Last but not the least we like to thank all the staff members, teaching and non-teaching staff for helping us during the course of the project.

Meghana Ellur - 2BV11IS049
Nikhil Kalal - 2BV11IS054
Suma Doddmani - 2BV11IS125
Pooja Kulkarni - 2BV11IS127

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

This chapter gives brief introduction about the project report, overall idea of the project, its problem definition, objective of the project, limitations of the existing systems which was the main motivation behind choosing of this project.

The existing system of storing the themes, frames and backgrounds in database is not efficient. Since these are in JPG, PNG format. Thus, it becomes difficult to store and manage the database and the scalability reduces due to inefficient method of storage and accessibility becomes low. As the number of users increases, the theme combinations also increase which is to be stored in database, thus that would lead to inefficient management of the data. Hence we propose theme builder application which can manage large data base using Hadoop technology.

Theme builder is an application which is used to create themes using the available frames and backgrounds. This is possible in two ways. First method is to use the relational database. The duplication of the data was the disadvantage of the system when relational database is used .Here the database is the collection of the themes. These themes are huge in number. So to handle the huge data a second method is used. Second method includes the processing of the themes in the hadoop. Apache Hadoop is an open-source software framework for distributed storage and distributed processing of Big Data. Hadoop Distributed File System (HDFS) splits files into large blocks (default 64MB or 128MB) and distributes the blocks amongst the nodes in the cluster. For processing the data, the Hadoop Map/Reduce ships code (specifically Jar files) to the nodes that have the required data, and the nodes then process the data in parallel.

CBIR is used at the client side to select themes of interest based on Euclidean distance of user selected image and the set of stored images. Once the user selects the themes he will be provided access to the cloud, if he is an authorized user of Albumizer.

1.1 Literature Survey/Existing System

This section describes the survey done for the technologies that could be used to overcome the problem identified. It includes the knowledge gained in the literatures that we have used in learning our project.

The method of creating photo book involves storing and retrieval of images. Existing method uses simple data base methodologies. With the increasing number of backgrounds, frames and sizes degrades the handling of database. Hence to accomplish this, we studied different technologies like map-reduce, hive storage, pig-technique etc.

1.1.1 Paper 1

Title - Hadoop and its Role in Modern Image Processing

Authors - Seyyed Mojtaba Banaei, Hossein Kardan Moghaddam

Published - Open Journal of Marine Science, 2014, 4, 239-245

The huge volume of visual data in recent years and their need for efficient and effective processing stimulate the use of distributed image processing frameworks in image processing area. So that up to the coming years, many algorithms which have been introduced in the field of image processing and pattern recognition should consider the requirements for macro image processing in order to be welcomed by the outside world. This paper gives an overview of distributed processing methods and the programming models. Also some works are studied which have been done in recent years using Hadoop open source framework. Hadoop and its processing model are newly formed and like any other new technologies may have its own issues, such as lack of familiarity of the majority of IT society with it, lack of enough expert forces, and unwanted defects and problems due to its novelty. However, this processing style that uses MapReduce model and distributed file system, will be among the most useful tools for image processing and pattern recognition in the coming years due to its consistency with cloud computing structures.

1.1.2 Paper 2

Title - The rise of "big data" on cloud computing: Review and open research issues

Authors - IbrahimAbakerTargioHashem, IbrarYaqoob, NorBadrulAnuar, Salimah Mokhtar, AbdullahGani and SameeUllahKhan

Published - Information Systems 47 (2015)98–115

The size of data at present is huge and continues to increase every day. The variety of data being generated is also expanding. These data provide opportunities that allow businesses across all industries to gain real-time business insights. The use of cloud services to store, process, and analyze data has been available for some time; it has changed the context of information technology and has turned the promises of the on-demand service model into reality. In this study, we presented a review on the rise of big data in cloud computing. We proposed a classification for big data, a conceptual view of big data, and a cloud services model. This model was compared with several representative big data cloud platforms. The background of Hadoop technology and its core components, namely, MapReduce and HDFS is explained in this paper. Some of the challenges in big data processing has also been discussed. The review covered volume, scalability, availability, data integrity, data protection, data transformation, data quality/heterogeneity, privacy and legal/regulatory issues, data access, and governance. Furthermore, the key issues in big data in clouds are highlighted.

1.1.3 Paper 3

Title - Implementation of Content Based Image Retrieval Using Clustering technique

Authors - Junaid Khan and Prof. S.S. Kulkarni

Published - IJCSMC, Vol. 3, Issue. 5, May 2014, pg.523 – 529

The main objective of the image clustering is to remove the data loss and extracting the meaningful information to the human expected needs. The images are pre-processed with various techniques. Here, images are clustered based on RGB Components, Texture values and K- mean algorithm. This application can be used to classify the medical images in order to diagnose the right disease verified earlier

1.2 Motivation

The existing system of storing the themes, frames and backgrounds in database is not efficient. Since these are in JPG, PNG format, it becomes difficult to store and manage the database. Thus the scalability reduces due to inefficient method of storage and accessibility becomes low. As the number of users increase, the theme combinations also increase which is to be stored in database, which would lead to inefficient management of the data

1.3 Problem Statement

Developing theme builder for multimedia content using hadoop.

Theme builder for multimedia content refers to the dataset which includes the data i.e., the frames and backgrounds which constitute the multimedia content. The goal of Theme builder is to handle with the frames and backgrounds and to create the themes as required. The theme package which is built is later uploaded on to the cloud. So the availability and the accessibility of the application increases.

Chapter 2

REQUIREMENT ANALYSIS

The purpose of this chapter is to provide the software requirement specification to build theme builder application. This document provides the intermediate steps involved in the application and is meant to outline the features of the project so as to serve as a guide to developers on one hand and also describe the functionalities of the product to the users on the other. It also defines the requirements for a system and the methods to be used to ensure that each requirement is satisfied.

2.1 System Model

This section describes the system model of theme builder application.

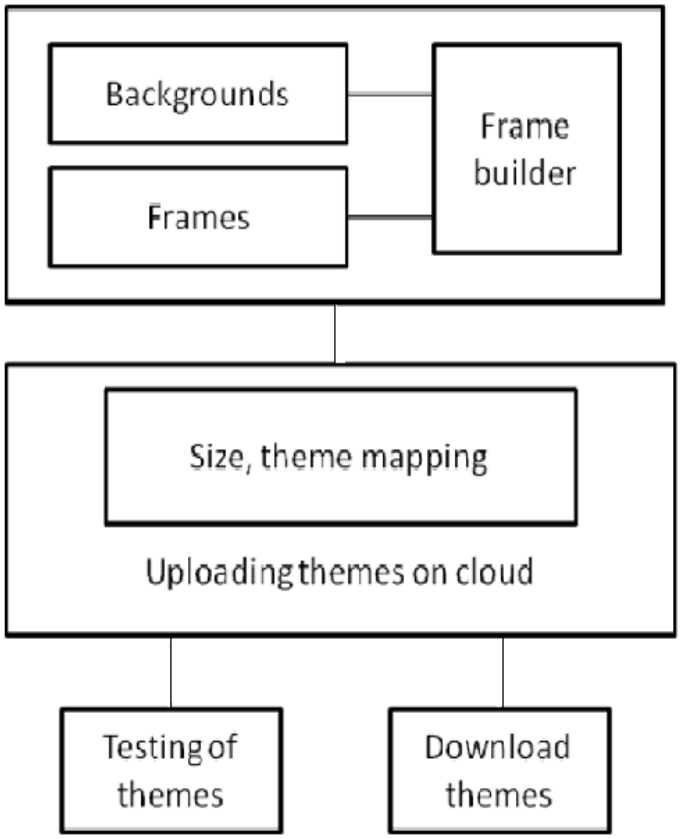


Figure 2.1: system model

In the first module, the backgrounds and frames are used in building of themes using various combinations. The data stored in present system is in the database. As storing of unstructured data (PNG, JPG) in database becomes inefficient, using hadoop the unstructured data is converted into structured form. This includes matching of color formats for backgrounds, graphics and frames. Matching of graphics is done for different formats like Event Text Graphics (ETG), Event Graphics (EG), Paragraph Graphics (PG) and Regional Graphics (RG).

The frames built in the first module is input for second where the theme built will be mapped to different sizes and is uploaded into the cloud.

The uploaded themes are tested for proper working for real time scenarios and testing is also done to check proper downloading of themes at user side.

2.2 Functional Requirements

This section describes the functional requirements of the theme builder application. Each system provides the functionality as listed and the final output is cumulatively displayed

2.2.1 Software developer functionalities

This section describes the functional requirements of the Software developer.

- Developer shall be able to map theme with all backgrounds and frames
- Developer shall be able to test theme for all the dimensions (sizes).
- Developer shall be able to upload theme package to server and shall be allowed to perform functionality test.

Theme Builder has two entities one is the software developer and the other entity is the user. Here the software developer has various functionalities. Developer is allowed to login to the system. Then he is allowed to select the background and themes and is allowed to generate the new themes according to the specification requirements given by the users. Then developer is allowed to map the themes with the predefined size and he is allowed to upload the theme package to the server. Developer is allowed to perform the functionality tests. He checks for the proper working of the modules by giving different inputs. Then he checks for the proper downloading of the theme packages which are uploaded to the server.

The following use-case diagram shows the functions that could be performed by developer perspective.

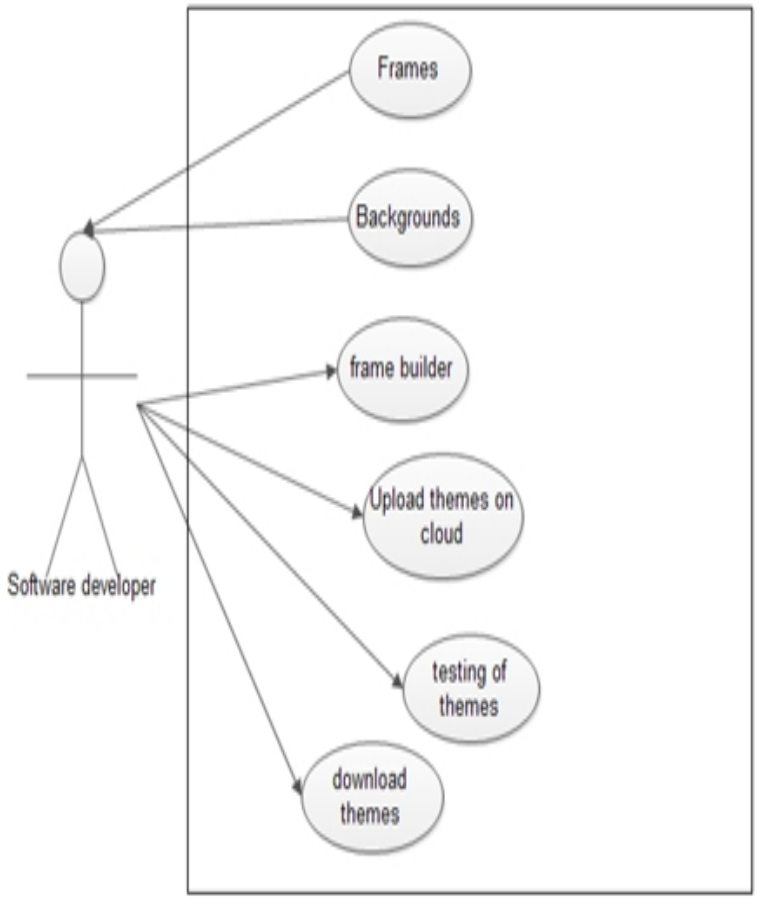


Figure 2.2: Use case for software developer

Use-case Name	Theme Builder
Actor	Software Developer
Pre condition	Developer should have access to the database
Normal Scenario	Mapping backgrounds, frames and size
Post Condition	Developer is able to build the new theme

Figure 2.3: use case description for software developer

2.2.2 User Functionalities

This section describes the functional requirements of the User.

- User shall be allowed to login to the system and shall be allowed to select the theme package which he is interested in.
- User shall be allowed to download the selected package.

User is another entity of the Theme Builder system. User is allowed to login to the system by providing his username and password. Then he is permitted to access the system. User is allowed to select the theme package and is allowed to download it.

The following use case diagram how the functionalities that could be performed by the users.

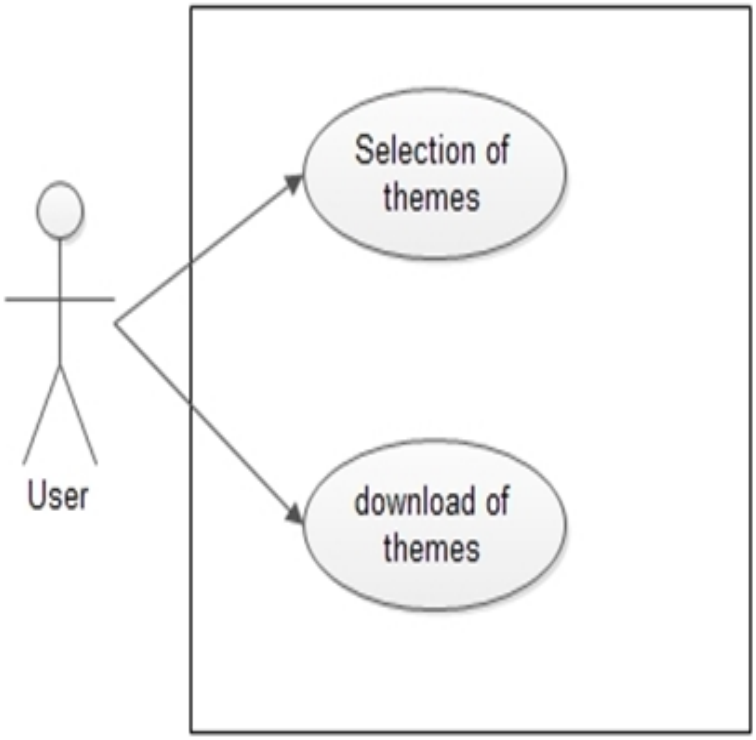


Figure 2.4: use case for user

Use-case Name	Theme Manager
Actor	User
Pre condition	User should have authenticated username and password
Normal Scenario	Selecting the required package and downloading
Post Condition	User is able to download the required packages.

Figure 2.5: use case description of user

2.3 Non Functional Requirements

This section describes the non functional requirements of the theme builder application.

- Performance : Scalability improves with the use of cloud
- Availability: The application will be easily available with 24X7 accessibility.
- Maintainability: Security and software updating is required to maintain a healthy server capable of serving its clients.

2.4 Database Requirements

In addition to the regular database we require hadoop technology (database for big data) to store the themes created. Hadoop is an open-source software framework for storage and large-scale processing of data-sets. It achieves reliability by replicating the data across multiple hosts. Since we are using simple file systems, hadoop mainly is useful for storing and mapping of images.

2.5 Function Means Tree

In engineering design, a function means tree is a method for functional decomposition and concept generation. At the top level main functions are identified. Under each function, a means is attached. Alternative solution elements are also attached.

In the fig shown, the selected solutions are colored and alternatives are left uncolored. The tree also represents the subsystems of the main system and their testing.

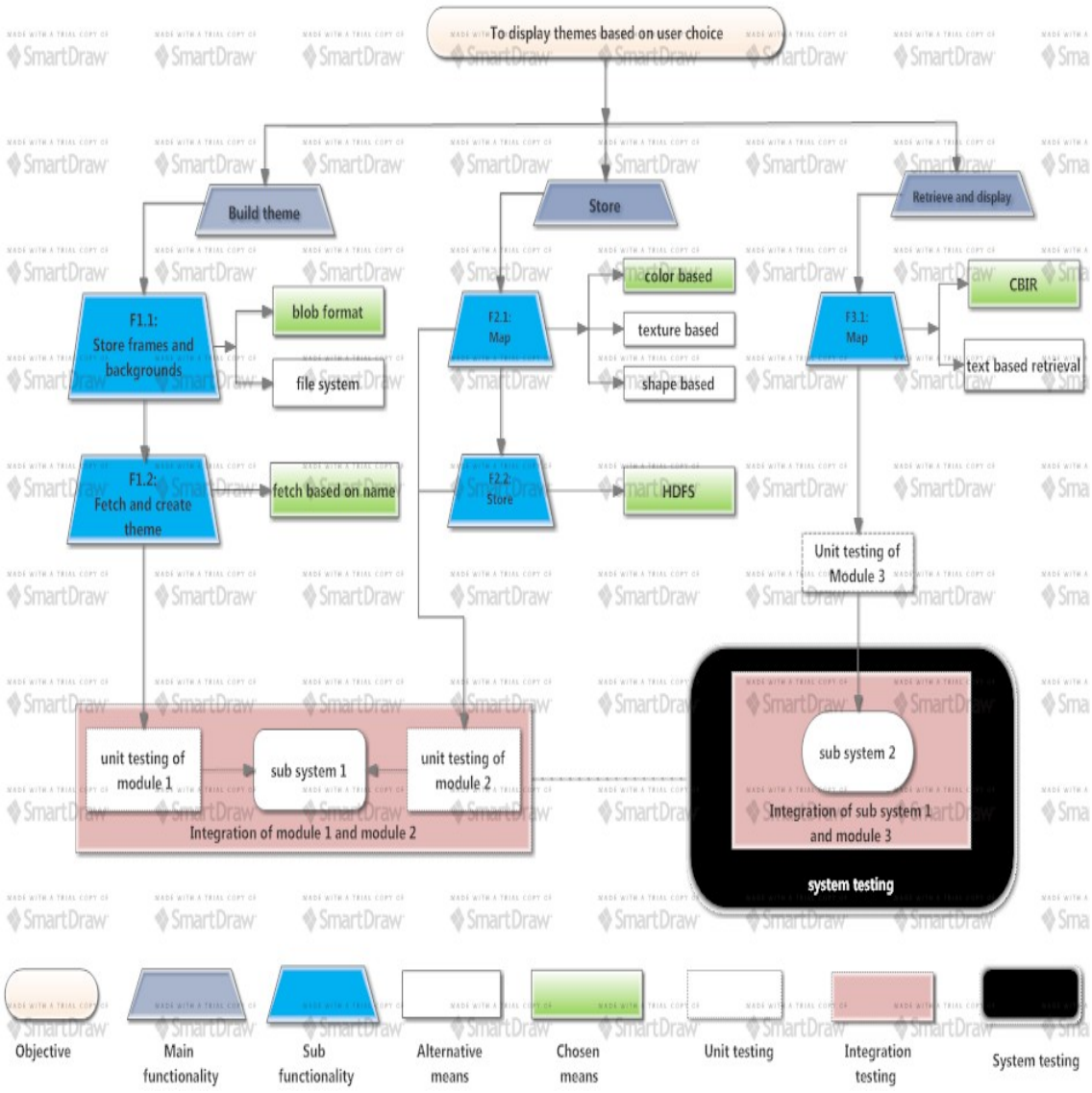


Figure 2.6: function means tree for theme builder

Chapter 3

SYSTEM DESIGN

This section describes the architecture design, database design of the Theme Builder application and provides overall guidance of the architecture of the system. It accompanies the architecture design with the detailed specification of the components of the system. It also specifies alternative architectural design.

Software design document is a stable reference, outlining all parts of the software and how they will work. This document is commanded to give fairly complete description, while maintained a high-level view of the software and serves as a references for the team members

3.1 Architecture Design

Theme builder is divided into three modules as theme builder, theme uploader and Functionality test. This section gives the architecture design (Activity diagram) for the individual modules. The architecture design specifies the design components that collaborate to perform all the functions included in the system.

Each of these components has an abstract description concerning the services that it provides to the rest of the system. The architecture of the system is the set of structures needed to reason about the system, which comprise software elements, relations among them and properties.

3.1.1 Module 1

This section gives the architecture design for theme builder module

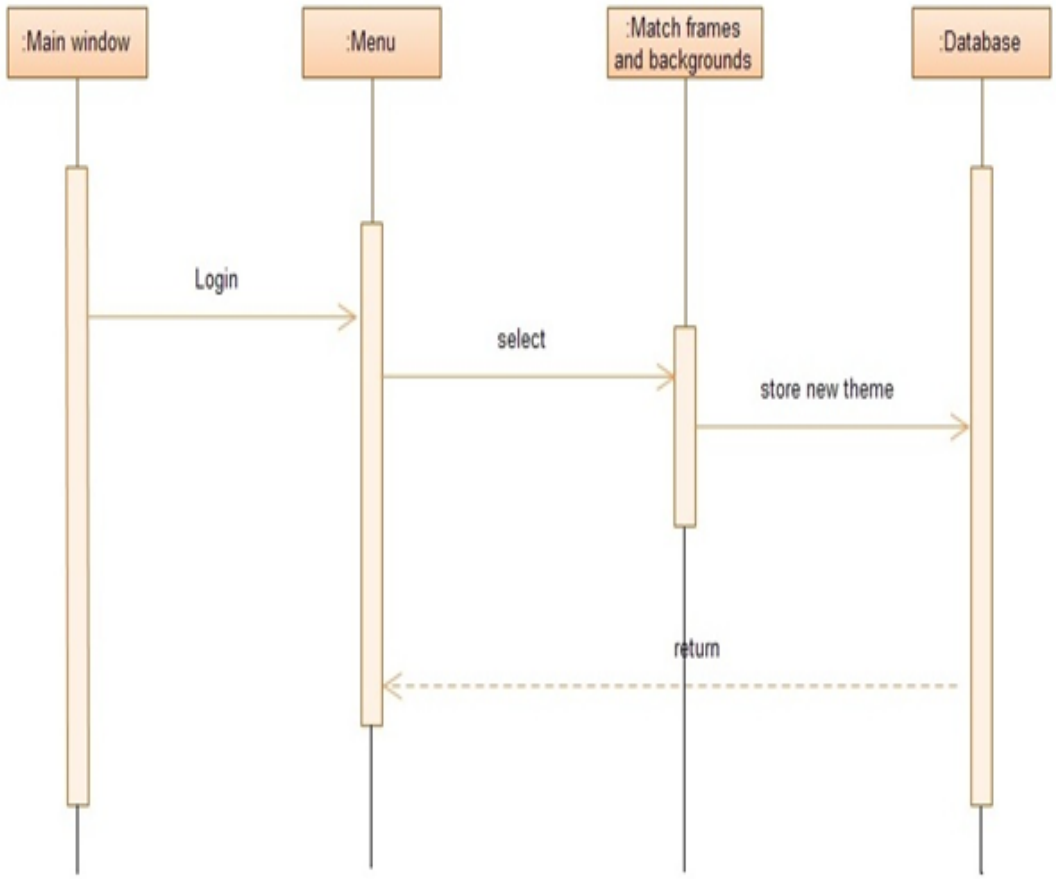


Figure 3.1: Architecture diagram for module 1[Theme Builder]

In this module a main window appears where the software developer is allowed to build new themes. First the developer logs in to the system by his username and password that provides him access to the database where the background and frames are stored. Then developer selects the backgrounds and frames of his interest and he is allowed to map the background with frame and new theme is generated and is stored back to the database.

3.1.2 Module 2

This section gives the architecture design for theme Uploader module.

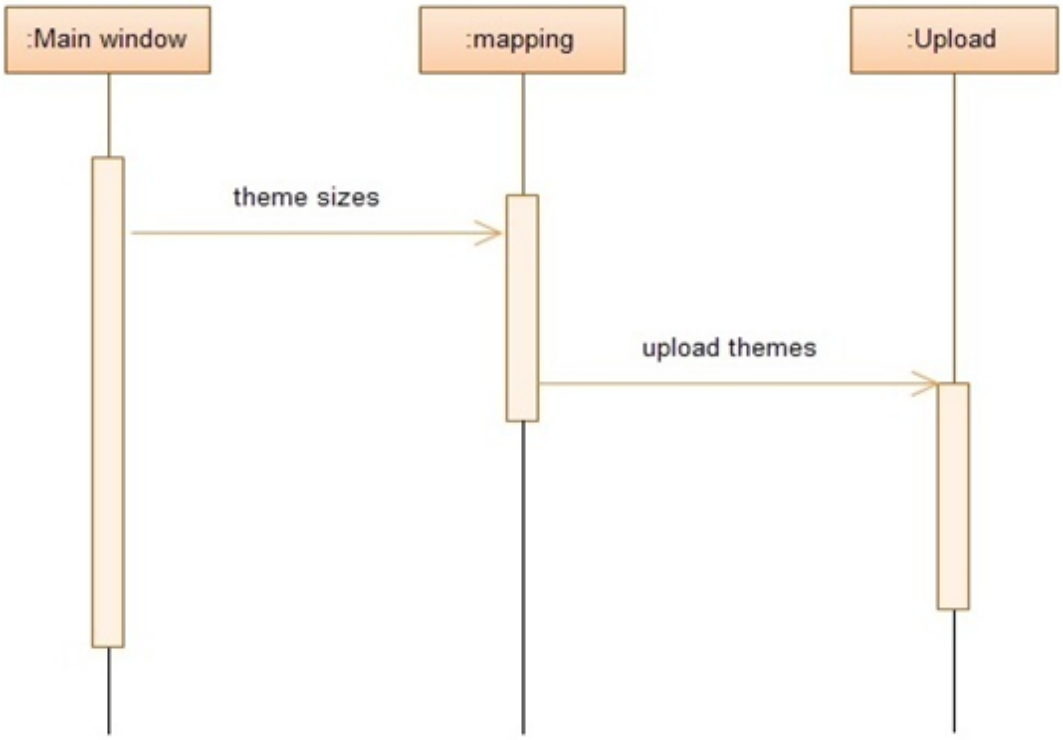


Figure 3.2: Architecture diagram for module 2[Theme Uploader]

In this module a main window appears where developer selects the theme which is generated in the previous module and then he maps the themes with the pre-defined sizes. After the mapping between the themes and size is done the developer uploads the themes to the server.

3.1.3 Module 3

This section gives the architecture design for functionality test module.

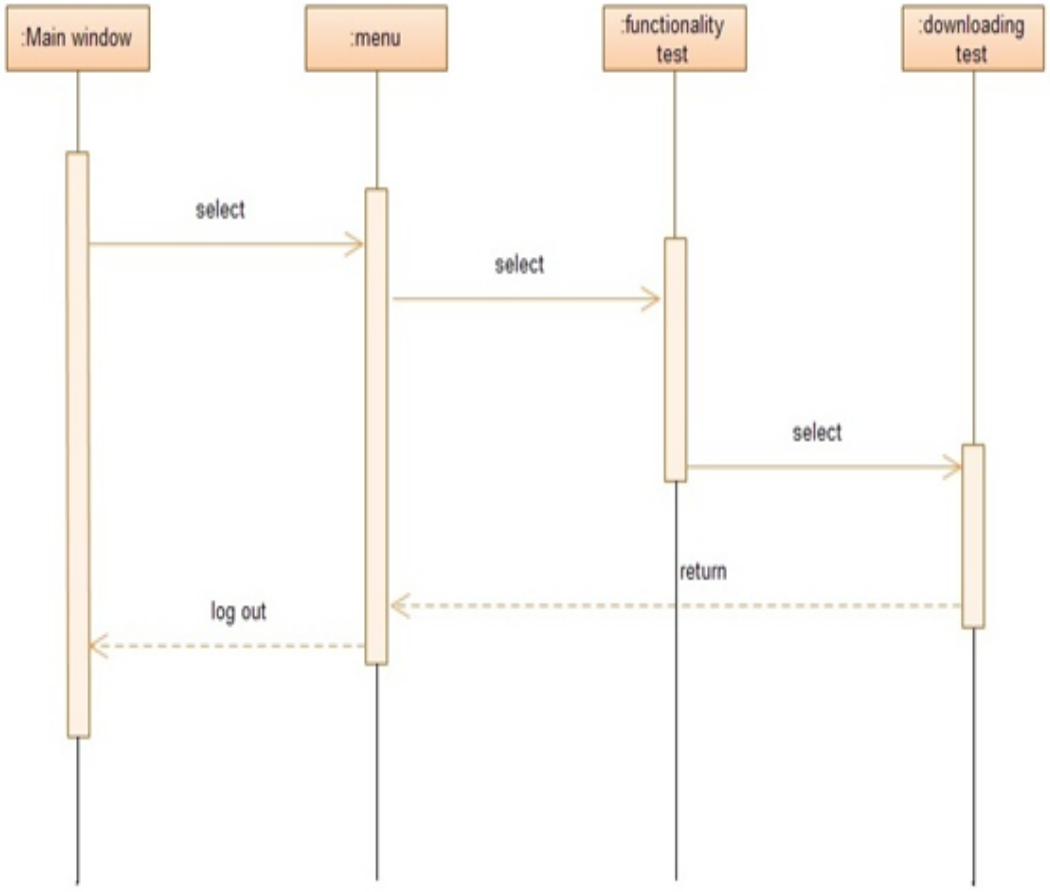


Figure 3.3: Architecture diagram for module 3[Functionality Test]

In this module a window appears where developer selects the theme according to his requirements and performs test on the functionalities of the package by giving different inputs and checks for the working of matching of all kinds of background, themes and size. Then developer checks for proper download of the theme package.

3.2 Alternative Architecture

This section gives the alternate architecture for the theme builder system.

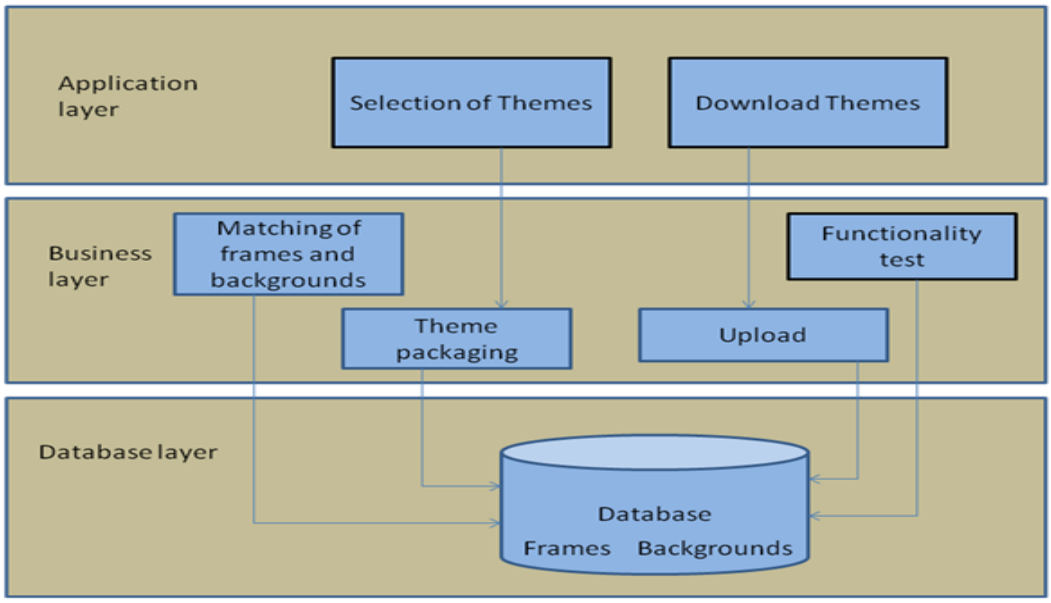


Figure 3.4: Architecture diagram for backend design [3-Tier Architecture]

The architecture provides the logical view of the theme builder system. In this the system is dived in to 3 layers as Database Layer, Business Layer and Application Layer. Database Layer has the dataset collected for the theme builder which includes the backgrounds and frames. These are accessed by the software developer while developing new theme. Business Layer includes the implementation of the functions of the theme builder. In this layer the developer fetches the background and frames and matches them to generate new themes. Then theme packaging is done where all the themes generated are mapped with various sizes and are packed to form theme packages and are uploaded to the server. Then developer checks for the proper download. The next layer is the application layer which is the interface between the user and the system where user selects the desired package and can install the theme package which he has selected.

3.3 Data Design

This section describes the persistent data, and/ or any other data important enough to justify a separate section. Data design of the theme builder is described in this section.

A blob (alternately known as a binary large object, basic large object, BLOB, or BLOb) is a collection of binary data stored as a single entity in a database management system. Blobs are typically images, audio or other multimedia objects, though sometimes binary executable code is stored as a blob. In theme builder the images can be stored either in blob or in simple file systems

Hadoop technology is used to store the themes built. Since the company has a large number of frames and backgrounds consequently the number of themes created is also large. To have an efficient processing of this data set hadoop is used.

- Map-Reduce :

  Map-Reduce is the programming model used for processing large data sets and importantly used with big data. It exploits data independence to do automatic distributed parallelism. The main task of the developer is to implement map and reduce functions. The input data is distributed in blocks to the participating machines using Google File System [GFS].

  When a job is launched, the system automatically spawns as many Map functions as there are data blocks to process. Each mapper reads the data iteratively as a key/value pair record, processes it, and if required, outputs key/value pair bound for a reduce function. All records with the same key go to the same reduce task

- Hadoop and HDFS

  Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters. It fosters two tasks: massive data storage and fast processing. Hadoop consists of a storage part – Hadoop Distributed File System [HDFS] and processing part Map-Reduce.

- HDFS Architecture

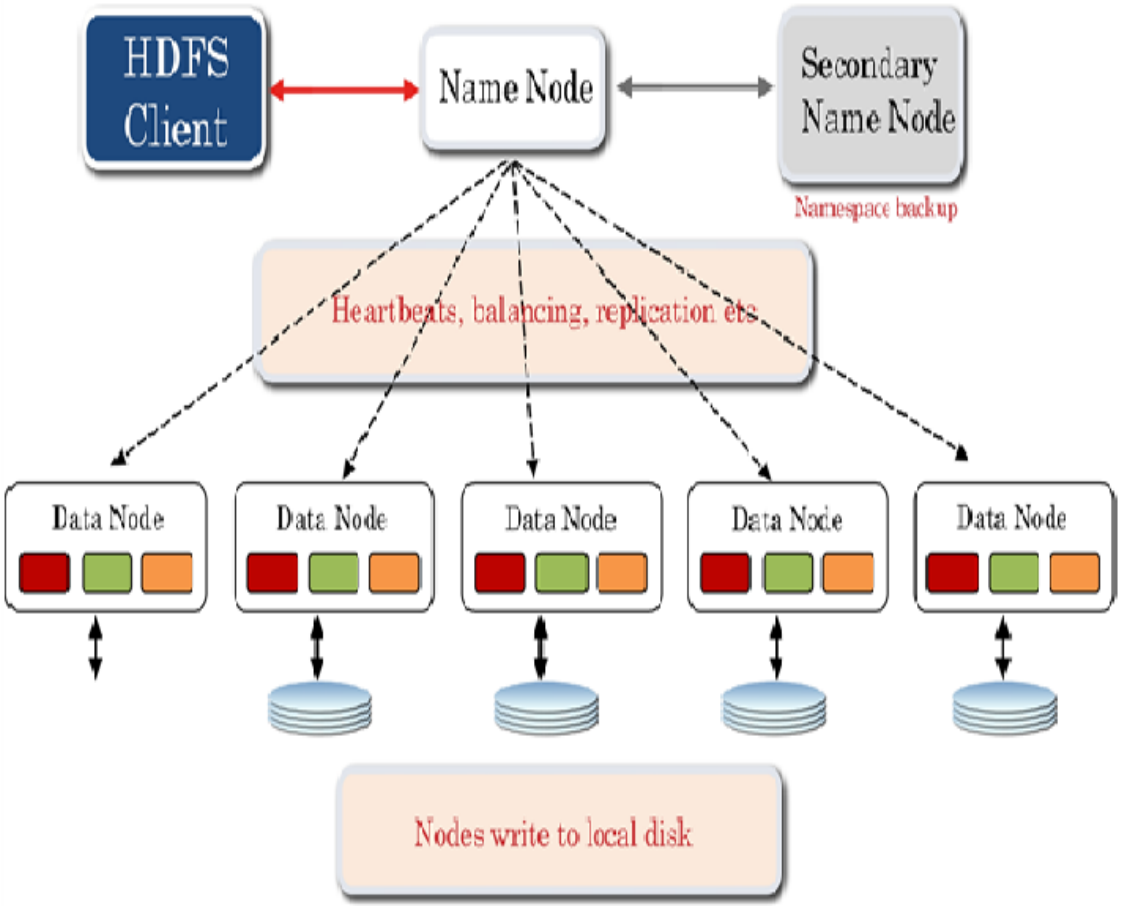


Figure 3.5: Structure of Hadoop Distributed File System

HDFS has a master slave architecture containing a single NameNode as the master and multiple DataNodes as workers. To store the data or a file in this architecture, the HDFS splits the file into fixed-sized blocks and stored them on workers. The mapping of blocks to DataNodes is determined by the NameNode. The NameNode also manages the file system's metadata and namespace. Image Colour based retrieval system using the Hadoop framework improves the efficiency of image storage and retrieval and provides better search results.

The proposed system of Image Retrieval based on Colour content of the image uses the Hadoop framework for storage and Map-Reduce for processing of the images.

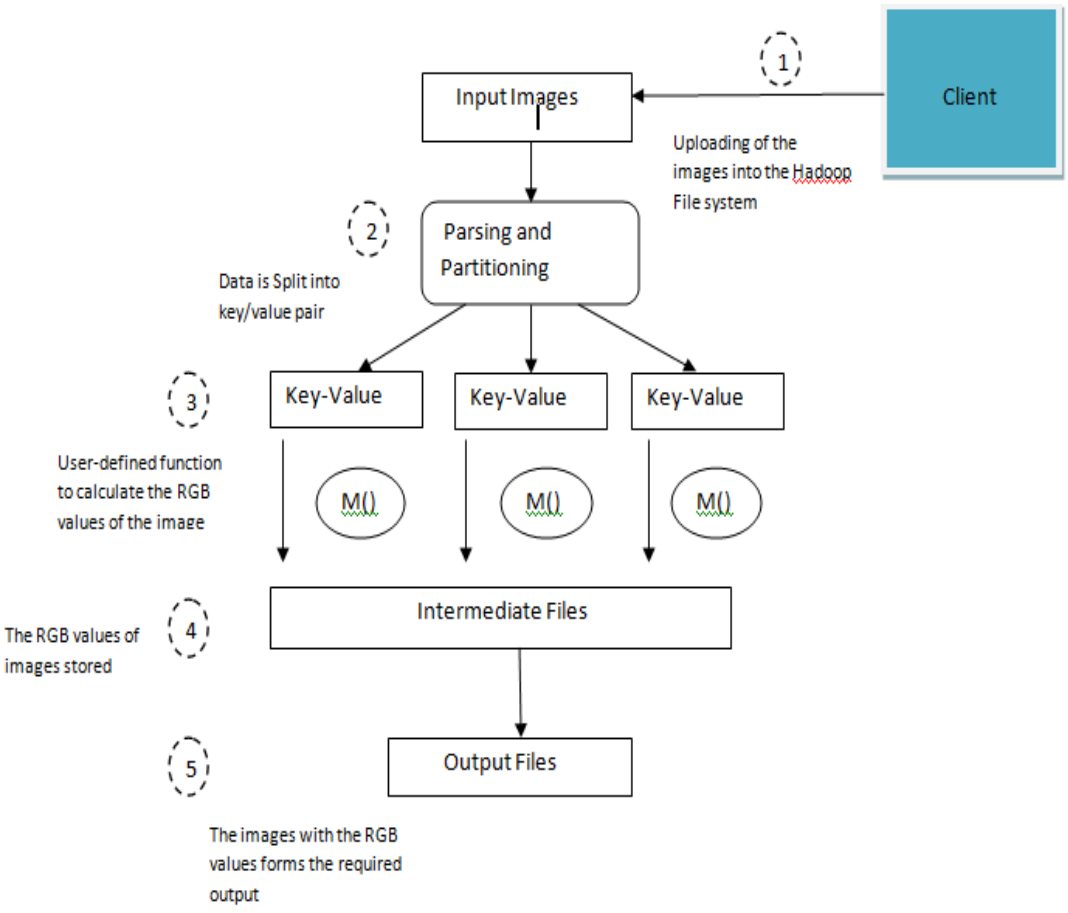


Figure 3.6: Map-Reduce Steps

The images from the client system are uploaded into the Hadoop system. The input consists of all the images uploaded into the file system by the user. Images form the data for the system. Further processing uses the CBIR method.

The Content Based Image Retrieval [CBIR] is used for retrieving the corresponding images from the database based on their features which are derived from the image itself like colour, texture and shape. Retrieval based on the contents of the image is more effective than the textual based retrieval especially for the image search. In the traditional approach the search for images is using tags, keywords etc. which are stored along with the images as a reference to the same.

The attribute used in the proposed system is Colour. Retrieval is done on the basis of similarity in colour. For each of the image in the database, colour histogram is computed which shows pixel position of each colour in the image. Colour Histogram is a simple method of counting the number of pixels for each 256 scales in each of the 3 RGB channel.

In general, a colour histogram is based on a certain colour space such as RGB or HSV. Here, we have used the RGB values for the colour histogram. The images and the corresponding RGB were stored in the intermediate files.

For each of the corresponding image which is fed into the system as the input by the user who wants to retrieve the images of the colour similar to the input image, the RGB value of the image is calculated and the Euclidean distance of the RGB values between the input image and the images stored in the database is calculated and images are ranked in the increasing order of the Euclidean distance.

3.4 User Interface Design

This section deals with the design of the interface part that is used in theme builder application.

3.4.1 Input Window - Admin Module

Admin module has the input window which has the backgrounds displayed in it. Three options are provided to the admin as next, creeate and add.

On Next button admin is taken to next set of backgrounds on create user is allowed to create the themes and on the add button admin is alllowed to add the image which is required by him for the background.

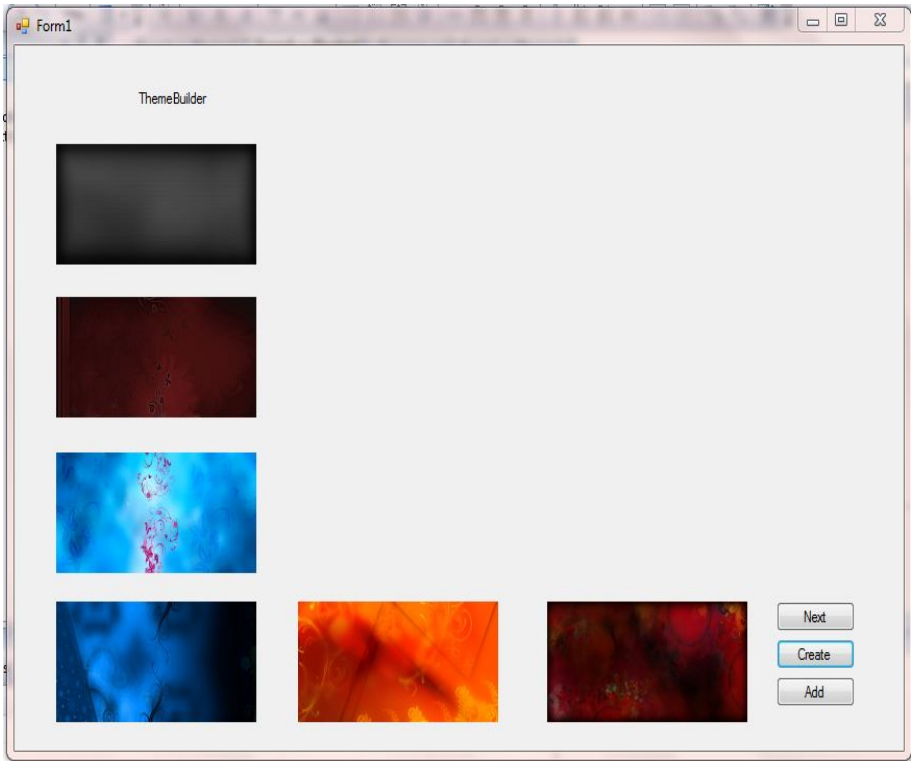


Figure 3.7: Input window for the admin module

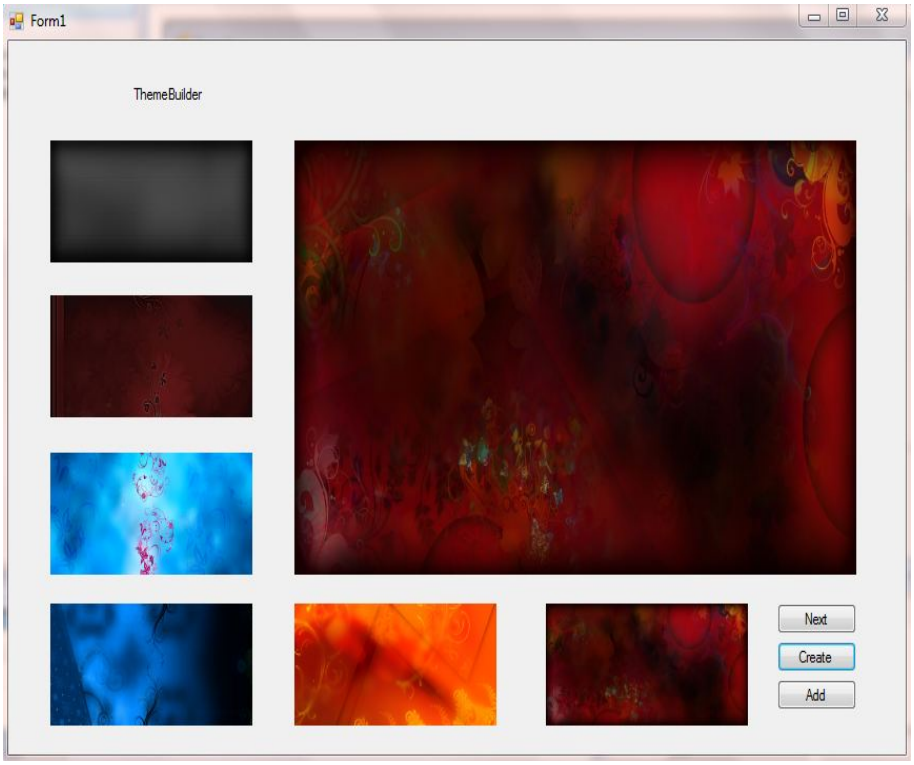


Figure 3.8: Selected image

On the create option admin is allowed to create the themes. The following interface is displayed to the admin. The options are provided for the admin to create the themes with 2,3,4 and 5 frames is provided. Admin selects the option based on his requirements.

On selecting 2 frames the following output window is displayed.

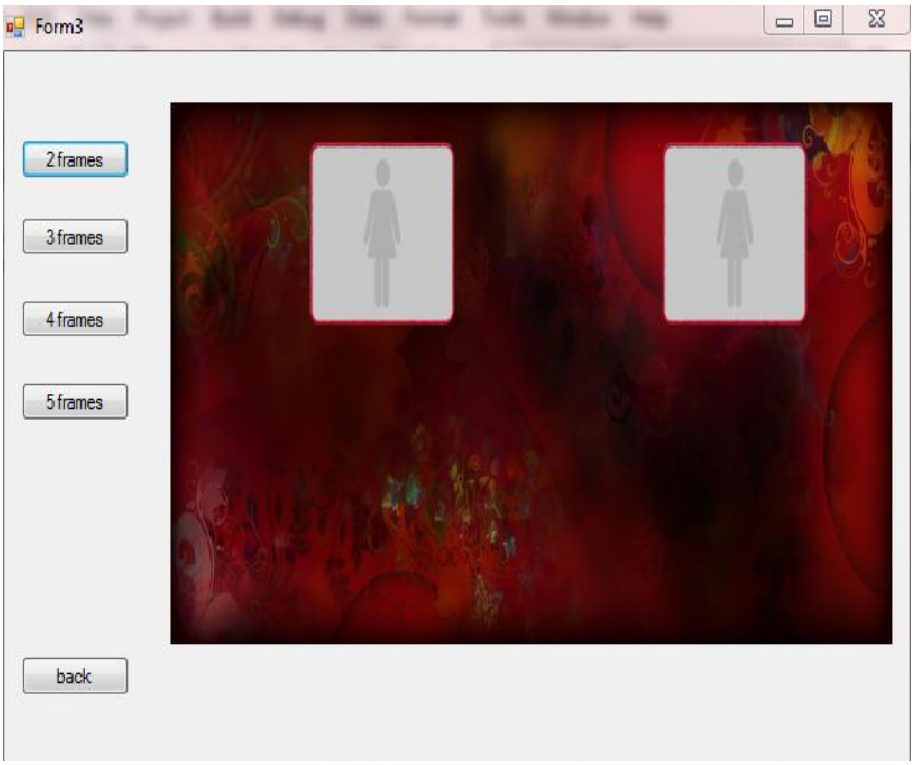


Figure 3.9: Output window for 2 frames

On selecting 3 frames the following output window is displayed

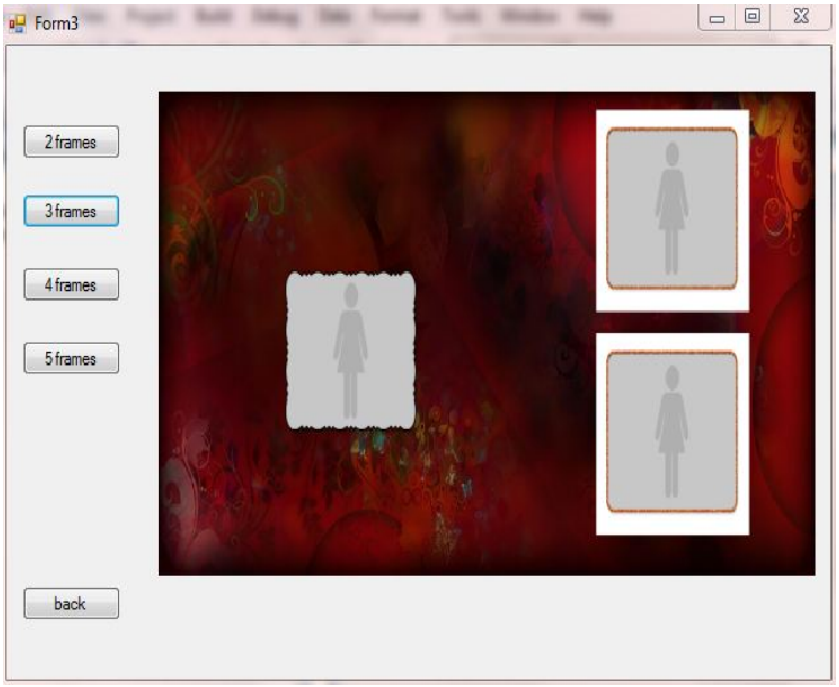


Figure 3.10: Output window for 3 frames

On selecting 4 frames the following output window is displayed.

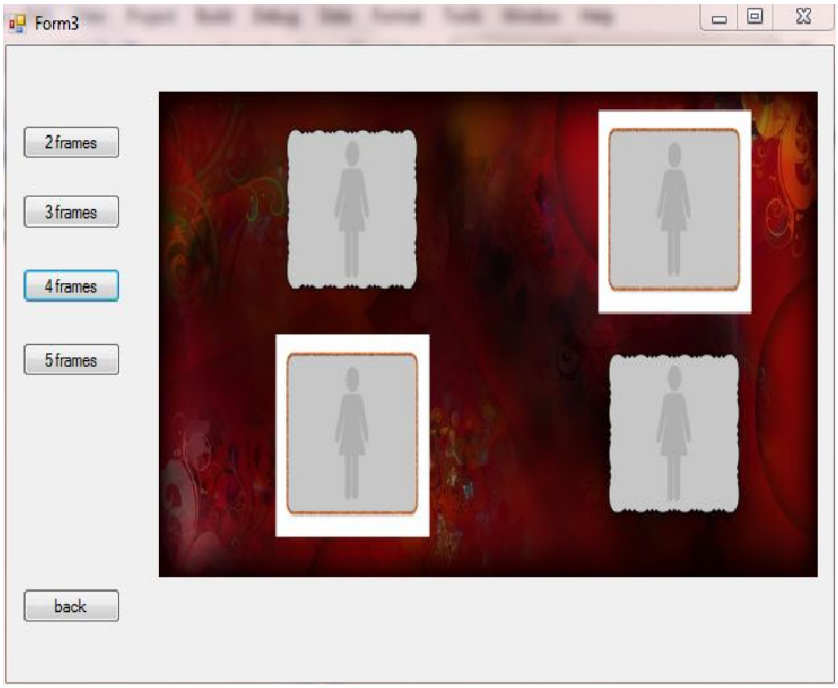


Figure 3.11: Output window for 2 frames

3.4.2 Input Window – User Module

The input window consists of the dialogue box. The user is given the option to choose an image of his choice that will serve as the basis for the image retrieval from the hadoop system.

The input image given by the user is used to calculate the Euclidean distance between the images store and the input image to retrieve the images that are similar to the input image.

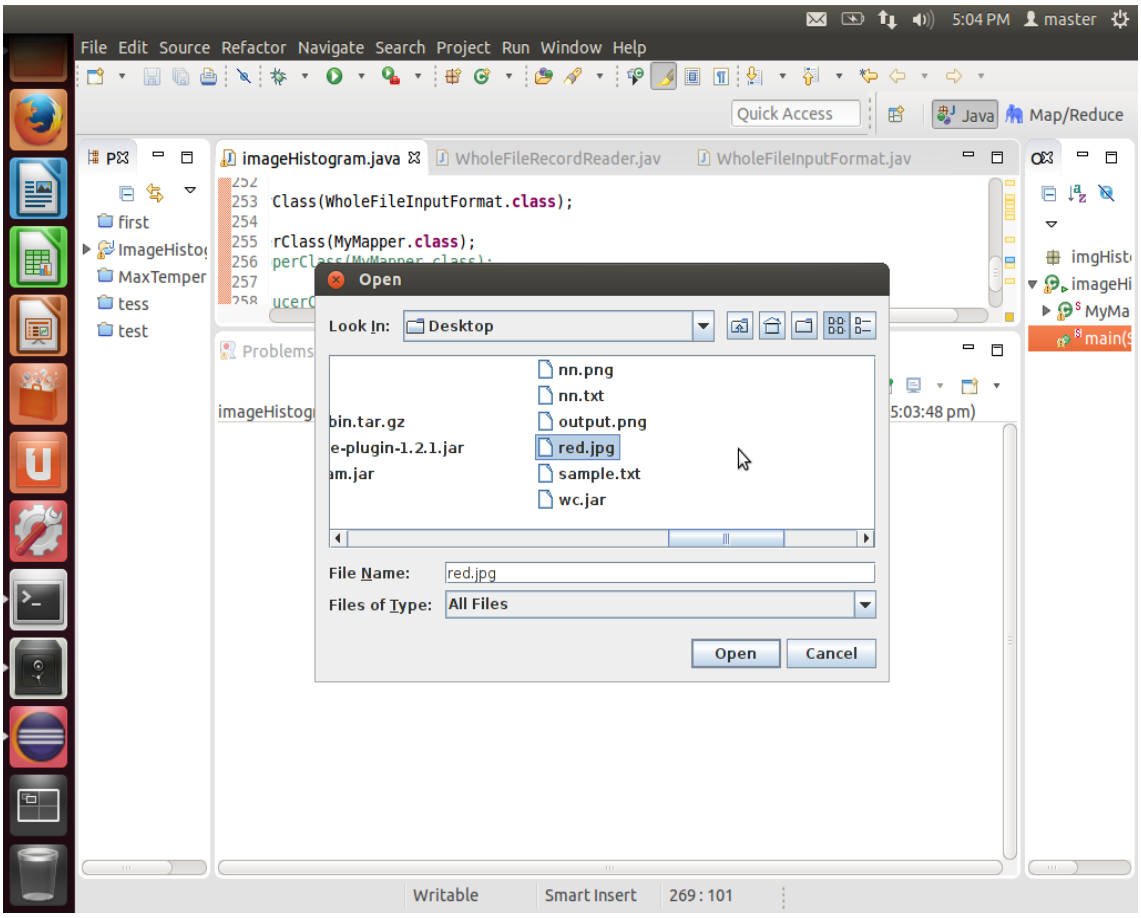


Figure 3.12: The user interface to select the image of the choice

3.4.3 Output Window – User Module

The output window shows the retrieved images after the calculation of the Euclidean distance between the input image and the images stored in the hadoop system. The retrieved images are those images whose Euclidean distance values are the least indicating their nearness to the input image.

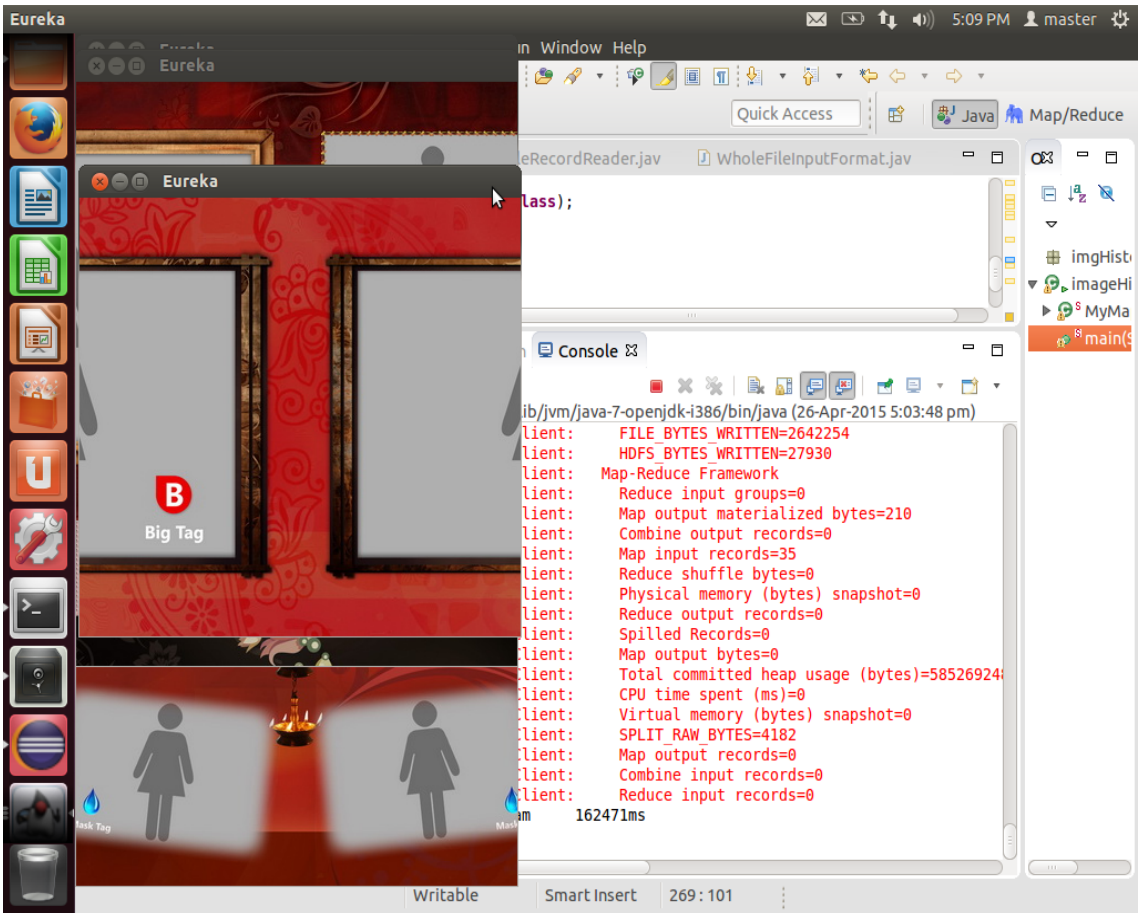


Figure 3.13: shows the user interface of the retrieved images

Chapter 4

IMPLEMENTATION

This chapter gives a brief description about implementation details of the system by describing each component with its code skeleton

4.1 Model 1 : Theme Builder

Different techniques can be used to store and retrieve images from database or file systems. One such technique we implemented is to use simple file systems.

Code snippet:
Displaying the backgrounds:

```
public Form1()
{
InitializeComponent();

pictureBox1.Image=Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\The
me_Builder\\B&W Stroke\\G_2.jpg");
pictureBox2.Image =
Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\Theme_Builder\\Rich
Mix\\RBFB_1.jpg");
pictureBox3.Image =
Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\Theme_Builder\\Rich
Mix\\RBFB_2.jpg");
pictureBox4.Image =
Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\Theme_Builder\\Rich
Mix\\RBFB_3.jpg");
pictureBox5.Image =
Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\Theme_Builder\\Rich
Mix\\RBFB_4.jpg");
pictureBox6.Image =
Image.FromFile("C:\\Users\\Asus\\Desktop\\Online_Themes\\Theme_Builder\\Rich
Mix\\RBFB_5.jpg");
}
```

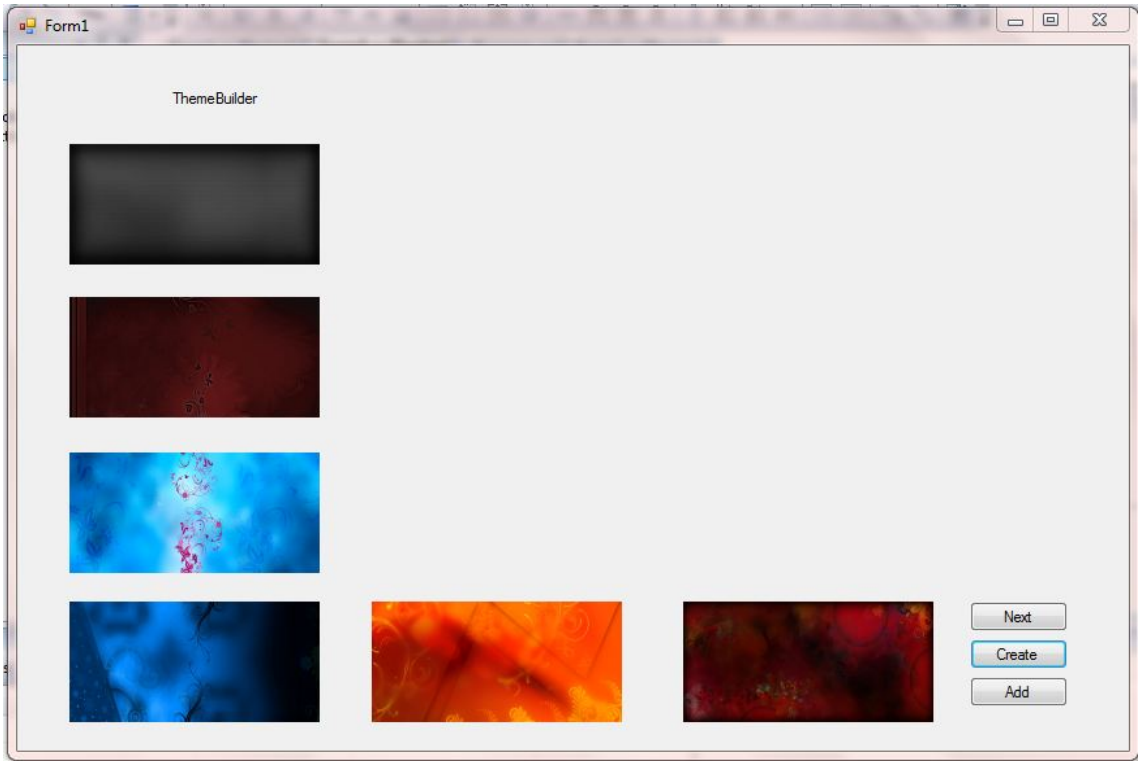


Figure 4.1: Displaying of backgrounds

4.2 Module 2: Selection of themes

As the application is user friendly, the user can select the background color of interest to which a set of themes will be retrieved from hadoop database.

```
private void button1_Click(object sender, EventArgs e)
    {
        System.Drawing.Image canvas =
        Bitmap.FromFile("C:\\Users\\Asus\\Desktop\\themes\\new.jpg");
        Graphics gra = Graphics.FromImage(canvas);
        Bitmap smallImg = new
Bitmap(@"C:\Users\Asus\Desktop\Online_Themes\Theme_Builder\Rich
        Mix\Frame\H005.png");
        Bitmap secSmallImg = new
        Bitmap(@"C:\Users\Asus\Desktop\Online_Themes\Theme_Builder\Rich
Mix\Frame\H005.png");
        gra.DrawImage(smallImg, new Point(800, 70));
        gra.DrawImage(secSmallImg, new Point(3000, 70));
        canvas.Save(@"C:\Users\Asus\Desktop\pojo\image1.jpg",
        System.Drawing.Imaging.ImageFormat.Jpeg);
        pictureBox1.Image = Image.FromFile(@"C:\Users\Asus\Desktop\pojo\image1.jpg");
    }
```

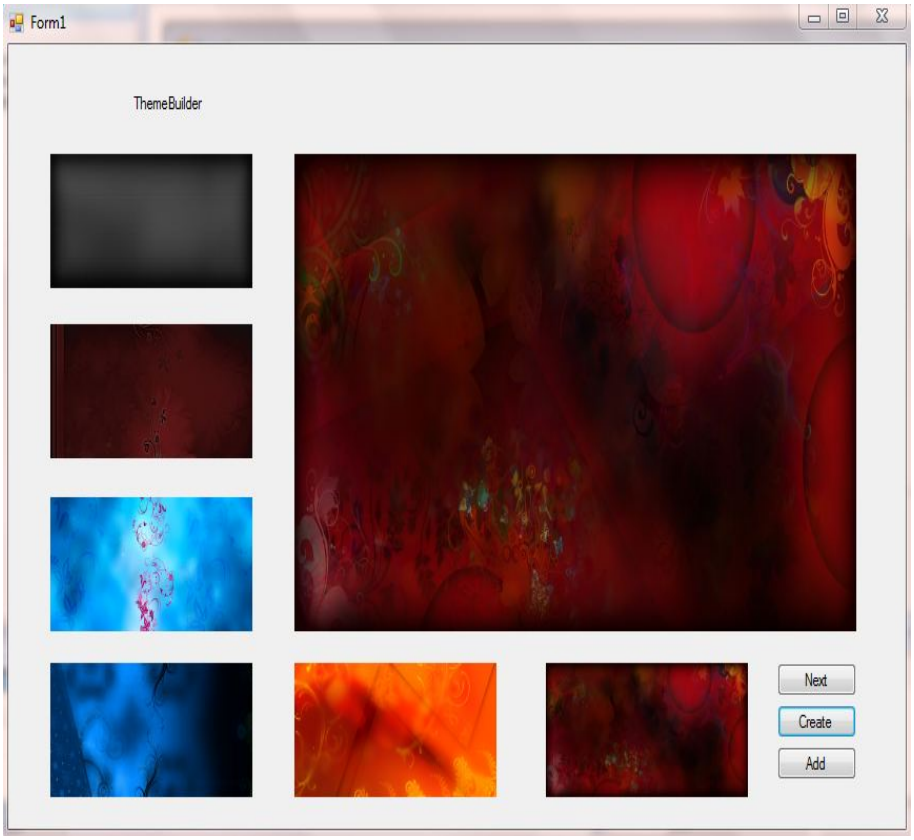


Figure 4.2: Selecting background to create themes

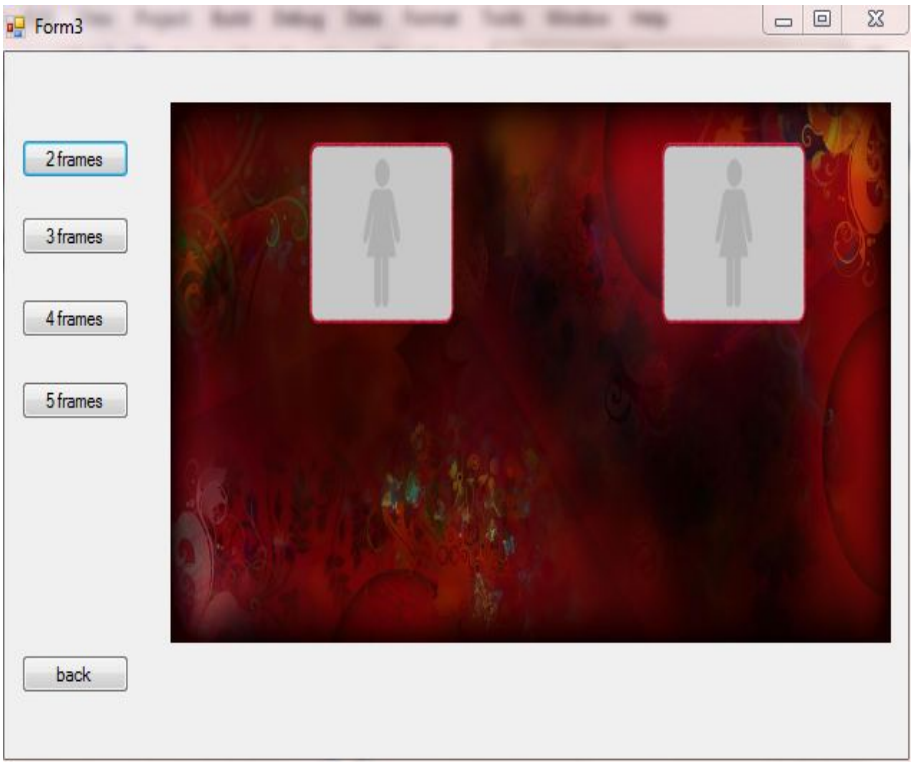


Figure 4.3: Theme having 2 frames

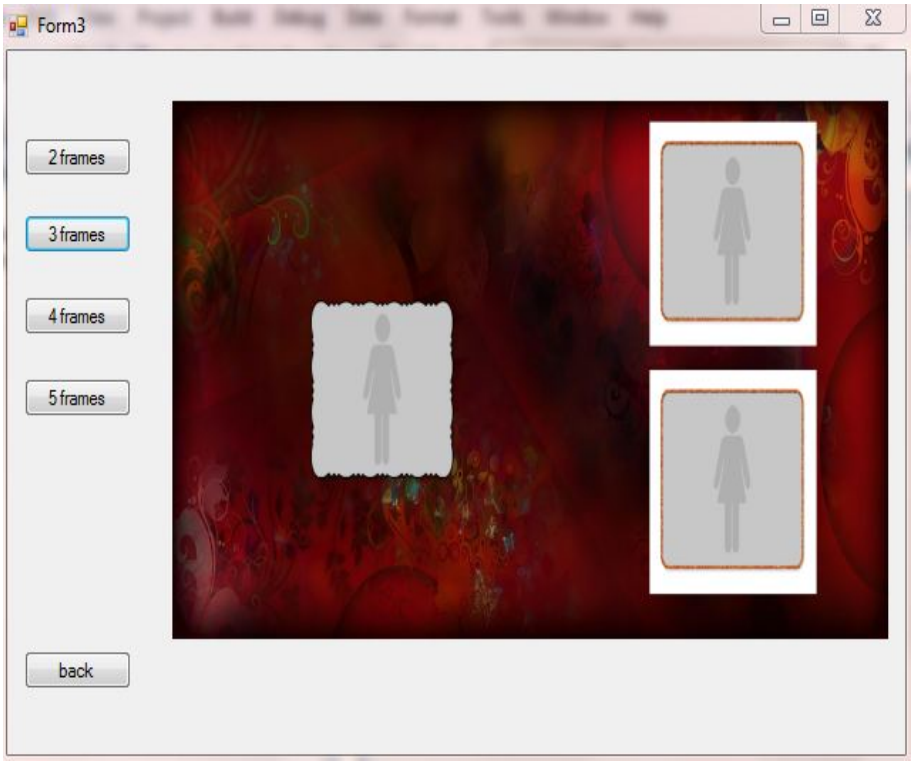


Figure 4.4: Theme having 3 frames

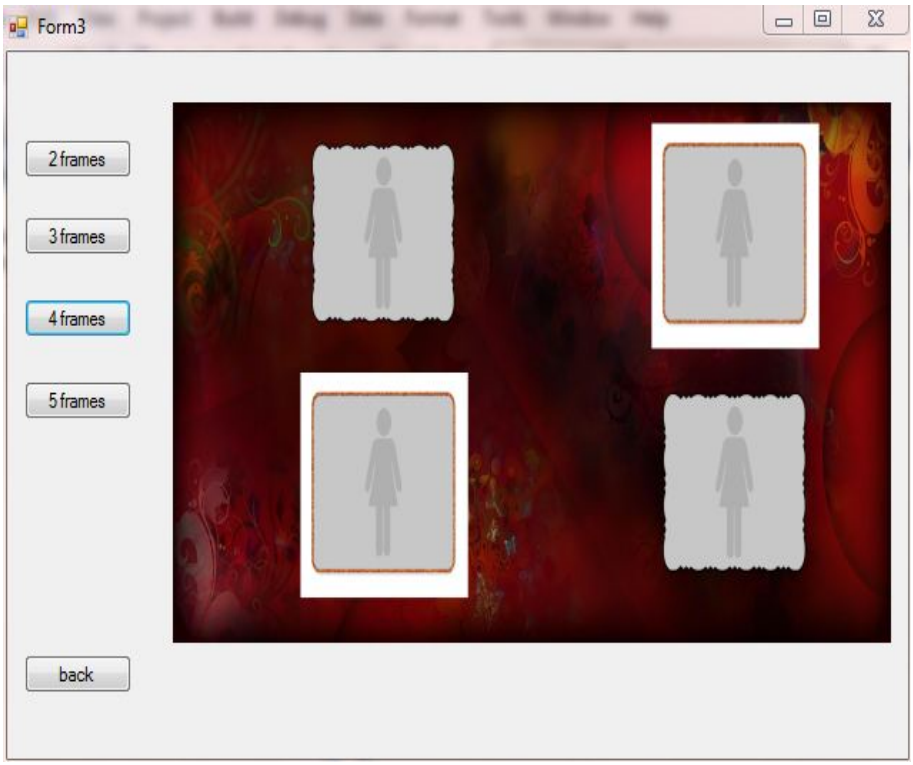


Figure 4.5: Theme having 4 frames

4.3 Module 3: Colour based image retrieval

The themes built are stored in the Hadoop system. The color based image retrieval is performed using the calculation of the RGB values of the images. The RGB values of all the images in the Hadoop file system are calculated. The input image is fed by the user using the prompt. The RGB value of the input image is also calculated and the Euclidean distance between the input image and all the images stored in the database is calculated. The images with the least Euclidean distance value are retrieved which are similar in color as the input image.

4.3.1 Code Snippet

Calculation of the RGB values:

```
int w = image.getWidth();
int h = image.getHeight();
for (int i = 0; i < h; i++)
for (int j = 0; j < w; j++)
int alpha = (pixel » 24)  0xff;
int red = (pixel » 16)  0xff;
int green = (pixel » 8)  0xff;
int blue = (pixel)  0xff;
```

Calculating the Euclidean distance and output:

```
double dist = Math.sqrt((Math.pow(r-r1,2) +
Math.pow(g-g1,2) + Math.pow(b-b1,2)));
Configuration conf = new Configuration;
Job job = new Job(conf, "Whole file input format");
job.setInputFormatClass(WholeFileInputFormat.class);
job.setJarByClass(WholeFileInputFormat.class);
job.setMapperClass(MyMapper.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new
Path("hdfs://localhost:54310/user/hduser/jpgdata"));
FileOutputFormat.setOutputPath(job, new
Path("hdfs://localhost:54310/user/hduser/pgmoutdata2"));
boolean sucess=job.waitForCompletion(true);
```

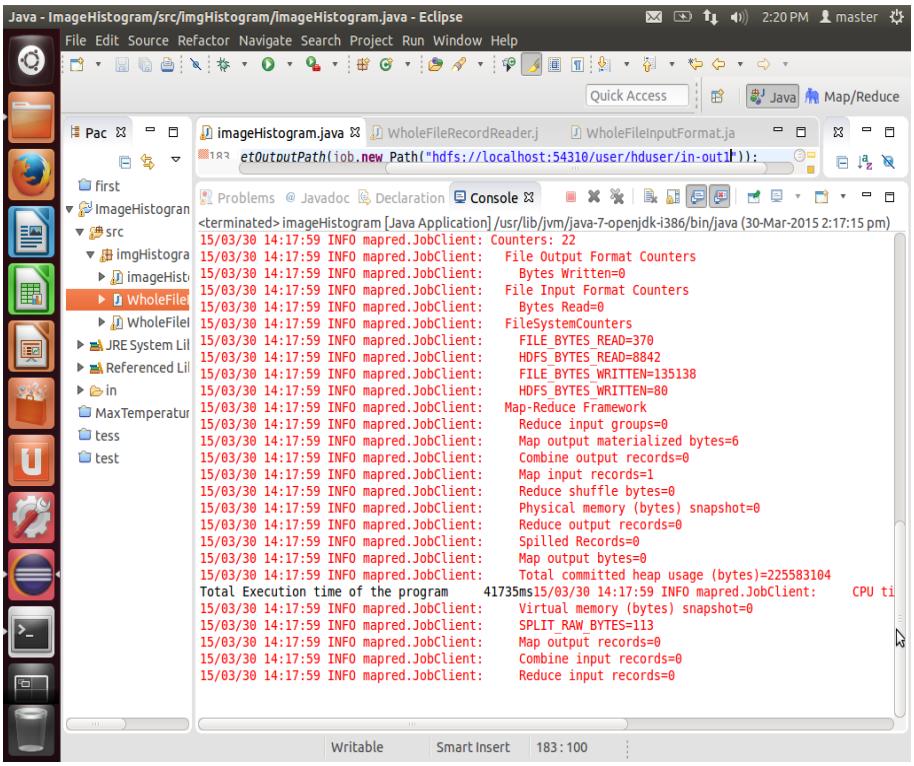


Figure 4.6: Running of Map-Reduce program

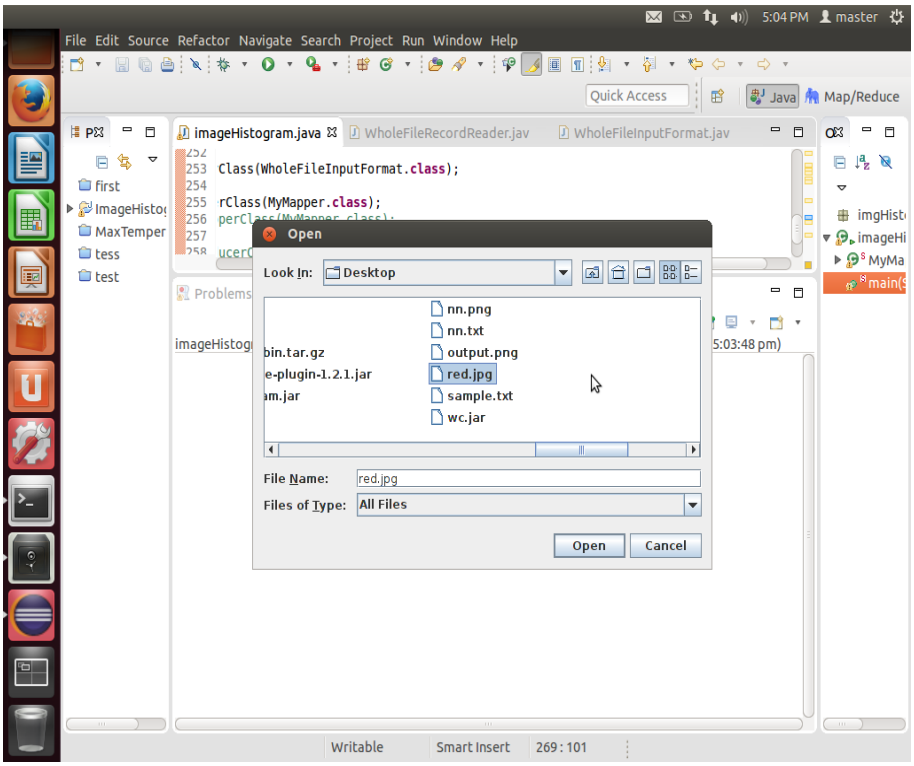


Figure 4.7: User Selection of input image

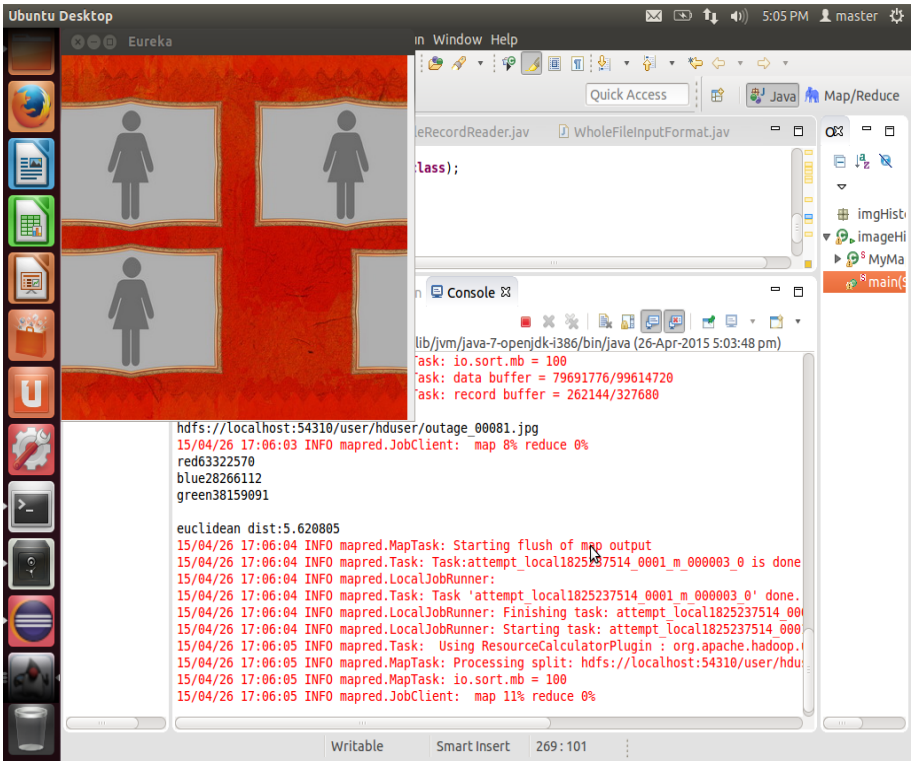


Figure 4.8: Calculation of RGB values and Euclidean distance and image retrieval

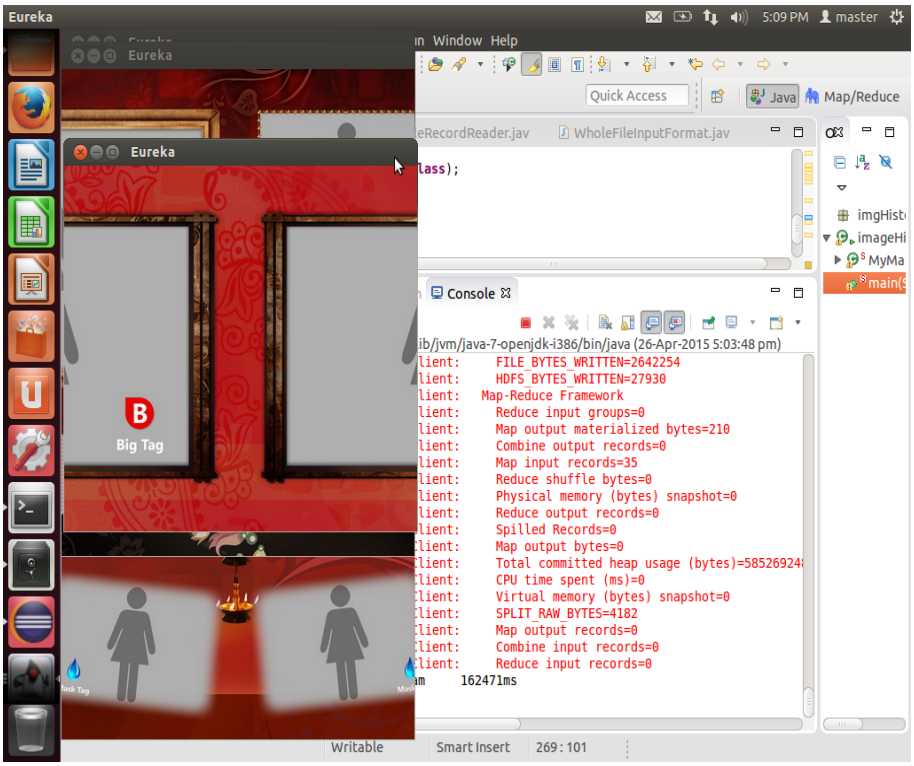


Figure 4.9: Images retrieved

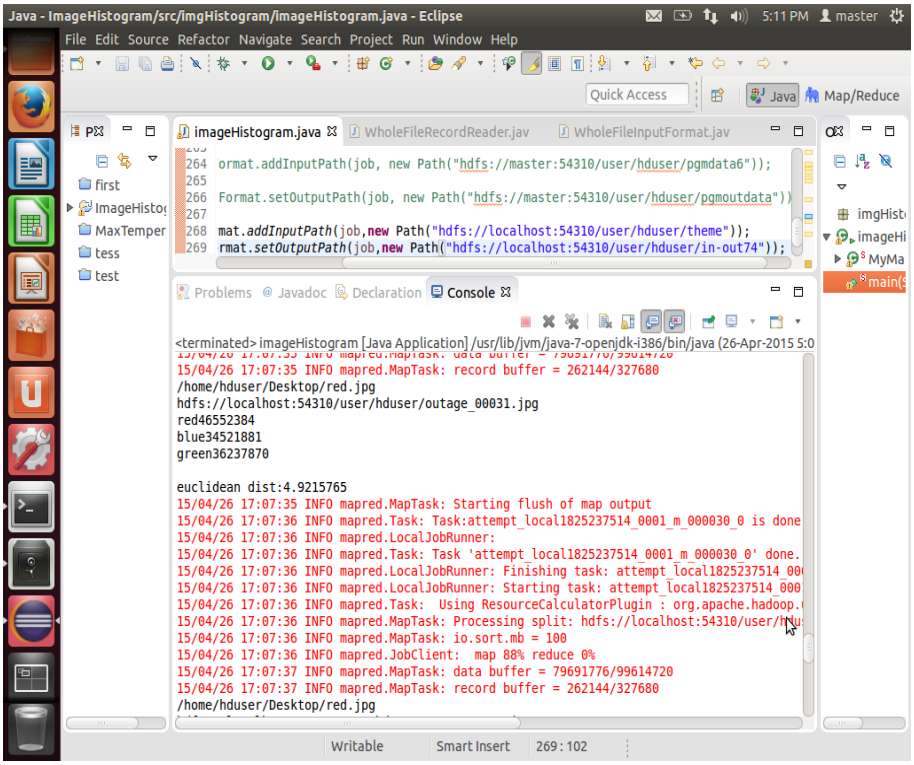


Figure 4.10: Calculation of RGB value and Euclidean distance for each image

Chapter 5

TESTING

This document explains the testing methodology for the theme builder application. It lists out all the testing items and the procedures conducted on those items. It also describes testing approaches followed by the test case specifications. This section provides the overview of entire test document. This document describes both the test plan and the test procedures. It includes the test items which may be a single program, module, function or class that need to be tested. It identifies all the software features that need to be tested. It describes the different testing approaches followed along with the methods and criteria to be used in performing test activities for the project. The rest of this document is organized as follows; it includes the test items, identifies all the software features that need to be tested, describes different testing approaches followed along with the methods and criteria to be used in performing test activities and specifies the conditions that need to be met to approve test results for each item. It gives the test execution report with actual results.

5.1 Test Items

The various units of the system that will be tested are:

1. The number of images that can be processed
   - The input is given to the system
   - Hadoop system is tested for various input sizes
   - Execution time is measured
2. The precision/recall of the retrieved images
   - The images retrieved are checked for their precision and recall

5.2 Test features

The features to be tested are:

- Whether the system can handle the images upto 1 GB and more

- Whether the system presents a good precision and recall rates
- How much time it takes for the execution and whether it is less than the execution time of the present system in place.

5.3 Testing Approaches

This section describes the overall approaches and strategies used in testing. For testing the system black box testing approach is used. Specification based testing is used to test the functionality of the application which includes testing:

- The execution time of the programs
- The precision / recall of the retrieved images
- The number of images the system can process and in what time.

5.4 Unit Testing

The unit testing was done for the Hadoop system with the image set up to 1GB and the results were obtained as follows

The execution time vs. No. of images	Use case number: UC1
Module 1 Overview The system is designed to calculate the RGB values of all the images stored in the hadoop system and also the Euclidean distance between the input image and images stored in Hadoop. The testing is done to check as to what time it takes for Processing the images and testing is done up-to 1Gb of data Input to the module The images stored in hadoop and also the user input image Output of the module The retrieved images, RGB values, Euclidean distance and execution Time.	

No. of images	Execution time in milliseconds
35	188695
230	1249553
250	1358210
3717	21190715

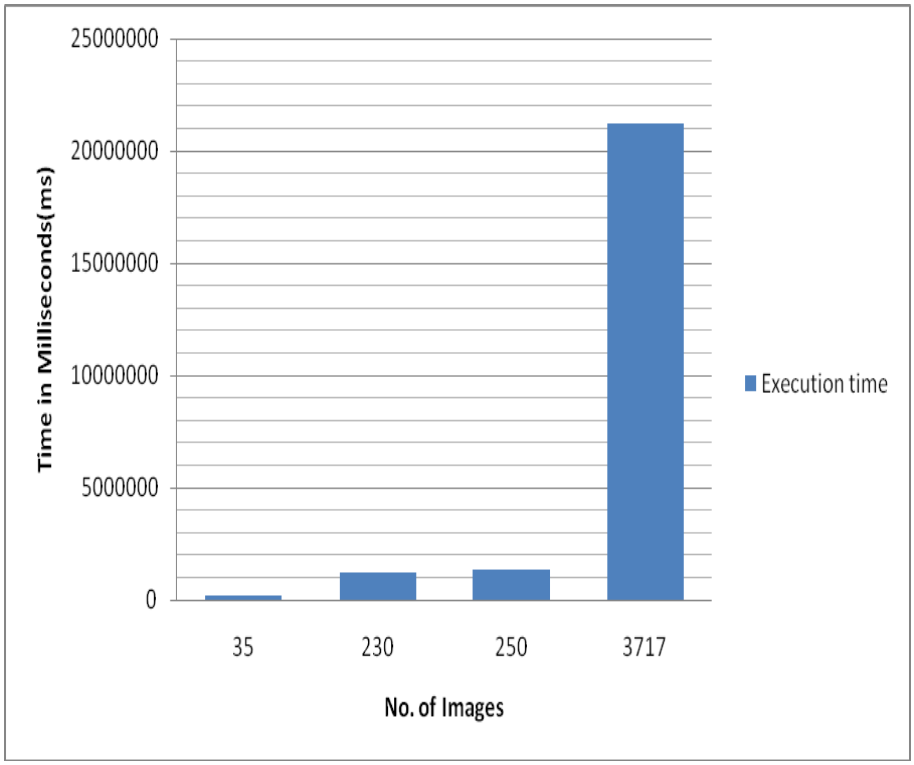
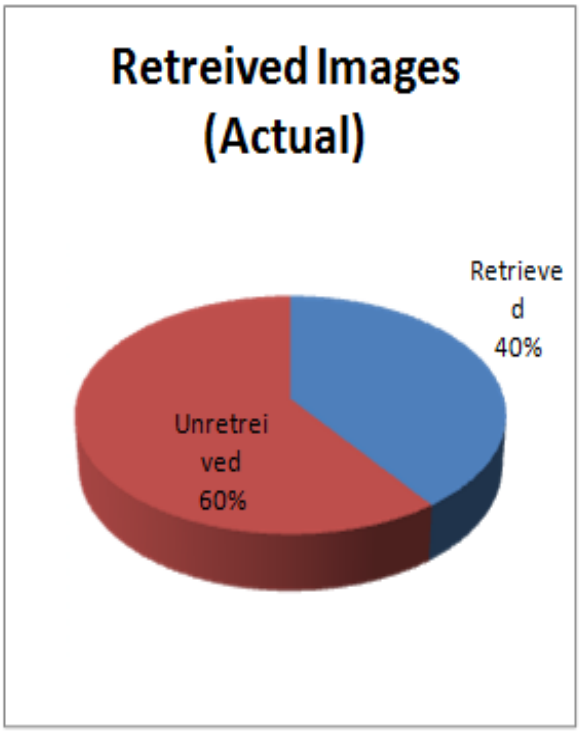


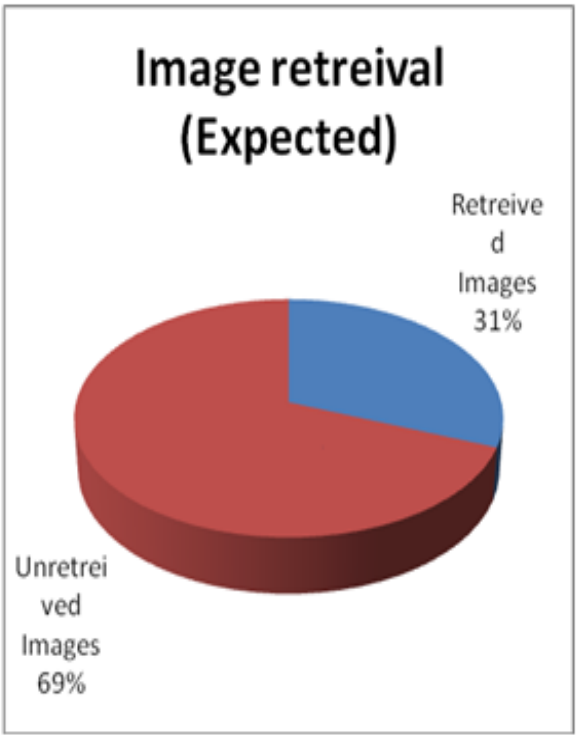
Figure 5.1: No. of images vs. Time in milliseconds

5.5 Integration Testing

Testing was also done for the relevance feedback to check what percentage of the correct images were retrieved and results obtained were as follows

Testing which was done for 35 images initially where the expected number of retrieval were 11 but the number of retrieved images are found to 13.





Chapter 6

CONCLUSION and FUTURE SCOPE

6.1 Conclusion

The storage of images in database is not favored as the amount of space it consumes and the processing time that it takes is huge. Thus, most appreciated solution for storage of images and that too in large numbers so that processing time is minimized, is Hadoop. The other important aspect considered is the retrieval with use of tag words is not well suited in case of images; much better results can be obtained using other features of images like color, texture and shape. Thus the project intends to provide such a solution using the Hadoop system and CBIR method for image retrieval

6.2 Future Scope

The images were stored on Hadoop system for processing, studies show that the other big data database structures like Hbase, Pig can be used instead of normal Hadoop. Thus, the work can be scaled to an Hbase system as well. Also, the system built only the color property of image for retrieval of images but the other features like shape and texture can also be considered for better results.

Chapter 7

REFERENCES

[1] Albumizer: www.albumizer.in [dated: 25-08-2014].

[2] Kai Hwang, Geoffrey C. Fox, Jack J. Dongarra, "Distributed and Cloud Computing from Parallel Processing to the Things", Morgan Kaufman, Elsevier- 2012.

[3] www.thinkbiganalytics.com/hadoop-sequence-files-and-a-use-case/ [dated: 19/11/2014].

[4] wiki.apache.org/hadoop/SequenceFile [dated: 19/11/2014]

[5] www.databasejournal.com/features/mysql/articla.php/3719221/storing-Images-and-BLOB-files-in-SQL-Server.html[dated 22/11/2014]

[6] Jing Zhang, Xianglong Liu, Junwu Luo, Bo Lang. DIRS: Distributed Image Retrieval System Based on MapReduce. 2010 IEEE.

[7] Banaei, S.M. and Moghaddam, H.K. (2014) Hadoop and Its Role in Modern Image Processing. Open Journal of Marine Science, 4, 239-245. http://dx.doi.org/10.4236/ojms.2014.44022

[8] Michael, K. and Miller, K.W. (2013) Big Data: New Opportunities and New Challenges. Journal of IEEE Computer Society, 46, 22-24.

[9] Denis Shestakov, Diana Moise (2015) Scalable high-dimensional indexing and searching with Hadoop.

[10]IbrahimAbakerTargioHashem, IbrarYaqoob, NorBadrulAnuar, Salimah, - Mokhtar, AbdullahGani and SameeUllahKhan (2015) The rise of "big data" on cloud computing: Review and open research issues, Information Systems 47 (2015)98–115

[11] Junaid Khan and Prof. S.S. Kulkarni, Implementation of Content Based Image Retrieval Using Clustering technique. IJCSMC, Vol. 3, Issue. 5, May 2014, pg.523 – 529