The Australian National University
Faculty of Engineering and Information Technology
Department of Engineering

Undergraduate Thesis

Interactive Mobile Interface for Alternate Video Browsing

By: Anshul Saini (u4545909)

Supervisor: Dr. Lexing Xie

Study Programme: Bachelor of Engineering
Field of Study: Multimedia Understanding and Retrieval
November 2nd, 2012

[THIS PAGE HAS BEEN INTENTIONALLY LEFT BLANK]

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university. To the best of the author's knowledge, it contains no material previously published or written by another person, except where due reference is made in the text.

Anshul Saini
November 2nd, 2012

[THIS PAGE HAS BEEN INTENTIONALLY LEFT BLANK]

Acknowledgement

As an undergraduate student, I am deeply grateful to Australian National University for the various opportunities that I have been presented with; the people that I have met, and the education that I have received, over the years.

Among the people that I have met, is my thesis supervisor, Dr. Lexing Xie. Last year, I was browsing through the list of potential supervisors for my proposed research topic, when I approached Dr. Lexing Xie. Although, my topic had changed, I wanted to work under her.

With little background in server-side programming, I was shown immense support and guidance through regular weekly meetings that I had with her. Provided her background in multimedia, social media and in general, Human Computer Interaction, I have learnt a lot from this project (Python Language, Server-Side Programming, the User Interface and Project Management), and am humbled by the opportunity of working under her.

Honglin Yu, a PhD candidate under Dr. Lexing Xie, has also provided me with the support and technical assistance that I needed over the duration of the thesis project.

Last but not least, I would like to thank my parents and friends, for their encouraging words and support, in all my endeavours. I really appreciate your support.

Abstract

The thesis outlines the implementation of an Android mobile application that allows users to view and browse videos for the available categories, as well as find matching videos to a selected video, on the basis of frame-matching or 'visual meme tracking'[1].
The goal is to utilise the available 'visual meme tracking' tools and provide a new platform for the users to engage through smartphones.
Provided these tools, the report outlines the progression of the mobile application from the concept stage, into a prototype, that is then transformed into the application's User Interface. Alongside, the database on the server-side for storage and retrieval of data, is designed and developed. Finally the application is connected to the server in the backend, and thoroughly tested to provide the complete system.
Throughout the entire process of mobile application development, various processes and standards have been used to streamline the delivery of the application, while ensuring any changes in the requirements or in the scope of the project are reflected across appropriately. These have been categorised and documented into relevant sections, following the standard principles of software engineering.

[1] Extraction of similar features across multiple frames of various videos, through image matching techniques such as color-correlogram etc

Assumed Knowledge

Although, best efforts have been made to explain any introduced concepts, a basic understanding in the below areas, has been assumed, as an in-depth description in the following areas is beyond the scope of the project:

Class-diagrams - It is assumed that the reader is able to interpret and understand the various relationships between classes presented on a class diagram.

Project Management – It is assumed that the reader understands the basic concepts of project management, and is able to read and interpret figures and values from Work Break-down Structure and Gantt chart.

List of Figures

List of Tables

Glossary of Terms

Terms	Description
Android	A linux-based operating system developed by Google Inc. designed for touchscreen mobile devices such as smartphones and tablets.
Android Gallery	Viewgroup for displaying a list of horizontally scrollable items in Android.
Android ListView	Viewgroup for displaying a list of vertically scrollable items in Android.
Angry Birds	A puzzle video game developed by Rovio Entertainment.
ASCII	American Standard Code for Information Interchange (ASCII), a character-encoding scheme for representing text in computers.
Backend	Another name for servers.
Balsamiq	Software used for designing User Interface prototypes.
Base64	An encoding scheme that represents binary data in ASCII format through the use of radix-64 representation.
Binary	A numbering system consisting of two digits: 0 and 1.
Black box	A system or device with the knowledge of only it's inputs and outputs and none of its internal workings.
Bug	A error, fault or flaw in a computer program
Cantabile	Name of server provided by ANU
Colour-correlogram	Technique used for indexing and comparison of video frames.
Data structure	A way of organising and storing data in a computer to promote efficiency.
Debug	The process of detecting and removing the number of bugs in a computer program.
Doxygen	A tool to generate software reference documentation within code. Allows user to refer to actual code while reading the documentation.
Driver	A computer program that allows connection between higher-level software programs and computer hardware.
Eclipse	An multi-language integrated development kit (IDE). It contains an extensible plug-in system.
Firmware	The persistent memory, the data in it and program code within devices the provide control for it.
Front-end	Another name for Android Application.
Gantt Chart	Visual chart for viewing scheduling

GitHub	A software development project web-based hosting service using the Git revision control system.
Hash map	A data structure using a hash function to map a key to its corresponding value.
HTTP	Hypertext Transfer Protocol (HTTP), an application level protocol for hypermedia information systems. It is the main communication protocol for the World Wide Web.
Human Computer Interaction (HCI)	The field of study that investigates interaction between humans and computers.
IDE	Integrated Development Environment (IDE), a software application providing a suit of facilities (source code editor, build automation tools, debugger) for software development.
iOS	A mobile operating system developed by Apple Inc. for the iPhone, iPad, iPod Touch and Apple TV.
Java	A general-purpose, object-orientated and concurrent programming language designed to have as little dependencies as possible.
Java libraries	A set of libraries provided by Java natively.
JSON (JavaScript Object Notation)	Lightweight data-interchange format, that is easy for humans to interpret, and simple for machines to parse and generate. It consists of a 'key-pair' value. E.g. "VideoTitle" : "Charlie Bit My Finger"
Lazy loading	An android feature which allows individual components on the screen to load individually.
Metadata	Appended text to a video.
Operating system	A set of software that manages computer hardware resources and provides typical functionalities for computer programs.
Putty	A free and open source terminal emulator. It can act as a client for the SSH, Telnet, rlogin and raw TCP protocols.
Python	Programming language used for server communication.
SDK	Software Development Kit (SDK), a set of software development tools enabling the creation of applications for different computing platforms.
Splash screen	An introduction screen to the Android application.
SQL	Programming language designed for managing relational databases.
SQL Commands	Specific set of commands for querying (communicating) with a SQL Database and retrieving relevant results
SQLite	A small C relational database management system.

SSH	Secure Shell (SSH), a cryptographic network protocol used for secure data communication and remote shell services between two networked computers.[2]
UML	Unified Modelling Language (UML), a standardised modelling language for object-orientated concepts.
URL	Uniform Resource Locator (URL), a global address for finding documents and other files on the World Wide Web.
Waterfall methodology	Software Engineering Methodology.
Web service	Communication method between two devices over the World Wide Web.
WinSCP	Windows Secure Copy (WinSCP), a free and open source SFTP, FTP and SCP client for secure file transfer between a local and remote computer. It is for the Microsoft Windows.
Work Breakdown Structure (WBS)	Captures the breakdown of work packages into smaller unitary work units.
XML	Extensible Markup Language (XML), a markup language for encoding documents in human and machine-readable format.

Table of Contents

1. Introduction

This report outlines the implementation of an Android application to provide a new domain for accessing videos. The report has been written as such, to allow readers to gain an understanding of the various requirements and procedures that were followed, before the development of the application.

1.1 Background

With the growth of the internet in recent years, the use of social media as a means of communication has grown tremendously. This explosive growth of social media has resulted in sharing of media across various social networking sites. Among the shared content, is the uploading of videos, covering similar (if not the same) topics. These videos, containing common video content, are collectively known as, 'video memes'.

However, before identifying video-memes over the internet, an analysis into the formation of a video is outlined below.

1.1.1 Video – a series of images (frames)

A video is a series of images (or frames), interwoven and meshed together in a specific order, through various techniques (Figure 1).

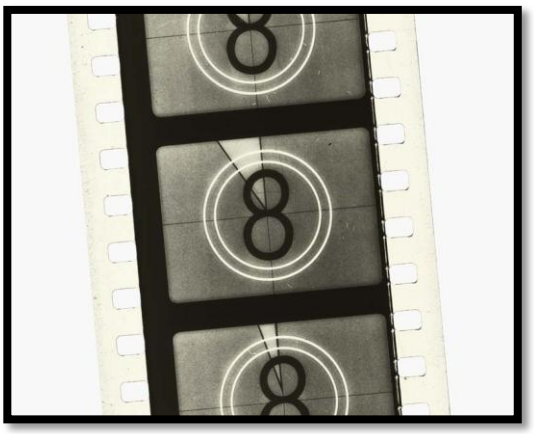


Figure 1: Video reel outlining the formation of a video [13]

Recent update on the YouTube website has allowed users to view these frames that constitute to form a video (story boards). Internally, YouTube splits each video into segments, where each segment is then split into frames. These frames are then stitched together, to allow users to visualise the video, at a glance. [12]

Figure 2: Updated YouTube browser outlining video frames [6]

1.1.2 YouTube – the preferred choice

For identifying video-memes, YouTube has been chosen as the preferred source of the videos, in the project. Although, there are other domains for uploading, browsing and obtaining videos online, YouTube, being the largest video content provider on the internet, with over 800 million unique visitors each month [14], has been widely adopted and caters for a large audience worldwide.

1.1.2.1 Metadata

Metadata refers to the content that describes a certain video, as presented by the video publisher. For a video published on YouTube, the metadata comes in various forms including:

- Video Title,
- Video Description,
- Video Tags,
- Upload Date and Time,
- Video Publisher,
- View Count,
- Ratings Count etc.

This metadata is especially useful in describing or identifying a video uniquely.

1.2 Limitations of existing search algorithm

The conventional way of searching videos is through the use of metadata. When a user types in a search query into the search box on YouTube, the search algorithm retrieves the relevant videos, based on the number of words and the order, in which the search query matches with the video's metadata. Although this method is effective, it does have an inherent drawback of potentially generating false results, due to its heavy reliance on user generated metadata, for the search. These limitations have been described through the use of real-life scenarios below.

1.2.1 Scenario 1 (incorrect metadata):

On 25th July 2012, when the search query 'PlayStation' was entered into YouTube, the following result was displayed on the first page. [7]

Figure 3: Search result for the term 'PlayStation' on YouTube [7]

In this scenario, YouTube matched the search term with the video description provided by the user. However, after viewing the video, it was clear that Lady Gaga was using a black Xbox controller. As such, in this scenario, an incorrect search result was produced due to incorrect metadata supplied by the publisher.

1.2.2 Scenario 2 (incomplete metadata):

Another search that was performed on 25th July 2012 was for 'top 100 videos'. The figure below was the first result that was displayed on YouTube. [11]

Figure 4: YouTube video with incomplete metadata [11]

The video is a compilation of top 100 videos on YouTube. It contains small clips from 100 original videos that have been previously uploaded by other YouTube users. However, the metadata does not provide any information regarding the source of these video clips, making it almost impossible to track down the original videos, if needed.

1.3 Proposed Solution

Identifying this problem, an algorithm was implemented last year, which provided an alternative means of searching videos, independent of its metadata. By matching the key frames of a reference video, against a pool of frames from various videos, reliable results that accurately matched the key frames from the reference video, were returned. The matching process as outlined in the paper "*Tracking Viral Videos on YouTube: Tool Set Construction*" by David Hehir [9], used the colour matching (colour-correlogram) and feature matching techniques, for retrieving the results.

1.3.1 Proposed Solution Example

The example below outlines the algorithm developed by David Hehir. By utilising the 'visual-meme tracking' techniques, frame by frame comparison can be achieved. The image on the bottom left in the figure below, passes the reference frame, whereas the video on the right, fails.

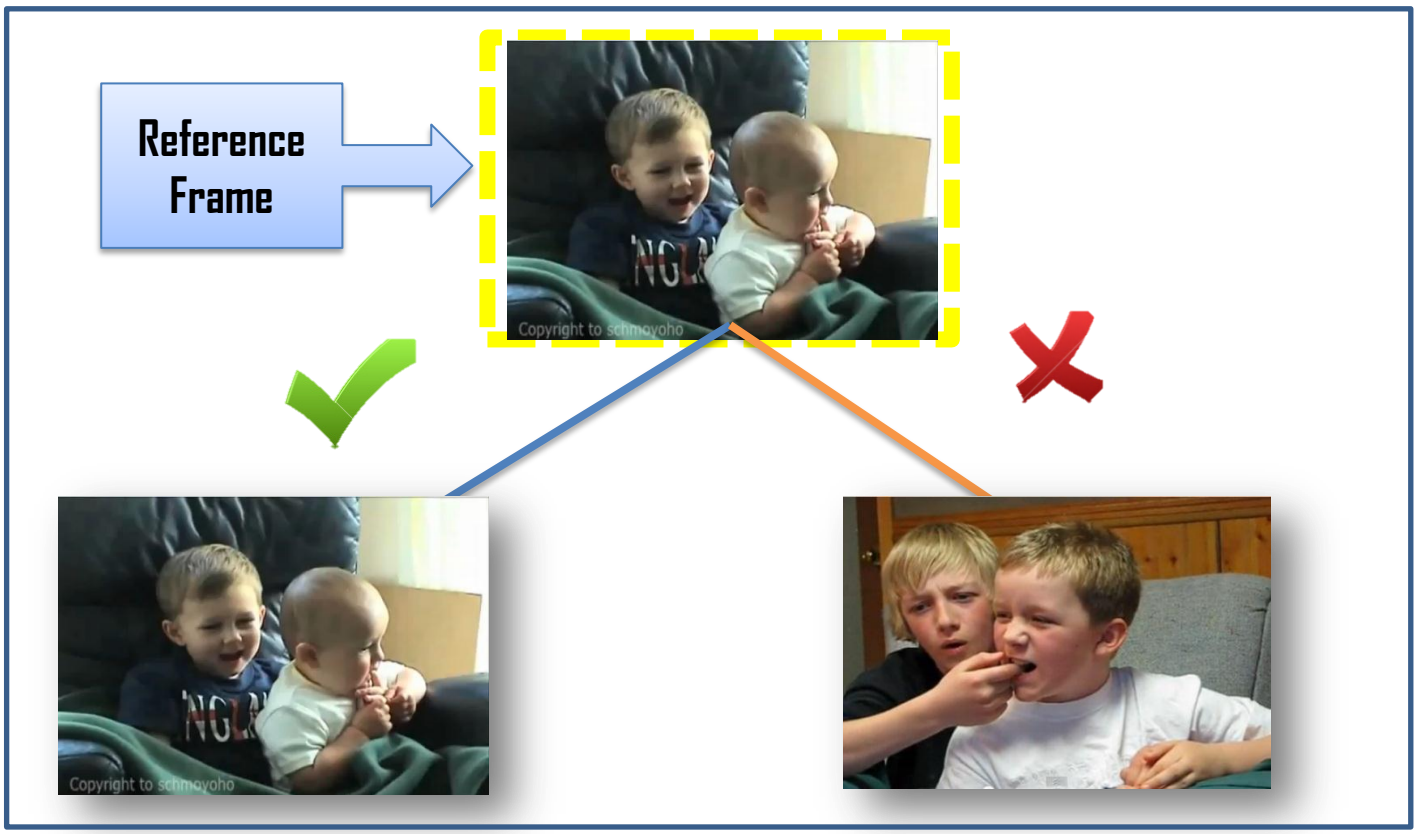


Figure 5: Example outlining the results produced by the matching-frames algorithm

2. Goal and Scope

2.1 Goal

The goal of the project is to utilise the available 'visual meme tracking' tools and provide a new platform for the users to engage through smartphones. The feedback obtained from the application users, can be collated to research further into developing and improving the required features of the search algorithm.

2.2 Scope

The scope of the project entails creation of an Android smartphone application that allows users to view and browse videos within the available categories, as well as find matching videos, to a selected video.

The main feature of the application will allow the Android application users to:

- View metadata for each of the videos
- View the available videos in each of the categories
- Provide a key-frame breakdown for a selected video in chronological order
- Provide matching key-frames for the selected key-frame

The project will also implement a back-end system utilising the Cantabile Server for providing the relevant content to the application.

The back-end system will include the following features:

- The required content for the application will be stored in a Sqlite database on the server
- Retrieval of data will be through JSON on HTTP Protocol

2.2.1 Constraints

Constraint	Description
Schedule	The time of the project cannot exceed past Week 13 of Semester 2 (2nd November 2012)
Resource	Only one student is working on the project, providing approximately 10 hours of effort per week

Table 1: Constraints table for the project

2.2.2 Assumptions

The following assumptions were made in the completion of the project:

- Access will be provided to the Cantabile Server by Australian National University.
- Any costs and resources required for the deployment of the system (apart from what has been listed in the scope of the project) will be provided by Australian National University.

2.2.3 Stakeholders

Before the requirements are defined, it is essential to first define the various stakeholders involved in the project. For this project, the various stakeholders are:

- Dr. Lexing Xie – Project supervisor and an acting client
- Anshul Saini – System Developer/ Project Manager
- Honglin Yu – Technical assistant for server side queries
- David Hehir – Developer of matching frames algorithm
- Android Application Users – Review the application, test it and provide relevant feedback such as additional features, identification of bugs, or any UI changes etc.

3. System Design and Requirements

As part of the development process, it is essential to first develop a high-level architecture diagram. Using this diagram, a set of requirements for the system specifications, can be defined. Using these requirements, the high-level diagram can be broken down into its low-level design, depicting each individual module's input and output parameters. The procedure ensures that any changes in the client requirements are correctly reflected before the development of the application, saving time and resources, later down the track. These documents are also used in the testing phase where the software is verified and validated against the pre-defined requirements.

3.1 Architecture diagram

The following diagram below provides a high-level overview of the system and its interaction with the various sub-systems. It also lists the system boundary, and how the work undertaken as part of the project, communicates and builds upon the works of others, in the field (as outlined through blue and green boxes).

As can be seen, the actual process of meme-tracking (identifying matching frames to a reference frame) is pretty complex and was undertaken separately by David Hehir for his Honours project, last year. As such, these set of tools to perform 'meme-tracking' (as outlined in the blue box in the figure below) have been treated as a black box. Essentially, through the use of provided documentation, the input parameters were provided in the specified format to retrieve required output for the development of the application.

The green box in the figure outlines the work that was undertaken as part of the thesis. It depicts the development of the back-end (server-side) on the Cantabile server provided by Australian National University (ANU) and its interaction with the front end (Android smartphone application). Essentially, the output of the scripts and programs that have been provided for the retrieval of matching frames, were stored and retrieved through a database (on the server) using Python Scripts in JSON Format over HTTP Protocol.

In the Android application, a web service communication client was built to retrieve the content from the server over the HTTP Protocol for its display in the relevant sections of the application (Categorised Videos, Matching Videos and Matching Key frames).

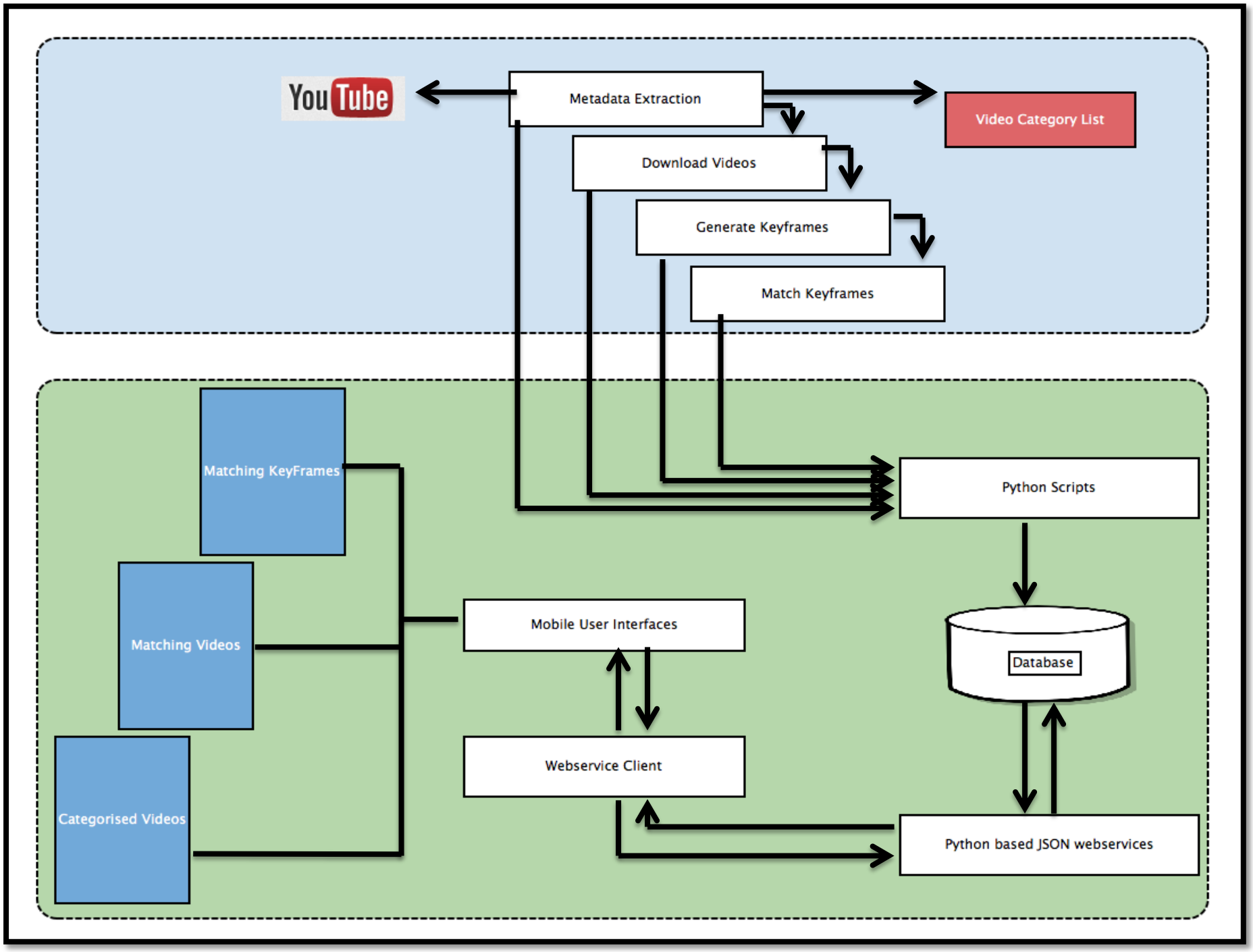


Figure 6: High-level architecture diagram of the project

3.2 Software Requirements Specifications (SRS)

In this section, the software requirements specifications (SRS) for the system are listed. Through SRS, a description of the system's behaviour can be developed, by listing all the necessary requirements for the development of the project.

For large software projects entailing various stakeholders, this is usually developed as a separate document. However, for an individual project, only relevant sections have been used.

The requirements for a project can be broken down into two main categories namely, functional and non-functional requirements. The section below outlines and categorises the requirements that have been gathered and analysed through various meetings with the project client (Dr. Lexing Xie).

3.2.1 Functional Requirements

Functional requirements capture the intended behaviour of the system by defining the expected capabilities and functions of the system. These requirements have been sub-categorised into Android Application requirements and Server-Side requirements.

3.2.1.1 Android Application Requirements

- The android application must contain three screens.
- Each page must contain a header, home button, title and a bar at the bottom to separate it from the displayed page contents.
- The application must have a consistent theme (including colours and images) across all screens.
- It must retrieve updated content from Cantabile server.
- It must cater for different screen sizes.

Screen 1 content (Videos within Categories):

- The screen must list all the categories under which the videos have been categorised.
- Each category must have a name associated with it.
- The application must list all the videos for each of the categories.
- The videos must be displayed as horizontal gallery where a user can swipe across to view more videos within each category.
- Each video must contain an image, video title and the name of the publisher.
- The user must be able to select one of the videos.

Screen 2 content (ListView Video Matching):

- The screen must have two tabs at the top.
- The tab names must be provided.
- In screen 2, first tab must be pressed to inform the user that he/she is currently viewing tab 1.
- Below the tabs, the metadata for the selected video must be shown.
- The metadata must list the title, description, author, upload date, number of views and any other associated tags.
- The frame for the selected video must be shown.
- In the frame of the selected video, near the bottom right corner, the length of the video must be shown.
- Below the reference video metadata, the metadata for all the matching videos must be shown.
- This metadata (including the matching reference frame) for the matching videos, must be shown in a list view.
- The user must be able to scroll up and down in a listview, if in case; there are more videos than what the screen can show.
- When a listview item is pressed, relevant YouTube link should open up.

Screen 3 content (Cross Browser Frame Matching):

- The screen must have two tabs at the top.
- The tab names must be provided.
- In screen 3, the second tab must be pressed to inform the user that he/she is currently viewing tab 2.
- Below the tabs, a cross browser must be present.
- A cross browser must display a list of key frames for the reference video.
- The user must be able to select one of the key frames as the reference frame.
- The user must be provided with some form of visual feedback to depict his/her selected key frame.
- Matching frames must be displayed for the reference key frame.
- Each key frame must have an associated timestamp with it at which they occur in time axis.
- The key frames must be in chronological order to portray the sequence in which they appear in the video.
- When a matching key-frame is pressed, relevant YouTube link should open up.

3.2.1.2 Server-Side Requirements

- A database must exist on the server.
- The database must store the output from each of the provided scripts (metadata, key frames etc.).
- The database must reflect any changes made in the category list.
- Python scripts must be used for querying the database.
- The queried results must be displayed via HTTP Protocol on a valid public URL link (web address).
- The output over HTTP Protocol must be through JSON format.

3.2.2 Non-functional Requirements

Non-functional requirements define the operation of the system, rather than a specific feature and/or behaviour. If these are not analysed and validated as part of the design process, the system can have unexpected behaviour. For the project, the non-functional requirements are:

3.2.2.1 Android Application Requirements:

- The application must be robust i.e. should not be prone to common errors.
- The application's loading, processing, response and browsing times must be fast.
- The application must be extensible i.e. modular approach should be taken.
- The application must perform (possible) fault trapping, to convey meaningful error messages to the developers, when the application crashes.
- The code must be well documented.

3.2.2.2 Server Requirements:

- The database down-time must be minimal.
- The server architecture must be designed as such, to allow it to extend to other platform applications.
- The database must be capable of storing arbitrary amount of categories and video related content within it.

The loading time of the application is inherently dependent on the internet speed available. As such, it is assumed that a fast, reliable connection for the application is always available.

The following server requirements, 'the database down-time must be minimal' and 'the database's capacity to hold arbitrary amount of data', inherently depend on the server's uptime and capacity to hold data. As such it is assumed, that these two requirements have already been met, as the Cantabile server (and as such, its testing), was provided by Australian National University.

3.3 Low level Design

Using the high level architecture diagram, and the specified requirements, a low level design detailing the exact interactions, relationships and linkages between the various classes and modules, is developed.

3.3.1 Android Application Design

An Android application is usually built in the programming language named Java, with the use of XML for designing the user interface (or the layout) of the application. For such software, spanning multiple classes, one of the ways of depicting a low-level design is through a UML class diagram which outlines the various classes, attributes, operations and relationships between different classes of the system.

3.3.1.1 Class Diagram

The figure below represents a class diagram for the Android Application. It shows the three different activities utilising three different custom adapters.

There are also two main elements present in the class diagram, namely, Video Attributes and Key frame attributes. These two elements represent two different entities for the two different data sets. The element 'Video Attributes' is associated with the metadata attributes of a video such as its author, time duration, description, etc. This is different to the 'Key frame Attributes' element which represents the frames associated with a particular video. Its attributes are time (in seconds) at which that frame appears in the video and its associated thumbnail image. The adapters populate the appropriate view of these elements as per the User Interface needs. The data for these elements is retrieved through a URL (web address) in JSON format. The data obtained in that particular format is then processed and organised using native data structures such as hash maps. The information from these data structures is then efficiently utilised by the adapters for populating the relevant views.

Each attribute is represented as a tag in JSON Object containing a non-empty value. As these two elements are associated with a different set of attributes, they have to be processed differently. However, as the process of collating and storing the content from the URL is the same, an interface class named Processing, interacting with various classes, is designed.

Although the two aforementioned elements have unique attributes, the thumbnail image attribute is common to both. As such, the process of handling and displaying an image is the same across the application. Therefore, a separate class named ImageProcessing is designed specifically for processing images.

By separating out the sub-components of the application into different classes, decoupling (low dependency between classes) can be achieved. This allows the design of the application to be extendable, requiring minimal changes to existing design for implementing further features in the application. Essentially, change in one module, will not create a ripple effect of changes in other modules. This design process, of low coupling and high cohesion, was applied across the application, allowing it to be more robust, for future needs.

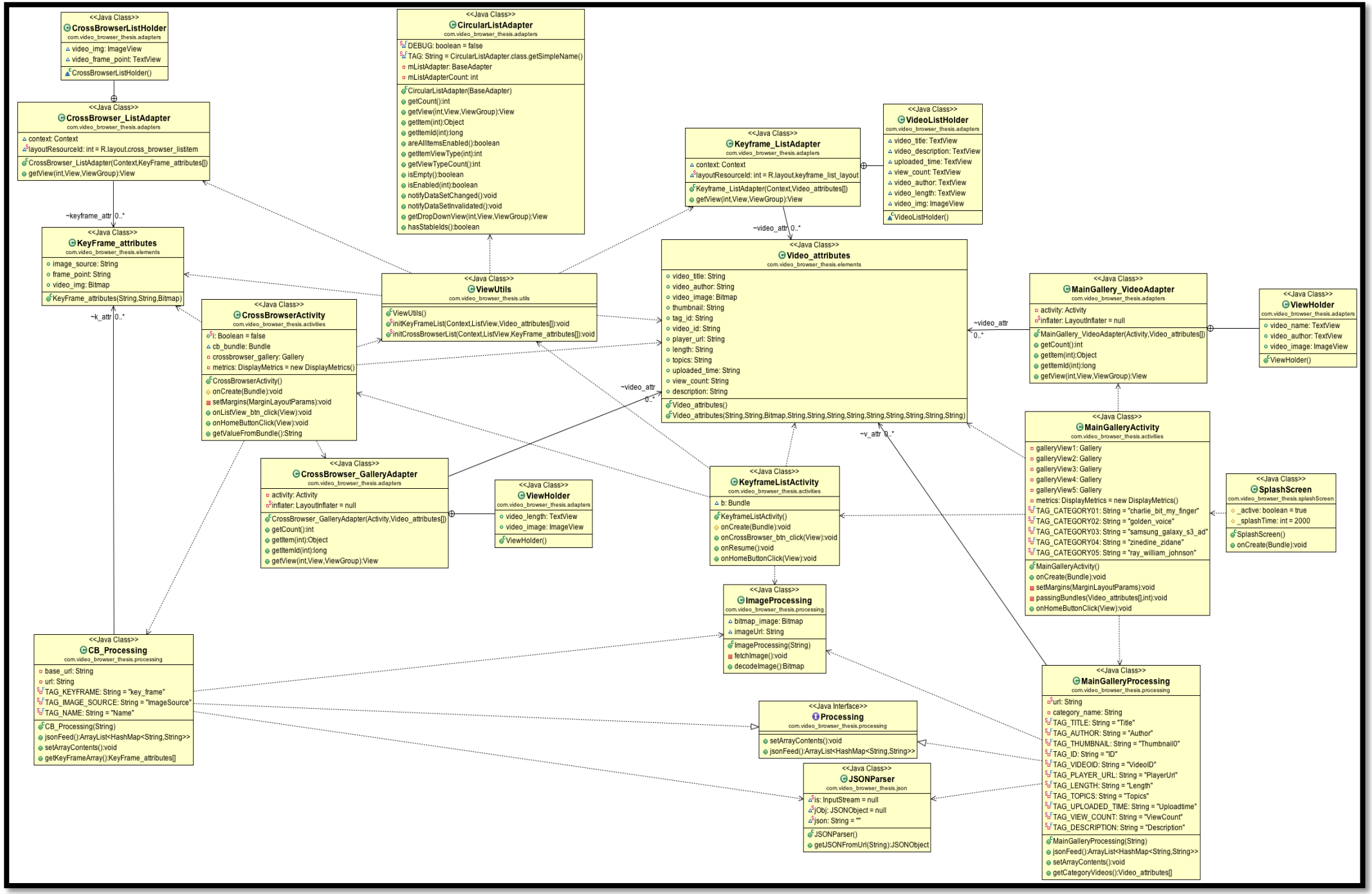


Figure 7: Class Diagram for Android Application

3.3.1.2 User Interface Prototype

As part of the low-level design specifications, mock-ups were also drawn to prototype the User Interface design before its development. This allowed for any potential changes in the design, early in the project's lifecycle, while providing the client with an opportunity to visualise the application.

The application mock-ups were drawn in the Balsamiq software. Using Balsamiq for drawing mock-ups has become the standard for generating software prototypes. It has a distinct advantage over generic sketches drawn in Microsoft Paint, Word or PowerPoint, as it provides customised designs and stock functionality catered for smartphone application prototyping, with the additional feature of a story-board (clickable links). This makes it easy for the users to browse the application.

In the project, the mock-ups have gone through two iterations, as shown below.

Iteration 1:

As specified in the requirements, the application was split across three screens. The first screen had two galleries. The first gallery provided a list of categories. Each category, when pressed, displayed the relevant videos corresponding to the category in the second gallery below. However, it was missing the appropriate metadata such as the video title and author in the second gallery.

The second screen was also prototyped using the client requirements. Following from the first screen, when the user clicks on a video, a video frame for the selected video, with its metadata, is shown at the top (below the tabs). The other videos' metadata for the videos containing at least one matching key frame is shown in the listview below. When a user selects on one of the listview items, he/she is directed to the relevant video on YouTube.

Pressing the 'cross-browser' tab, will open up the third screen, containing the cross browser. The cross browser will consist of a listview (vertical) and a gallery (horizontal) displaying the key-frames of both the selected video, and the key-frames from other videos, that match up with the currently selected key-frame, in chronological order.

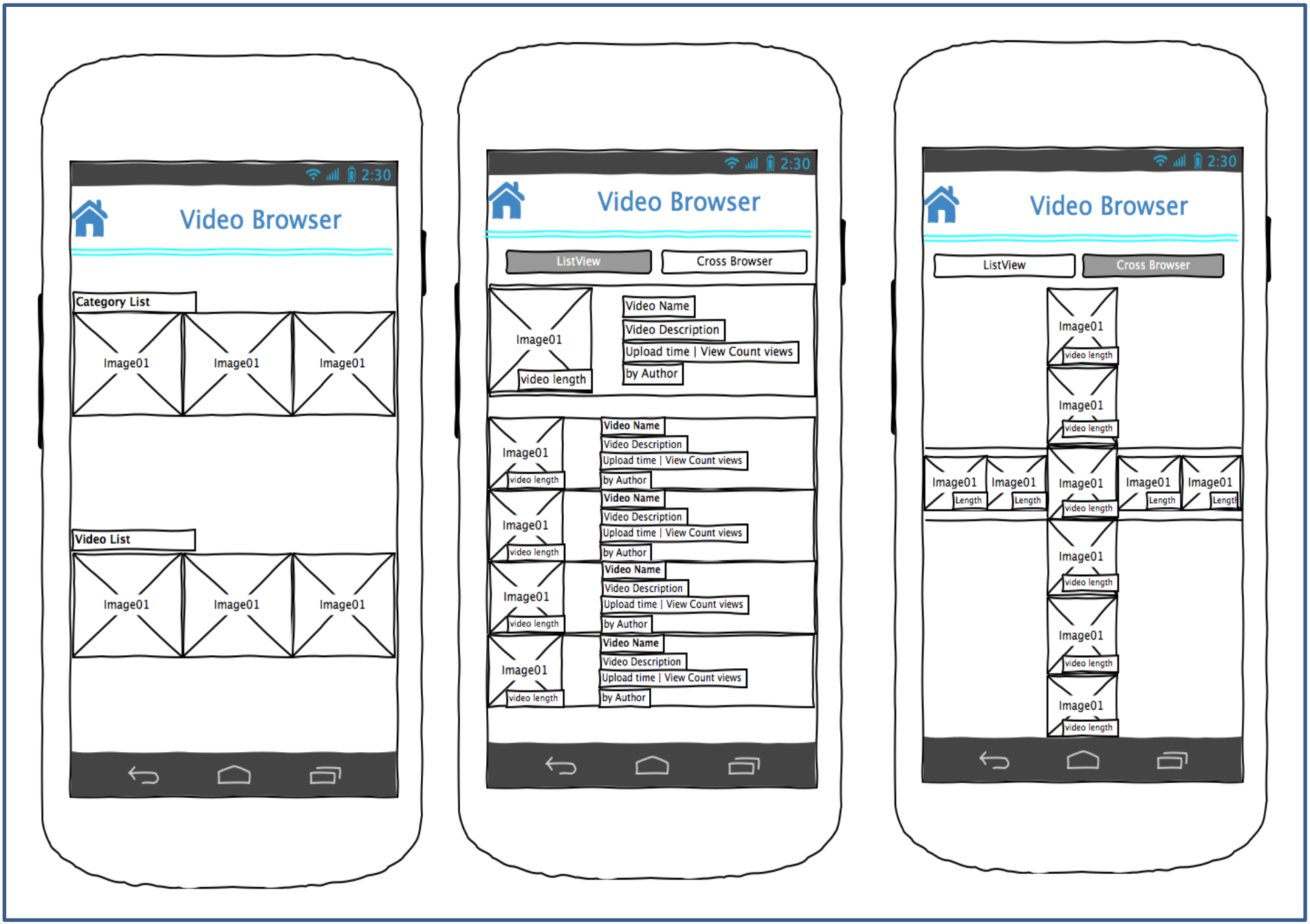


Figure 8: Iteration 1 Screen Prototypes (Mockups)

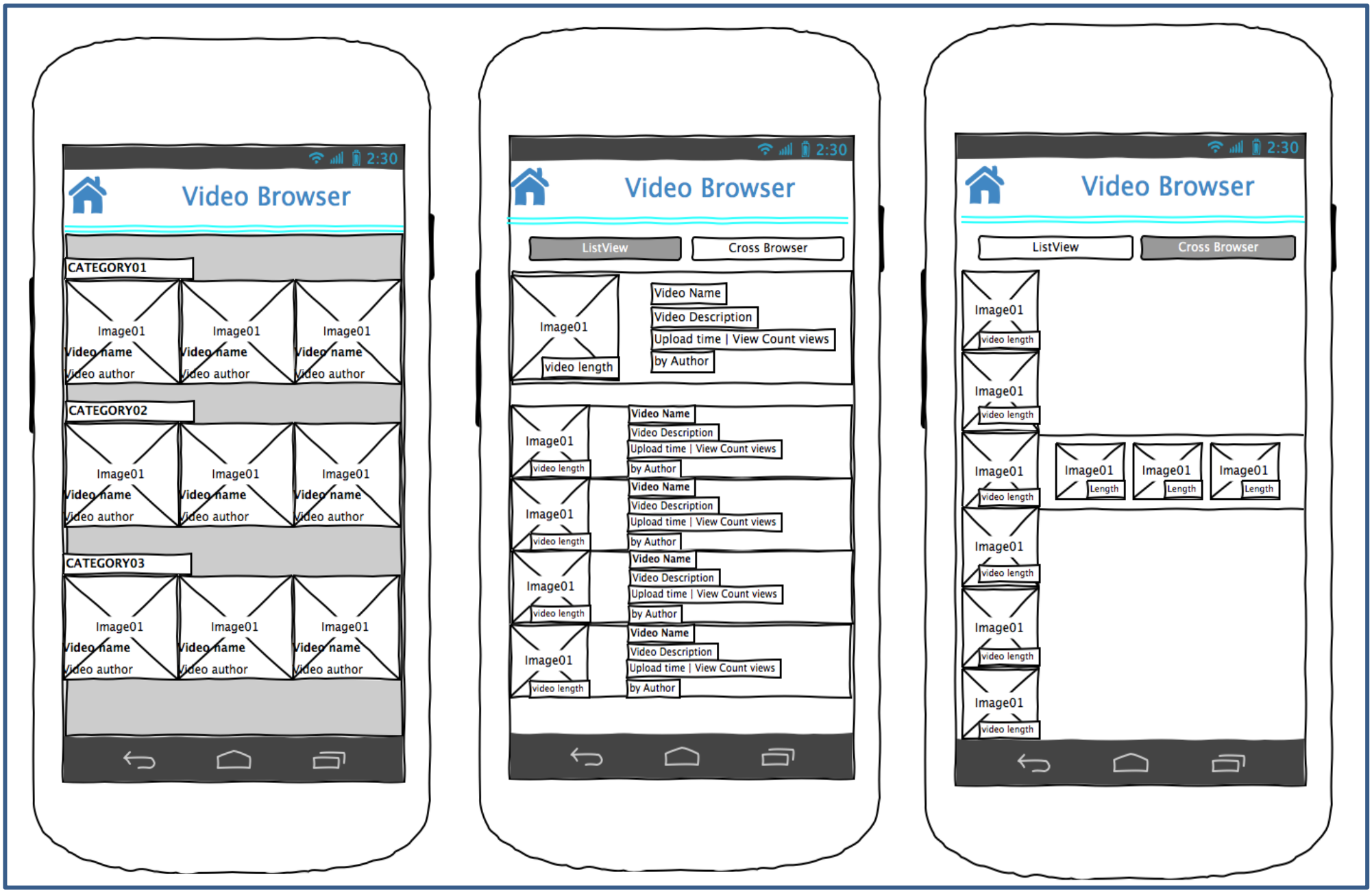


Figure 9: Iteration 2 Screen Prototypes (Mockups)

Comparing iterations 1 and 2, a few changes can be observed in the mock-up. Instead of two galleries in the first screen, a gallery now exists for each of the categories, allowing vertical scrolling if several categories exist. Also, each video now contains the relevant metadata, specific to each video. This change in the first screen allows for better utilisation of the smartphone screen, while allowing users to view all the videos for each of the categories, at once.

In Screen 3, the cross browser has been changed to a T browser. Provided the constraints of the available features on Android platform and the ease of implementation, it was decided that a T browser was better suited for the project. While making these changes over the two iterations, it was ensured that the requirements integrity was maintained.

3.3.2 Server-side Design

In this section, the low level design for the implementation of the server-side is described using the high level architecture diagram and the specified requirements.

3.3.2.1 Reading scripts

As the project uses scripts that were previously built by others, it is first essential to understand the provided scripts, its interactions and execution procedures. On the server, the following scripts have been provided:

1. **config.ini**: This file stores all the input parameters (or the path to access the input parameters) for various scripts listed below.

2. **crawlxml.py**: This script provides the necessary inputs using config.ini file for xmlcrawler.py script to run and stores the results in a directory. It also creates a tar (compressed zipped directory of the results directory) for easy storage on the server.

3. **xmlcrawler.py**: This script downloads the metadata (text attached to each video) for the list of categories provided.

4. **crawlvideo.py**: This script provides the necessary inputs using config.ini file for videocrawler.py script to run.

5. **videocrawler.py**: Using the inputs provided by crawlvideo.py, it uses youtube-dl.py script to download the videos and categorises the results into relevant sub-directories for easy access.

6. **youtube-dl.py**: This script downloads the videos. It uses videocrawler.py script to provide the input such as the video-link for downloading.

7. **batchrun.py**: This script identifies the key frames associated with each video. Using config.ini file, it iterates over an entire directory to search for videos.

Matching of key frames requires the use of two executables, namely, ColourCorrelogram and FeatureMatcher.

- ColourCorrelogram requires three inputs parameters: Image Directory, Output Directory and the Radius, where image directory corresponds to a folder of jpg images, Output directory is the location for storing serialized JSON, and the radius corresponds to the search radius.

- FeatureMatcher also requires three input parameters: (Serialized) JSON Input File, Output JSON Folder, and the Tau value, where JSON Input File is the serialised file produced by ColourCorrelogram, Output JSON Folder is the directory where the serialized cluster of image names are stored and the Tau value is the third parameter, as specified in '*Tracking Viral Videos on YouTube: Tool Set Construction'* by David Hehir. [9]

3.3.2.2 Low-level design

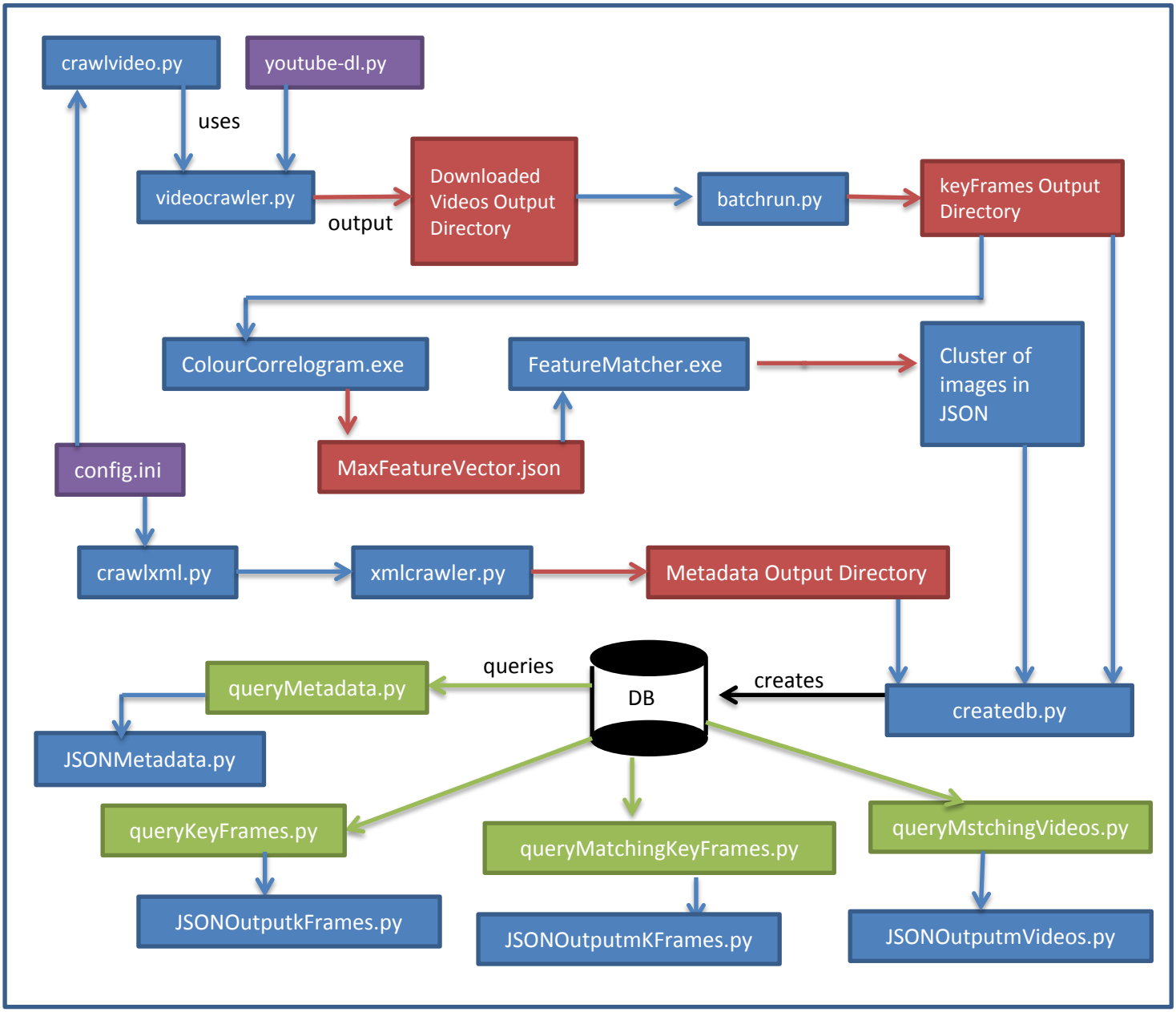


Figure 10: Low-level for server-side programming

3.3.2.3 Writing Scripts

In figure 10, the low level diagram of the server-side is provided. Using the provided scripts, the following python scripts are needed for collating the data into a single database, and accessing the output over the HTTP protocol. This will be useful as currently the data is accessed over the SSH protocol (using command window) requiring authentication. By using the HTTP protocol, the database results can become accessible through a public URL link (web address), requiring no authentication.

createdb.py: This script collates the output of the provided scripts and stores the results in a Sqlite database. The advantage of Sqlite over other databases is its lightweight processing that is especially beneficial for server communication.

Previously, each script, namely, crawlxml.py, crawlvideo.py and batchrun.py had to be run separately through SSH. By linking the scripts with each other and piping the output of one script as an input to another, only a single script will require execution. Also, previously, the outputs from the scripts were dispersed across several directories. By using a database, all the data can be easily accessed through one location.

Another advantage of creating a database is that the stored data will be always available and as such, can be accessed at all times, provided that the database is present and the server (hosting the database) is functioning correctly.

For the Sqlite database, the following schema (Figure 2,3 and 4) would be used for creating the tables and linking them with each other.

Video Metadata Table

Name	Type
Id (Primary Key)	Integer
Title	Text
VideoID	Text
Query	Text
Rank	Integer
Uploadtime	Datetime
Category	Text
Tags	Text
PlayerURL	Text
FlashPlayerURL	Text
Length	Double
Author	Text
CommentCount	Integer
FavouriteCount	Integer
ViewCount	Integer
RatingAvg	Text
Thumbnail0	Text
Thumbnail1	Text
Thumbnail2	Text
Topics	Integer

Table 2: Schema for Video Metadata Table

Image Frames Table

Name	Type
Id (Primary Key)	Integer
ImageURL	Text
ImageName	Text
VideoReference	Text

Table 3: Schema for Image Frames Table

Matching Frames Table

Name	Type
Id (Primary Key)	Integer
ReferenceFrameID	Text
MatchingFrame	Text
MatchingFrameURL	Text

Table 4: Schema for Matching Frames Table

As part of developing the database, foreign key constrains would also be considered.

queryMetadata.py: This script will use SQL commands to retrieve the metadata results. As all the metadata content is required at once, no input parameters would be required.

JSONMetadata.py: The raw data extracted from queryMetadata.py would be formatted to JSON, consisting of key-value pairs (e.g. "Title": "Charlie Bit My Finger" etc.) for transmitting over the URL (web-address). This format will make extraction of content easier and more robust, in the application.

queryKeyFrames.py: Similar to queryMetadata.py script, this script will extract the key frames from the database for a specified video (using unique video ID as the input parameter).

JSONOutputkFrames.py: Same functionality as JSONMetadata.py of converting the raw text from JSONOutputkFrames.py, to JSON for transmission over the web-address.

queryMatchingKeyFrames.py: This script will extract the matching key frames from matching frames table in the database for a specified reference frame (using the image name as the input parameter).

JSONOutputmKFrames.py: This script will convert the raw text from queryMatchingKeyFrames.py, to JSON for transmission over the web-address.
queryMatchingVideos.py: This script will provide the metadata for the matching videos, to the reference video.

JSONOutputmVideos.py: This script will convert the raw text from queryMatchingVideos.py to JSON for transmission over the web-address.

4. Project Management

The success of a project is measured by the way it is managed. This section outlines the various project management principles and tools that were used to streamline the process, and increase assurance in the outlined plan for the completion of the project.

4.1 Lifecycle Model

The first aspect of a software project management is identifying an appropriate lifecycle methodology, which best meets the needs of the project.

Analysis of different lifecycles revealed that Waterfall methodology would be best suited for the project, as the requirements were relatively straight forward. Using this lifecycle methodology, allowed for better scheduling of tasks, with definitive start and end times.

Although as a result of these set times, there was a possibility of compromising the output quality; the flexibility offered as part of the student project, in delivering the milestones, allowed for the expected quality to be delivered.

4.2 Work Breakdown Structure (WBS) and Scheduling

The following figures (Figure 11 and Figure 12) outline the Work Breakdown Structure (WBS) and the Gantt chart that was used in the project.

4.2.1 Work Breakdown Structure (WBS)

The Work Breakdown Structure shows the decomposition of the work packages into single workable units, allowing for better scheduling and allocation of tasks. It provides a high-level planning for the project, while ensuring all sub-components have been accounted for. For e.g. as part of the feature breakdown in prototyping (or wire-framing stage), all three features were listed.

It also assists in identifying dependencies in work packages represented on a Gantt chart. An example of this is, learning open project for WBS, before initiating the scheduling for the project.

Lastly, it breeds commitment, as the involved parties can get a better understanding of the amount of work, which needs to be completed. It can also assist in any potential future changes in the project.

4.2.2 Scheduling

Using WBS, a Gantt chart outlining the schedule and the dependencies between work packages, was prepared. With a full time study load, this allowed for better time management with other course assessments, across the two semesters, and preparation of contingency plans for any foreseeable delays.

By using WBS and Gantt chart, the project benefitted in the following ways:

- Transparency, traceability, and accountability for tasks.
- Planning out the required tasks.
- Scheduling of tasks and their completion dates.
- Identification and allocation of resources, and,
- Identifying any possible critical paths.

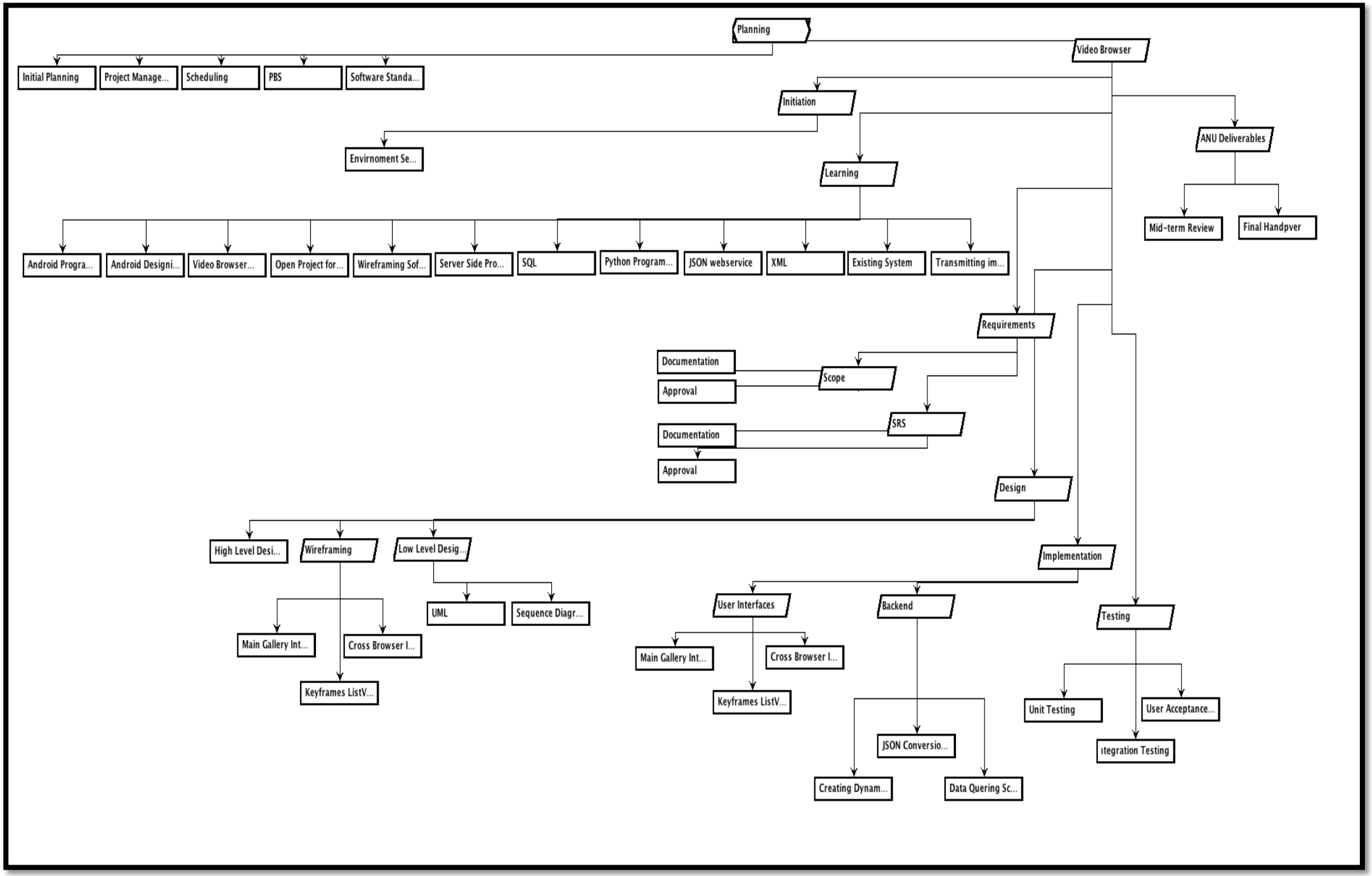


Figure 11: Work Breakdown Structure (WBS) for the project

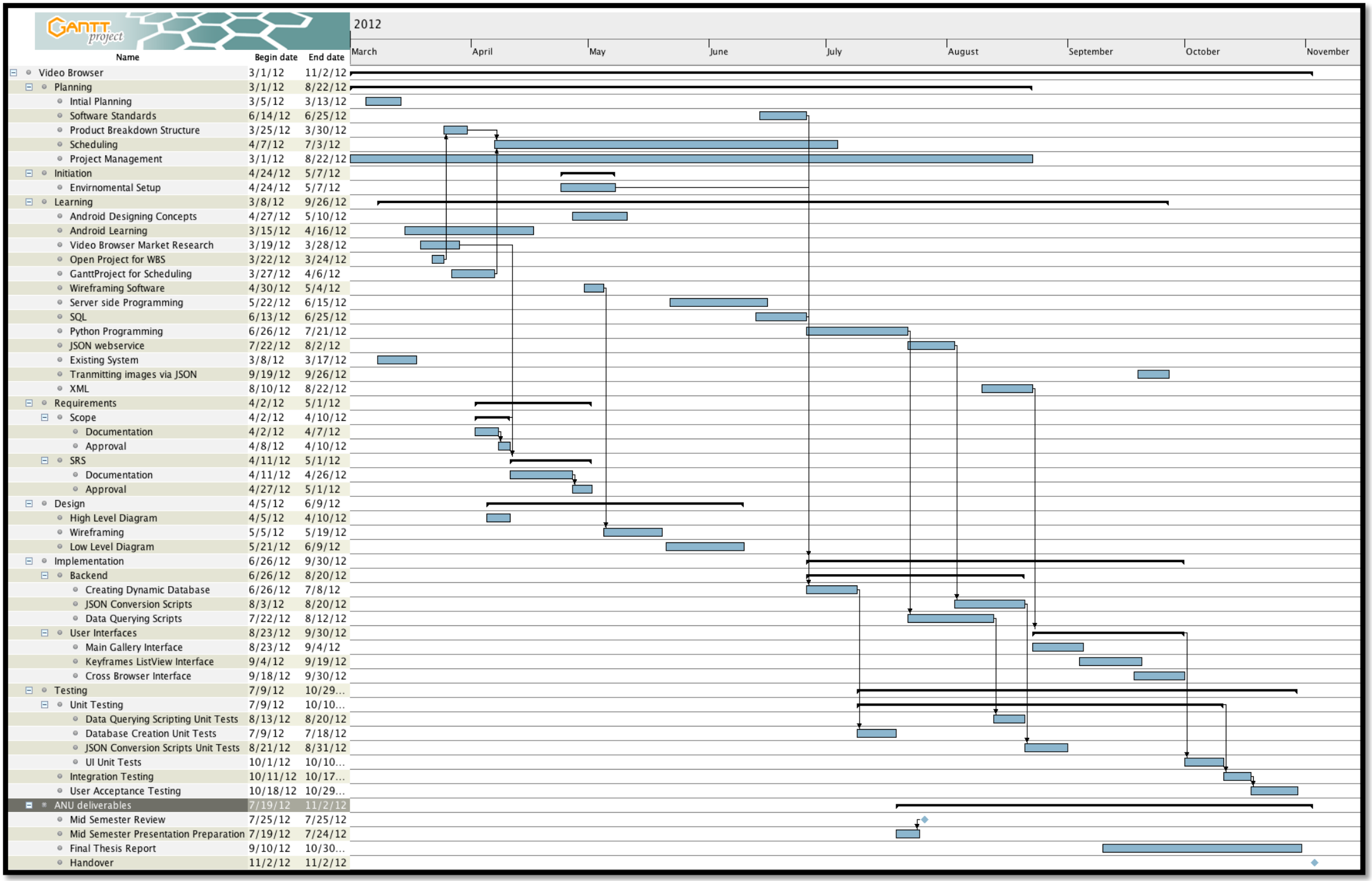


Figure 12: Scheduling of tasks using Gantt Chart

5. Code

This section discusses the various aspects of the code implementation for both server side and Android application programming, using low level designs as reference.

5.1 Design

5.1.1 Android Application Programming

As the low level design was thoroughly planned in outlining the required features, and meeting the project requirements, little change was needed between the low level design and its implementation.

During the code development phase, some assistance was used through the Android Open Source community, which has been referenced appropriately in the relevant sections of the code.

As part of the User Interface Development, external images have been used and have been listed in the bibliography.

The figures below show a comparison between the developed User Interface screenshots and the prototypes, initially designed. When developing the User Interface for the application, following points were considered:

- Application of a consistent theme across the application (Blue-Black colours)
- Caching (or temporary storage of data) to avoid un-necessary downloading, within the application, allowing fast browsing, within the application
- Addition of splash screen, while the application loads in the background

Screen 1 (Category Video Screen Comparison):

Figure 13: Screen 1 comparison

Screen 2 (Listview Screen Comparison):

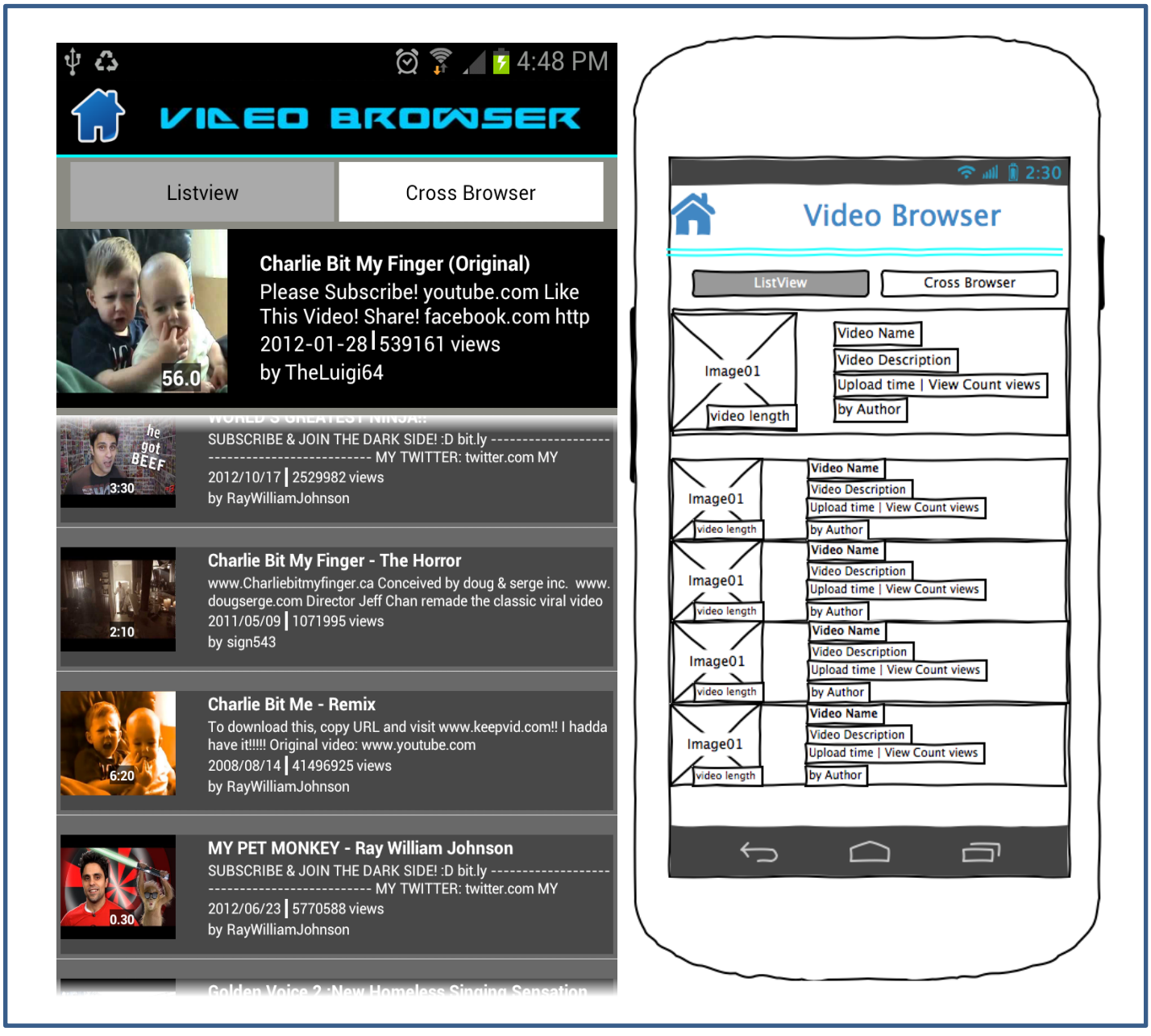


Figure 14: Screen 2 Comparison

Screen 3 (Cross Browser Screen Comparison):

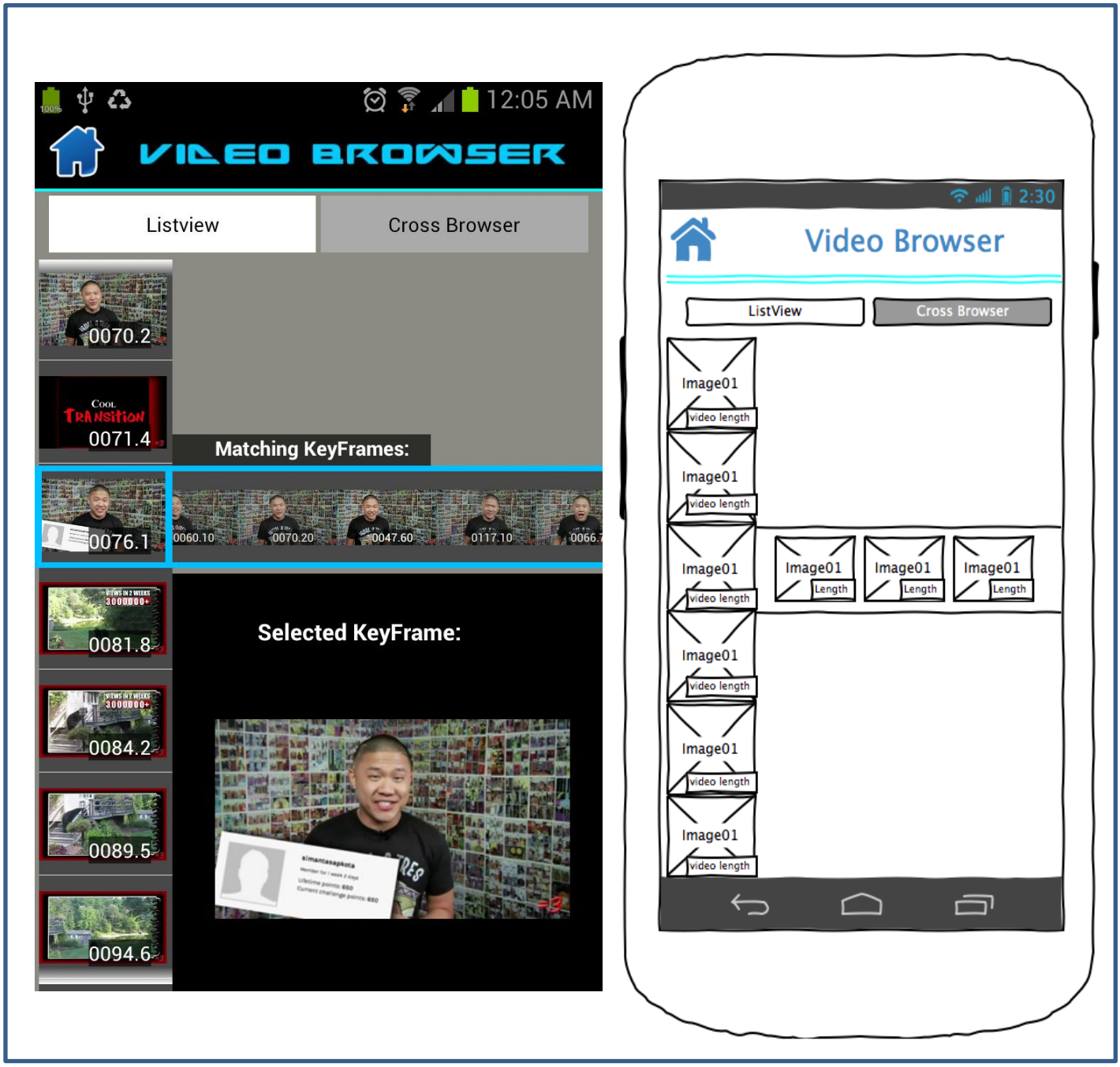


Figure 15: Screen 3 Comparison

Comparing the User Interface of the screens with the prototypes previously drawn in Balsamiq, similarities can be observed. This is expected, as the prototypes were used as the reference for designing the User Interface. As such, all the components, such as the listview and the gallery were strategically positioned on the screen, to replicate the prototypes.

5.1.2 Server-side Programming

In comparison to the low level design for Android, there was initially a slight change in the implementation of the server side scripts. Initially, for transmission of images over HTTP Protocol using JSON, a binary conversion of the images was performed, as Sqlite databases, don't have an inbuilt type to store images, natively. As such, for transmission, the binary was encoded in Base-64 to allow transmission of the images as ASCII characters.

On the Android, this was decoded back from Base-64 to Binary, and finally to a Bitmap for display. However, this ended up in large amounts of processing on the phone, resulting in unexpected crashing of the application. As such, the design was changed back, with the URL of the images provided in the database.

Another change was the grouping of python classes. In the low level design, a python script was designed for querying the database, while the second script handled its conversion to JSON format. However, during the implementation, it was realised that it would be better if both the querying and conversion occurred in a single python script.

5.2 Challenges

5.2.1 Android Application Programming

Traditionally, a gallery only consists of a set of images. As such, for displaying text such as the video length or the timeframe at which a frame appears in a video, a custom gallery adapter, extending the inbuilt gallery, had to be designed.

Similarly, an inbuilt Android listview can only contain a single text field. As such, for displaying an image and multiple lines of text (author, description etc.) required the development of a custom adapter. This required research into understanding of adapters.

Another challenge was displaying multiples galleries in a scrollable view, such that, galleries that are not presently visible on the screen, can be scrolled to, using the vertical swipe gesture.

5.2.2 Server-side Programming

Designing a python script, with no prior experience, was considerably harder than understanding existing scripts. This in general was a big learning curve and took a significant amount of time, due to the complexities involved in understanding, developing and debugging in a (previously) unknown language.

Initially, the scripts were relatively easy to make. However, as the number of features added within the script increased, such as, formatting of content to JSON, the complexity (and hence, it's debugging time) increased as well.

Python is a weak-type language. Essentially, the type checks such as whether the provided content is an Integer, character or a string etc., is not enforced on the user, resulting in type conversion problems at run-time. These issues, although technically easy to fix, can be tedious, as python scripts don't usually provide a meaningful error message, when accessing over a URL. This problem was overcome using SSH window for running the scripts, which provided meaningful output as to why (error type) and where (line location) the script failed.

5.3 Standards

As part of the development process, various coding standards were followed, as shown below:

5.3.1 Standard Java Coding Conventions

The Code Convention for the Java Programming, as recommended by Sun, has been adopted in the code development process of the Android application. The conventions

and practices are expected to improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly. It covers various aspects of programming, such as filenames, file organization, indentation, comments, declaration, statements, white spaces and naming conventions.

5.3.2 Android Open Source Project Code Style Guidelines for Contributors

Further to the Java Coding Conventions, Android has provided a list of guidelines that an open source developer must follow, for publishing Open Source Project online. These vary from commenting rules, to import of classes etc. [1]

5.3.3 JavaDocs Style Comments

For commenting, JavaDocs commenting style has been adopted. As part of the convention, each document comment begins with,
"/**

And ends with
**/"
For implementation details, such as purpose of specific variables and expressions, one line comments have been used. Also, every class and method, is preceded with a descriptive comment as part of the convention. Following this, will further assist in readability of the code. [4]

5.3.4 Doxyen

As part of the documentation process, Doxygen (or documentation generator) has been used as a tool for software reference documentation. It uses the commenting style adopted as part of JavaDocs convention, and generates a cross reference documentation and code, such that the readers can easily switch between the documentation and the code. [2]

5.3.5 JSON Validator

To ensure that the JSON output received through the HTTP Protocol was in the correct format, an online JSON Validator named JSONLint (www.jsonlint.com) was used for the validation. [5]

6. Testing

As part of the development process, testing methodologies were applied on the implemented code, to measure its Correctness and Completeness. Usually testing can also assist in identifying potential security issues related to software, but, provided the scope of the project, only standard testing was performed. Also as the data used by the application was publically accessible, no code implementation for the security of the application was needed.

6.1 Unit Testing

In unit testing, individual modules (or classes) alongside any associated data and procedures, are tested to see if they produce the expected results. For unit testing, knowledge of how the code functions is required, and as such, is usually categorised under white-box testing. [10]

For testing the Android application, unit tests were written for Java classes. For Python scripts, tools such as www.jsonlint.com were used to verify the output. The JSONLint website checks whether the content on a URL has the correct JSON format.[5]

6.2 Integration Testing

Integration testing builds upon the unit testing of the individual modules. It tests the output when two or more individual modules are connected together, and ensures that distinct components of the application still conform to the client requirements. Integration testing also falls under white-box testing, as the knowledge of the code for the module, and its interfaces with other modules, is needed. [3]

For the project, test cases were developed for main interfaces within the application.

6.3 System Testing

System testing of software is conducted on the complete, integrated system. The system's compliance is evaluated with its specified requirements. System testing falls under black-box testing as no knowledge of the inner code or logic is needed for the evaluation. [8]

For the project, the requirements of the project were validated with the specified requirements and the scope of the project.

6.4 User Acceptance Testing (UAT)

A user acceptance testing was performed on a small group of friends with Android Smartphones, during the development phase, while the various features were being implemented as outlined in Appendix A. This was to receive possible feedback on the aesthetics, the User Interface and identification of potential bugs in the application. The results were collated and plotted, as seen in the figures below:

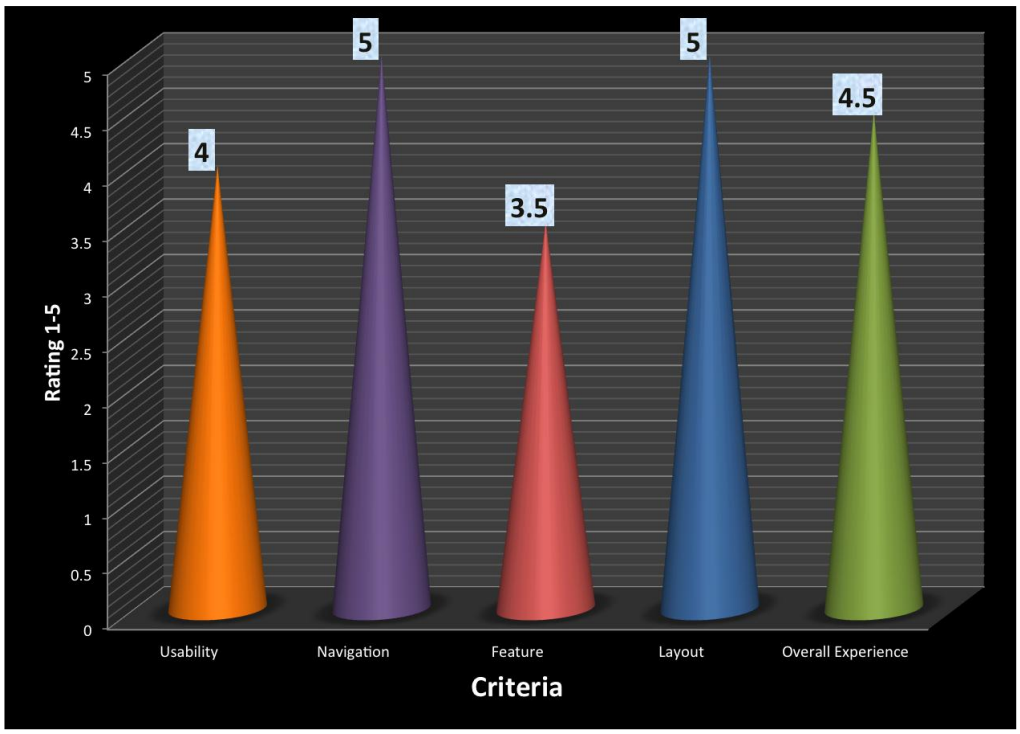


Figure 16: User's rating for different aspects of the Application

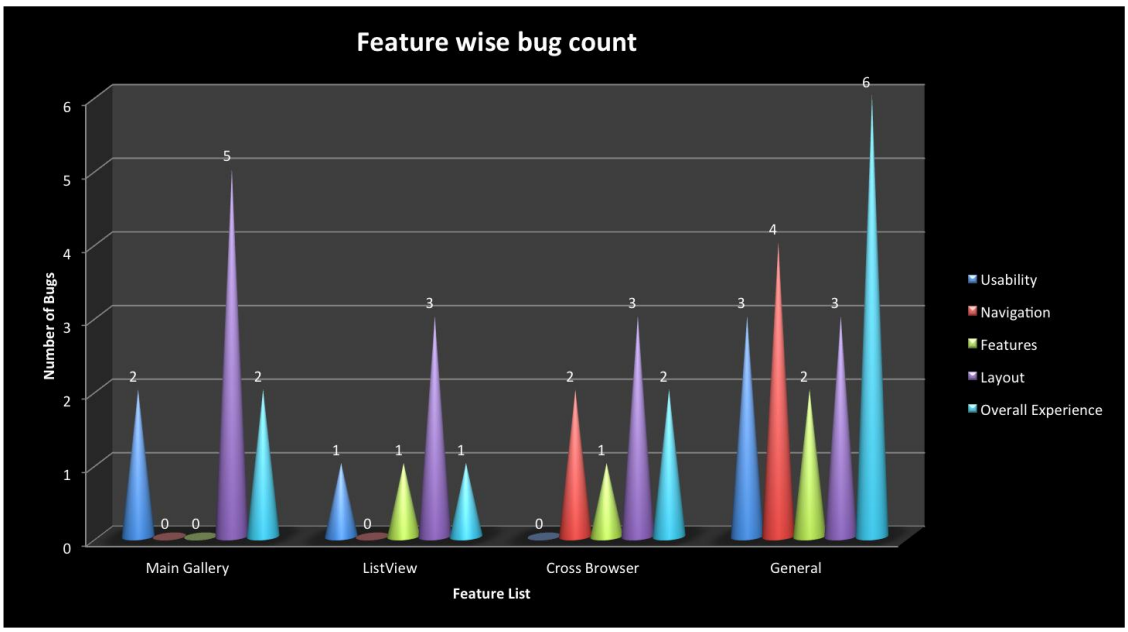


Figure 17: Feature Wise Bug Count for the Application

As the UAT was performed while the application was still under development, a number of bugs were identified and rectified, as a result of the User Acceptance Testing. Although, the overall feedback was generally positive, overwhelming number of people wanted the search feature implemented in the application, for them to consider using the application. Also, the overall application experience varied with the internet signal strength and speed.

7. Lessons Learnt/Reflection

7.1 Reasons for project's success

At the start of the year, I was unsure about the direction of the project and how it might possibly turn out. Over the months, I realised that although, the project itself wasn't very complex, the technical learning curve was huge, due to the unfamiliarity in server side programming, and its interaction with the application.

Late last year, when I approached Dr. Lexing, I had some experience in mobile application development, through previous applications that I had built. However, I was never presented with an opportunity to develop a full system (server side and application development), and always considered myself as a front-end application developer.

In the hindsight, although, I had a few ups and downs with the project, I consider myself very fortunate with the thesis opportunity, I was presented with. This is partially because of the supervision I received, and the technical challenges that I faced. I believe, thesis provides an ideal opportunity for all students to apply and build upon the knowledge that they have gained over the years, while making a real difference to the world, by working on a real life project. In my case, I was able to apply engineering and project management skills that I have learnt, as part of my degree, while providing a new domain to access the video search algorithm, that could potentially revolutionise the way, we browse and search videos.

Previously, I have been part of a successful start-up company named, Imagine Team Pty. Ltd and have developed several applications for clients. In comparison, although the timeframe for the project was relatively small, provided a full time study load, I believe, the project has been a success, due to the following reasons:

<u>Supervision</u>
One of the things that I learnt from this project is that supervision is the key. Finding a supervisor, who is able to meet up weekly and provide feedback on the work that was undertaken for the previous week, not only ensures that you are on the right track, but also lets you clarify any doubts for the work that would be taken in the following week.
As a student, weekly meetings might feel tedious, as it means, showing progress every week, but in the long run, this has allowed me to deliver the application that I have, by completing small blocks of tasks on a regular basis.

Another aspect of the weekly meetings that I enjoyed was the meet-up session with other students undertaking individual projects under Dr. Lexing. Although, most of them were taking semester long projects, I enjoyed the conversation and the weekly update that each of us had to provide. This often resulted in a discussion through a question and answer session, where the person holding the Angry Bird toy, was allowed to speak. Essentially, this made the meetings, informal, enjoyable and a great learning experience.

Project Management

Another key aspect of any thesis is time and resource allocation. Unlike other courses at university, where there are pre-defined hours, a thesis project, is entirely under the control of a student, for better or for worse. For most students, controlling a project's direction is a good relief from the other courses with pre-defined reading, assignments and examinations. However, things can quickly pile up, if the project is not managed properly. As part of Engineering Management course, I learnt about the various aspects of managing a project effectively, which I applied to this project.

Some students argue that undergraduate honours thesis project is the hardest, primarily due to a difference in the expectations of the supervisor and the student as well as incorrect analysis of the amount and complexity of work. Accounting for this is a key lesson in project management, which I experienced and learnt from, first hand, in my project.

Communication

Communication is the key. After talking with my peers, I have realised how busy some of the supervisors really are. For a few of my colleagues, the only form of communication is via emails. As such, provided the little time that they have with their supervisors for feedback, clear communication and presentation is vital.

In my thesis, I was required to provide weekly updates on my thesis, to the group of students undertaking individual projects under Dr. Lexing. For the first few weeks, I had to repeat my topic, and the progress that I had made, in a time span of 10 minutes or less, as the meetings were brief. As a result of the discussion and the verbal feedback received from my peers, I felt confident in pitching my idea, which assisted me in my mid-semester review presentation.

Another example of clear communication was for setting up the meeting times with my supervisor. The regular communication via email ensured that meetings times were set in advance, and any last minute changes in meetings, were reflected across to all involved parties.

Constructive Feedback

In the mid-semester review presentation, I received a grade of 6.5/10, which was below the average grade for the class. Although, I was left disappointed with the grade, I was happy with the feedback that I received from my supervisor.

In the presentation, my high level diagram was incomplete (in fact incorrect on some levels), and didn't portray the whole picture of the work that was needed to be done. Essentially, some of the communication channels across different modules on the server side, were either incomplete or incorrect. This was partly because of my lack of knowledge in servers and the lack of research in the topic, on my end.

Although this mistake costed me a few marks, I ended up rectifying the problem, sooner rather than later. Initially I wasn't sure, as to why the diagram was incorrect, but now comparing my current high level diagram with the previous, I can clearly see how I had misrepresented it.

Motivation

'Keep trying, even if you fail' has been the motto of my parents, and it is especially applicable in my thesis. There have been several moments during my thesis, where I have found things a little challenging, and have been stuck for days, with no solution. However, I have realised that pursuing answers through various domains, will eventually lead to a solution.

As an example, one of the technical challenges that I faced was, understanding where exactly JSON fits in, with the rest of the application. I knew nothing about the JSON format previously, and found it very difficult grasping my head around a very simple concept.

To understand the problem, I browsed technical forums, but never really understood the complete picture, until I browsed through YouTube videos, that explained the JSON concept, from ground up.

Software Engineering Principles

Coming into the project, I was under the impression that the documentation involved with a project is primarily done for the sake of documentation, as has been the case with some of the previous assignments.

However, provided the requirements of the project and my previously limited technical understanding in the required area, I can now see the true benefit of documenting and applying the software engineering principles, to a previously unknown project.

By taking time to understand the requirements of the project, I believe, I have saved a lot of time, during my late, but rather fast, code implementation. By following the coding standards and best practices in software engineering, such as modular approach, clear commenting etc., I have significantly reduced my time in testing and debugging the application. In fact, due to the modular approach, additional features can be easily integrated with the existing solution.

7.2 Things that I have learnt from the project

It has been over 9 months, since the project began, and I feel I have come a long way, by the end of its completion. Over the time of the project, I have developed and built upon several skills that will be very beneficial in my career ahead. These skills are:

Technical Improvement

With no prior knowledge of server-side programming, I feel much more confident in my technical abilities in developing server side scripts and its communication with the front-end applications. The biggest challenge was finding a starting point, for understanding the basics of server-side and python programming.

On the front-end, I was required to develop a cross-browser, as well as a set of galleries containing the videos for each category. The challenge was placing the galleries in a scroll bar, such that the galleries could be scrolled through, vertically. This tested my abilities in developing and delivering the front end to meet the client's needs. Although, I initially struggled, after some research and applying my prior knowledge in front-end Android development, I was able to provide the required feature.

Time Management

With a full work-load of 4 courses and no final exams, completing the project has been extremely challenging, with all the assessments in the other courses, also due in October. This was slightly unplanned, as one of the courses that I was enrolled in, was changed to an intensive course spanning across a month, in the last minute. In the end, I am glad, all the careful planning, in completing each of the assessments, one by one, has paid off.

Communication

Communication skill is also something that I have improved upon. Although, this was one of the factors for the project's success, I initially found it hard conveying some of the concepts, specific to Android platform, to the general audience. This is especially the case, during the mid-semester presentation, where it is assumed that the audience has no background knowledge in your area. Explaining a 'listview' and a 'gallery' to them, has allowed me to direct my pitch towards my audience.

Documentation

I have learnt the value of documentation especially during handover phase. David Hehir completed his project last year. However, installing the required programs and software, for his executable to run, was tedious, and something that had to be followed using his provided documentation, step-by-step. It also listed the exact input parameters (as well as the specified format for the inputs) that the executable required, alongside the order in which the scripts had to be run. This made it easy for me to execute and integrate his program's output into my project. The importance of documenting in enough depth, that will make it easy for the next person to use your code, is a skill I have learnt as part of this project.

Thesis projects provide an ideal stepping stone for students who wish to further pursue their career in Masters or Ph.D., and allow them to experience the required needs and expectations of a research project. For me, the project has been a great experience, and I have gained experience in several areas, which will be beneficial to me, regardless of which career path I choose.

Thesis projects provide an ideal stepping stone for students who wish to further pursue their career in Masters or Ph.D., and allow them to experience the required needs and expectations of a research project. For me, the project has been a great experience, and I have gained experience in several areas, which will be beneficial to me, regardless of which career path I choose.

8. **Deliverables and Future Work**

8.1 Deliverables

As part of the project handover, the following deliverables have been agreed to:

GitHub: A copy of the Android code placed on GitHub (place for storing open-source projects) for users to access. This will include:

- Readme file for compiling the code (including a list of required software)
- Doxygen files, for understanding the layout and structure of the code.
- Android Code

Server-side scripts already exist on Cantabile server. A Readme file will be added, to allow others to locate the script directories and what each script does.

8.2 Future Work

Although the project requirements were met in the provided timeframe, further development in the following areas of the project, will allow it to be better suited for future needs:

Feature Additions

The application, although functional, is only capable of providing pre-populated categories, for the users to browse through. This is not an issue, if the application is designed to work as a proof of concept. However, provided the potential of the algorithm and its impact on the way, users will browse videos, it is very likely for the application to become main-stream very quickly. As such, a search feature, to allow users to add their own categories, should be integrated into the application.

Designing the front-end for it, would be relatively easy. The back-end, however, would require certain scripts to be written, to accept user commands from the application (POST method over HTTP protocol). Also, the entire procedure of providing matching key-frames for a specific category, currently requires several hours (at least 5-6 hours) to process, as it has to download the metadata and the videos, split the videos and match the key frames. As such, optimisation is also needed in the algorithm, to provide the results quickly.

On the front end, currently the application retrieves all the data before loading. As such, it can take a significant amount of time, before the application loads, for slow internet speeds. With the implementation of a feature called 'lazy loading', the application can start, while the individual features of the application are loaded. This will make the application much more user-friendly and accessible in areas with slow internet speeds.

The images are currently retrieved through a public directory on the Cantabile server. However, from a privacy and security aspect, it would be better if the images are placed in a secure directory requiring authentication. By implementing this, access to the data from within the application can be restricted to the relevant parties.

Scalability

Back-end currently, is also not capable of catering for large amounts of queries, and as such, would also require upgrading, to handle several simultaneous search requests, at once. This would also require a reliable source for downloading videos, in large quantities. Currently, only a limited number of videos can be downloaded through YouTube at a given time, before it asks for subscription.

iOS and Tablet Application Development

The current application has been primarily designed for use on Android smartphones. With growing number of tablet users, and with half the smartphone market, currently owned by Apple, developing native applications for the iOS platform (iPhone and iPad), as well as Android Tablets, that connect with the existing back-end should be developed. This will cater for most of the existing smartphone market.

9. Conclusion

The goal of the thesis was to utilise the available 'visual meme tracking' tools and provide a new platform for the users to engage through smartphones. Over a period of 9 months, an Android application was formulated, designed and developed, allowing users to view and browse available categories, search matching key-frames and videos, on their smartphones. For the retrieval of data in the application, a database was designed and developed on the Cantabile server.

The project requirements as initially outlined in the report have been successfully met. However, there is still a large scope for improvement, even in the limited functionality, that it currently offers. Undoubtedly, one of the achievements has been the Application's User Interface. With the addition of the search functionality in the future, to allow users to add their own categories, and the lazy loading option for loading individual components, the application can be catered for a much wider audience.

In terms of thesis, the project has been a success for both the client and the student. For the client, an application matching the initial requirements of the project has been delivered. For the student, with little technical background, the opportunity to apply software engineering and project management, and successfully delivering the completed project on time has been a successful outcome.

Bibliography

Report References:

[1] Code Style Guidelines for Contributors | Android Open Source. 2012. Code Style Guidelines for Contributors | Android Open Source. [ONLINE] Available at: http://source.android.com/source/code-style.html. [Accessed 12 October 2012].

[2] Doxygen - Wikipedia, the free encyclopedia. 2012. Doxygen - Wikipedia, the free encyclopedia. [ONLINE] Available at: http://en.wikipedia.org/wiki/Doxygen. [Accessed 12 October 2012].

[3] Integration Testing . 2012. Integration Testing . [ONLINE] Available at: http://msdn.microsoft.com/en-us/library/aa292128(v=vs.71).aspx. [Accessed 14 October 2012].

[4] Java coding standards and Javadoc style comments. 2012. Java coding standards and Javadoc style comments. [ONLINE] Available at: http://www2.hawaii.edu/~tp_200/ics111/material/codingStandards.html. [Accessed 12 October 2012].

[5] JSONLint - The JSON Validator. . 2012. JSONLint - The JSON Validator. . [ONLINE] Available at: http://www.jsonlint.com. [Accessed 25 October 2012].

[6] London Summer Olympics 2012 - Intro - YouTube. 2012. London Summer Olympics 2012 - Intro - YouTube. [ONLINE] Available at: http://www.youtube.com/watch?v=2DY4IeVRZUA. [Accessed 25 July 2012].

[7] playstation - YouTube. 2012. playstation - YouTube. [ONLINE] Available at: http://www.youtube.com/results?search_query=playstation&oq=playstation&gs_l=youtube.3..0l10.90914.92356.0.92503.11.7.0.4.4.0.150.491.4j2.6.0...0.0...1ac.1.tHPB7Zu0W7o. [Accessed 25 July 2012].

[8] System testing - Wikipedia, the free encyclopedia. 2012. System testing - Wikipedia, the free encyclopedia. [ONLINE] Available at: http://en.wikipedia.org/wiki/System_testing#cite_note-ieee-0. [Accessed 18 October 2012].

[9] Tracking Viral Videos on YouTube: Tool Set Construction - David Hehir [PDF] Available at: http://cs.anu.edu.au/student/projects/11S2/Reports/David%20Hehir.pdf. [Accessed 15 July 2012].

[10] Unit Testing | Software Testing Fundamentals. 2012. Unit Testing | Software Testing Fundamentals. [ONLINE] Available at: http://softwaretestingfundamentals.com/unit-testing/. [Accessed 23 October 2012].

[11] Viral Video Legends Compilation - 100 Greatest Internet Videos In 3 Minutes - YouTube. 2012. Viral Video Legends Compilation - 100 Greatest Internet Videos In 3 Minutes - YouTube. [ONLINE] Available at: http://www.youtube.com/watch?v=LkOnsIhIcu8. [Accessed 25 July 2012].

[12] Watch any YouTube Video as a Series of Still Images. 2012. Watch any YouTube Video as a Series of Still Images. [ONLINE] Available at: http://www.labnol.org/internet/youtube-image-frames/24608/. [Accessed 27 October 2012].

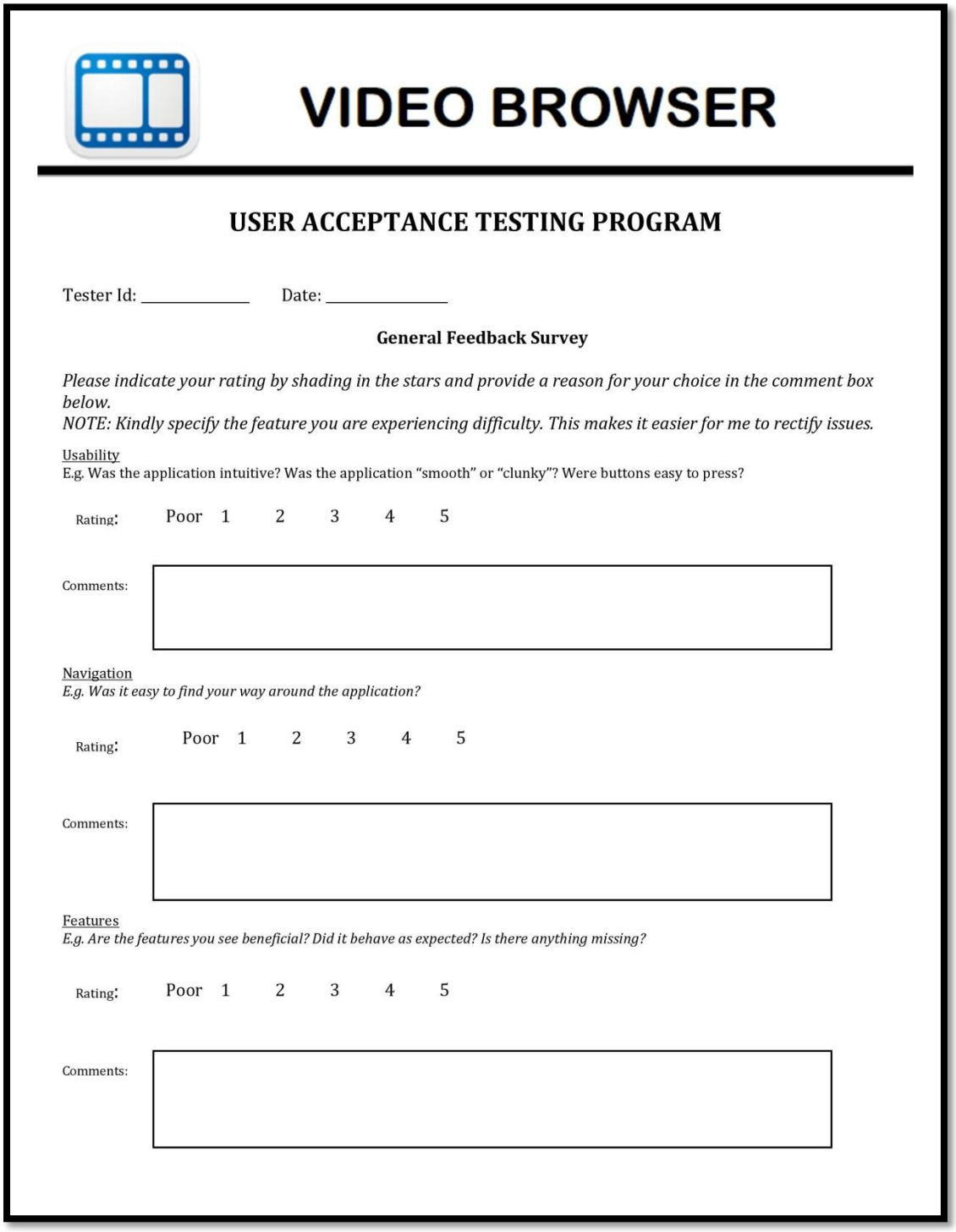
[13] Xwo's Video Roundup Thumbnail Image [Request]. 2012. Xwo's Video Roundup Thumbnail Image [Request]. [ONLINE] Available at: http://jumprs.org/forum/showthread.php?523-Xwo-s-Video-Roundup-Thumbnail-Image-Request. [Accessed 22 October 2012].

[14] YouTube. 2012. *YouTube*. [ONLINE] Available at:http://www.youtube.com/t/press_statistics. [Accessed 18 October 2012].

Appendix

Section A – UAT Testing

As part of testing, the following form was used for User Acceptance Testing.



VIDEO BROWSER

USER ACCEPTANCE TESTING PROGRAM

Tester Id: _____ Date: _____

General Feedback Survey

Please indicate your rating by shading in the stars and provide a reason for your choice in the comment box below.
NOTE: Kindly specify the feature you are experiencing difficulty. This makes it easier for me to rectify issues.

Usability
E.g. Was the application intuitive? Was the application "smooth" or "clunky"? Were buttons easy to press?

Rating: Poor 1 2 3 4 5

Comments:

Navigation
E.g. Was it easy to find your way around the application?

Rating: Poor 1 2 3 4 5

Comments:

Features
E.g. Are the features you see beneficial? Did it behave as expected? Is there anything missing?

Rating: Poor 1 2 3 4 5

Comments:

Figure A: User Acceptance Testing (Page 1)

Layout

E.g. Was the text too big or too small? Were images too big or too small? Was there any wastage of screen real estate?

Rating: Poor 1 2 3 4 5 Excellent

Comments:

Overall Experience

How would you rate your experience of the application?

Rating: Poor 1 2 3 4 5 Excellent

Comments:

Figure B: User Acceptance Testing (Page 2)

Section B – Attached Code

Compilation Guide

For the development of the Android Application and for connecting with the Cantabile server, Windows Operating System was used. As such, the required software guide is primarily targeted at Windows Operation Systems. However, similar software can be used for other operating systems.

Android Code

Software needed for accessing and compilation of Android code:

1. Eclipse
2. Java SDK
3. Android SDK

Hardware needed for the compilation of Android code:

1. Android Smartphone with a firmware of Android 2.3 or higher. It is assumed that the phone has had necessary drivers installed on the PC for it to be detected. Also the phone needs to be in the Debug Mode. The steps required for activation of debug mode are model dependent, and can be found online on Google.

Eclipse is an Integrated Development Environment (IDE), and Java SDK allows access to native Java libraries for use in Android application. Both Eclipse and Java SDK can be downloaded together through one link: http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/junosr1

An official installation guide for the Android SDK can be found here: http://developer.android.com/sdk/installing/index.html

For importing and compiling the project:

1. Download the Project on your PC
2. Open Eclipse IDE
3. File -> Import -> Android -> Existing Android Code Into Workspace
4. In the root directory, browse to the source of the project
5. Connect the phone and press Run (This should run the application on your phone)

Server-side Scripts

Note: It is assumed that the required scripts are available on the server and that the user has been granted the appropriate access on the server, for viewing the files.

Software needed for accessing python scripts:

1. Putty
2. WinSCP

Putty provides SSH access to the Cantabile Server. The latest copy of Putty can be downloaded from: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
WinSCP provides SFTP access to the server. The latest copy of WinSCP can be downloaded from: http://winscp.net/eng/download.php

The path to access the written scripts is as follows:

createdb.py: data/u4545909/crawling/createdb.py
keyFramesJSONOutput.py: data/u4545909/www/cgi-bin/keyFramesJSONOutput.py
metadataJSONOutput.py: data/u4545909/www/cgi-bin/metadataJSONOutput.py
mKeyFramesJSONOutput.py: data/u4545909/www/cgi-bin/KeyFramesJSONOutput.py
mVideosJSONOutput.py: data/u4545909/www/cgi-bin/ mVideosJSONOutput.py

Please Note: Starting from the next page, the code generated as part of the project has been appended. As the PDF for the codes were generated externally, the page numbering system does not match the page numbering system of the report.

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.video_browser_thesis.activities"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="8" />

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:screenOrientation="portrait"
        android:theme="@android:style/Theme.Light.NoTitleBar" >
        <activity
            android:name=".MainGalleryActivity"
            android:label="@string/app_name"
            android:screenOrientation="portrait" >
        </activity>
        <activity

android:name="com.video_browser_thesis.splashScreen.SplashScreen"
            android:screenOrientation="portrait" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".KeyframeListActivity"
android:screenOrientation="portrait" />
        <activity android:name=".CrossBrowserActivity"
android:screenOrientation="portrait">

            </activity>
    </application>

</manifest>
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

public class CB_GalleryProcessing implements Processing{

    private String base_url;
    private String url;

    private static final String TAG_MATCHING_KEYFRAME =
"matching_key_frames";
    private static final String TAG_MATCHING_IMAGE_SOURCE=
"MatchingFrameURL";

    KeyFrame_attributes[] k_attr;

    public CB_GalleryProcessing(String attach_url){
        base_url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
mKeyFrameJSONOutput.py?q=";
        url = base_url + urlProcessing(attach_url);
        setArrayContents();
    }

    @Override
    public void setArrayContents() {
        // TODO Auto-generated method stub
        ArrayList<HashMap<String, String>> temp_category_list =
jsonFeed();
        KeyFrame_attributes kf_attr[] = new
KeyFrame_attributes[temp_category_list.size()];

        for(int i=0; i<temp_category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURC
E).toString());

            kf_attr[i] = new KeyFrame_attributes(

temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURCE).toString(),
```

```
temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURCE).substring(97,
104),
                    ip.decodeImage());
        }

        k_attr = kf_attr;
    }

    @Override
    public ArrayList<HashMap<String, String>> jsonFeed() {
        // TODO Auto-generated method stub
        JSONArray category = null;
        ArrayList<HashMap<String, String>> keyframe_list = new
ArrayList<HashMap<String, String>>();

        /**
         * Creating JSON Parser instance
         */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String image_source;
        HashMap<String, String> map;

        /**
         * Getting JSON string from URL
         */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /**
             * Getting Array of Contacts
             */
            category = json.getJSONArray(TAG_MATCHING_KEYFRAME);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /**
                 * Storing each json item in variable
                 */
                image_source = c.getString(TAG_MATCHING_IMAGE_SOURCE);

                /** Creating new HashMap */
                map = new HashMap<String, String>();
```

```
                /**
                 * Adding each child node to HashMap key => value
                 */
                map.put(TAG_MATCHING_IMAGE_SOURCE, image_source);

                /**
                 * Adding HashList to ArrayList
                 */
                keyframe_list.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return keyframe_list;
    }

    public KeyFrame_attributes[] getKeyFrameArray(){
        return k_attr;
    }

    public String urlProcessing(String url){
        String temp = "http://cantabile.anu.edu.au/memexplore/";
        temp = url.substring(temp.length(),url.length());
        System.out.println(temp);
        return temp;
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

public class CB_GalleryProcessing implements Processing{

    private String base_url;
    private String url;

    private static final String TAG_MATCHING_KEYFRAME =
"matching_key_frames";
    private static final String TAG_MATCHING_IMAGE_SOURCE=
"MatchingFrameURL";

    KeyFrame_attributes[] k_attr;

    public CB_GalleryProcessing(String attach_url){
        base_url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
mKeyFrameJSONOutput.py?q=";
        url = base_url + urlProcessing(attach_url);
        setArrayContents();
    }

    @Override
    public void setArrayContents() {
        // TODO Auto-generated method stub
        ArrayList<HashMap<String, String>> temp_category_list =
jsonFeed();
        KeyFrame_attributes kf_attr[] = new
KeyFrame_attributes[temp_category_list.size()];

        for(int i=0; i<temp_category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURC
E).toString());

            kf_attr[i] = new KeyFrame_attributes(

temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURCE).toString(),
```

```
temp_category_list.get(i).get(TAG_MATCHING_IMAGE_SOURCE).substring(97,
104),
                    ip.decodeImage());
        }

        k_attr = kf_attr;
    }

    @Override
    public ArrayList<HashMap<String, String>> jsonFeed() {
        // TODO Auto-generated method stub
        JSONArray category = null;
        ArrayList<HashMap<String, String>> keyframe_list = new
ArrayList<HashMap<String, String>>();

        /**
         * Creating JSON Parser instance
         */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String image_source;
        HashMap<String, String> map;

        /**
         * Getting JSON string from URL
         */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /**
             * Getting Array of Contacts
             */
            category = json.getJSONArray(TAG_MATCHING_KEYFRAME);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /**
                 * Storing each json item in variable
                 */
                image_source = c.getString(TAG_MATCHING_IMAGE_SOURCE);

                /** Creating new HashMap */
                map = new HashMap<String, String>();
```

```
                /**
                 * Adding each child node to HashMap key => value
                 */
                map.put(TAG_MATCHING_IMAGE_SOURCE, image_source);

                /**
                 * Adding HashList to ArrayList
                 */
                keyframe_list.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return keyframe_list;
    }

    public KeyFrame_attributes[] getKeyFrameArray(){
        return k_attr;
    }

    public String urlProcessing(String url){
        String temp = "http://cantabile.anu.edu.au/memexplore/";
        temp = url.substring(temp.length(),url.length());
        System.out.println(temp);
        return temp;
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

import java.util.HashMap;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import com.video_browser_thesis.elements.KeyFrame_attributes;
import com.video_browser_thesis.json.JSONParser;

public class CB_Processing implements Processing{

    private String base_url;
    private String url;

    private static final String TAG_KEYFRAME = "key_frame";
    private static final String TAG_IMAGE_SOURCE= "ImageSource";
    private static final String TAG_NAME = "Name";

    KeyFrame_attributes[] k_attr;

    public CB_Processing(String attach_url){
        base_url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
keyFramesJSONOutput.py?q=";
        url = base_url + attach_url;
        setArrayContents();
    }

    /**
     * Organizes the data received from the URL in JSON format into a
HashMap
     *
     * @param  categoryList     Hash Map which contains all the
segregated metadata under different tag names
     * @param  jParser          An instance of JSONParser which parses
data from the provided URL
     * @param  c                An instance JSONObject which stores
all data contained within that JSON Object
```

```
     * @param  id               A temporary array to store video ids
     * @param  image_source     A temporary array to store video
images
     * @param  video_url        A temporary array to store video url
     * @param  video_name       A temporary array to store video name
     * @return categoryList
     * @see
     */
    public ArrayList<HashMap<String, String>> jsonFeed(){

        JSONArray category = null;
        ArrayList<HashMap<String, String>> keyframe_list = new
ArrayList<HashMap<String, String>>();

        /**
         * Creating JSON Parser instance
         */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String image_source,video_name;
        HashMap<String, String> map;

        /**
         * Getting JSON string from URL
         */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /**
             * Getting Array of Contacts
             */
            category = json.getJSONArray(TAG_KEYFRAME);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /**
                 * Storing each json item in variable
                 */
                image_source = c.getString(TAG_IMAGE_SOURCE);
                video_name = c.getString(TAG_NAME);

                /** Creating new HashMap */
                map = new HashMap<String, String>();
```

```
                /**
                 * Adding each child node to HashMap key => value
                 */
                map.put(TAG_IMAGE_SOURCE, image_source);
                map.put(TAG_NAME, video_name);

                /**
                 * Adding HashList to ArrayList
                 */
                keyframe_list.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return keyframe_list;
    }

    /**
     * Creates a KeyFrame_attributes object by extracting the relevant
data from the HashMap
     *
     * @param  kf_attr      An instance of KeyFrame_attributes object
     * @return kf_attr
     * @see
     */
    public void setArrayContents(){

        ArrayList<HashMap<String, String>> temp_category_list =
jsonFeed();
        KeyFrame_attributes kf_attr[] = new
KeyFrame_attributes[temp_category_list.size()];

        for(int i=0; i<temp_category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(temp_category_list.get(i).get(TAG_IMAGE_SOURCE).toStri
ng());

            kf_attr[i] = new KeyFrame_attributes(

temp_category_list.get(i).get(TAG_IMAGE_SOURCE).toString(),

temp_category_list.get(i).get(TAG_NAME).substring(32, 38),
                    ip.decodeImage());
```

```
            }

            k_attr = kf_attr;
        }

        public KeyFrame_attributes[] getKeyFrameArray(){
            return k_attr;
        }
}
```

```
/*
 * Copyright (C) 2012 Mobs and Geeks

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.view.View;
/**
 * Constructs a {@linkplain CircularListAdapter}.
 *
 * @param listAdapter A {@link Keyframe_ListAdapter} that has to
behave circular.
 */
public class CircularListAdapter extends BaseAdapter {

    static final boolean DEBUG = false;
    static final String TAG =
CircularListAdapter.class.getSimpleName();

    private BaseAdapter mListAdapter;
    private int mListAdapterCount;

    public CircularListAdapter(BaseAdapter listAdapter) {
        if(listAdapter == null) {
            throw new IllegalArgumentException("listAdapter cannot be
null.");
        }

        this.mListAdapter = listAdapter;
        this.mListAdapterCount = listAdapter.getCount();
```

```
    }

    @Override
    public int getCount() {
        return Integer.MAX_VALUE;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        return mListAdapter.getView(position % mListAdapterCount,
convertView, parent);
    }

    @Override
    public Object getItem(int position) {
        return mListAdapter.getItem(position % mListAdapterCount);
    }

    @Override
    public long getItemId(int position) {
        return mListAdapter.getItemId(position % mListAdapterCount);
    }

    @Override
    public boolean areAllItemsEnabled() {
        return mListAdapter.areAllItemsEnabled();
    }

    @Override
    public int getItemViewType(int position) {
        return mListAdapter.getItemViewType(position %
mListAdapterCount);
    }

    @Override
    public int getViewTypeCount() {
        return mListAdapter.getViewTypeCount();
    }

    @Override
    public boolean isEmpty() {
        return mListAdapter.isEmpty();
    }
```

```
    @Override
    public boolean isEnabled(int position) {
        return mListAdapter.isEnabled(position % mListAdapterCount);
    }

    @Override
    public void notifyDataSetChanged() {
        mListAdapter.notifyDataSetChanged();
        mListAdapterCount = mListAdapter.getCount();

        super.notifyDataSetChanged();
    }

    @Override
    public void notifyDataSetInvalidated() {
        mListAdapter.notifyDataSetInvalidated();
        super.notifyDataSetInvalidated();
    }

    @Override
    public View getDropDownView(int position, View convertView,
ViewGroup parent) {
        return mListAdapter.getDropDownView(position %
mListAdapterCount,
                convertView, parent);
    }

    @Override
    public boolean hasStableIds() {
        return mListAdapter.hasStableIds();
    }

}
```

```
/*
 * Copyright (C) 2012 Anshul Saini


 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a CrossBrowserGallery Video Adapter which extends a Base
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class CrossBrowser_GalleryAdapter extends BaseAdapter {

    private Activity activity;
    private static LayoutInflater inflater=null;
    KeyFrame_attributes keyframe_attr[] = null;

    public CrossBrowser_GalleryAdapter(Activity
a,KeyFrame_attributes[] keyframe_attr) {
        activity = a;
        inflater =
(LayoutInflater)activity.getSystemService(Context.LAYOUT_INFLATER_SERV
ICE);
        this.keyframe_attr = keyframe_attr;
    }
```

```
    public int getCount() {
        return keyframe_attr.length;
    }

    public Object getItem(int position) {
        return position;
    }

    public long getItemId(int position) {
        return position;
    }

    /**
     * Inflates the gallery item with different types of views.
     */
    public View getView(int position, View convertView, ViewGroup
parent) {
        View vi=convertView;
        ViewHolder holder;
        if(convertView==null){
            vi = inflater.inflate(R.layout.cross_browser_galleryitem,
null);
            holder=new ViewHolder();

holder.video_image=(ImageView)vi.findViewById(R.id.cross_browsers_gall
ery_img);

holder.video_length=(TextView)vi.findViewById(R.id.cb_gallery_video_le
ngth);

            vi.setTag(holder);
        }
        else{
            holder=(ViewHolder)vi.getTag();
        }

        KeyFrame_attributes v_attr = keyframe_attr[position];
        holder.video_length.setText(v_attr.frame_point);

        if(v_attr.video_img != null){
            holder.video_image.setImageBitmap(v_attr.video_img);
        }

        return vi;
    }
```

```
    /**
     * Holds all the elements of the view created in its super class
     * @param video_length  Video length in TextView format
     * @param video_image   Video image in ImageView format
     */
    public static class ViewHolder{
        public ImageView video_image;
        public TextView video_length;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a Keyframe KeyFrame Adapter which extends a Custom List
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class CrossBrowser_ListAdapter extends
ArrayAdapter<KeyFrame_attributes>{

    Context context;
    static int layoutResourceId = R.layout.cross_browser_listitem;
    KeyFrame_attributes keyframe_attr[] = null;

    public CrossBrowser_ListAdapter(Context context,
KeyFrame_attributes[] keyframe_attr) {
        super(context, layoutResourceId,keyframe_attr);
        // TODO Auto-generated constructor stub
        this.context = context;
        this.keyframe_attr = keyframe_attr;
    }

    /**
```

```
     * Inflates the gallery item with different types of views.
     */
    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        View row = convertView;
        CrossBrowserListHolder holder = null;

        if(row == null)
        {
            LayoutInflater inflater =
((Activity)context).getLayoutInflater();
            row = inflater.inflate(layoutResourceId, parent, false);

            holder = new CrossBrowserListHolder();
            holder.video_img =
(ImageView)row.findViewById(R.id.cross_browsers_img);
            holder.video_frame_point =
(TextView)row.findViewById(R.id.cb_video_length);

            row.setTag(holder);
        }
        else{
            holder = (CrossBrowserListHolder)row.getTag();
        }

        KeyFrame_attributes kf_attributes = keyframe_attr[position];
        holder.video_img.setImageBitmap(kf_attributes.video_img);
        holder.video_frame_point.setText(kf_attributes.frame_point);

        return row;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_frame_point Video frame point in TextView format
     * @param video_img         Video image in ImageView format
     */
    static class CrossBrowserListHolder{
        ImageView video_img;
        TextView video_frame_point;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.video_browser_thesis.activities;

import android.app.Activity;

/**
 * Displays all the matching key frames of a video
 * @author Anshul Saini
 * @version 5
 */

public class CrossBrowserActivity extends Activity{

    public static Boolean i = false;
    Bundle cb_bundle;

    private  Gallery crossbrowser_gallery;
    private DisplayMetrics metrics = new DisplayMetrics();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cross_browser);

        /**
         *  Retrieves data sent by the previous activity
```

```
         */
        cb_bundle = getIntent().getExtras();

        /**
         * Setting up the gallery view positioning
         * */
        getWindowManager().getDefaultDisplay().getMetrics(metrics);
        crossbrowser_gallery =
(Gallery)findViewById(R.id.cross_browser_gallery);
        MarginLayoutParams mlp1 = (MarginLayoutParams)
crossbrowser_gallery.getLayoutParams();
        setMargins(mlp1);

        CB_Processing cb_p = new
CB_Processing(cb_bundle.getString("video_id"));
        final KeyFrame_attributes[] keyframe_attr =
cb_p.getKeyFrameArray();

        final ListView crossbrowser_list = (ListView)
findViewById(R.id.cross_browser_listview);
        final ImageView selected_img =
(ImageView)findViewById(R.id.selected_matching_keyframe);

        CrossBrowser_ListAdapter cb_listAdapter = new
CrossBrowser_ListAdapter(this, keyframe_attr);
        final CircularListAdapter circularListAdapter = new
CircularListAdapter(cb_listAdapter);
        crossbrowser_list.setAdapter(circularListAdapter);
        crossbrowser_list.setOnItemClickListener(new
OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> arg0, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub
                CB_GalleryProcessing cb_gall = new
CB_GalleryProcessing(keyframe_attr[position].image_source);
                final KeyFrame_attributes[] keyframe_gall =
cb_gall.getKeyFrameArray();
                crossbrowser_gallery.setAdapter(new
CrossBrowser_GalleryAdapter(CrossBrowserActivity.this,keyframe_gall));
                crossbrowser_gallery.setSelection(1);

                crossbrowser_gallery.setOnItemClickListener(new
```

```
OnItemClickListener() {

                    @Override
                    public void onItemClick(AdapterView<?> arg0, View
arg1, int position_gall,
                            long arg3) {
                        // TODO Auto-generated method stub
                        String temp =
keyframe_gall[position_gall].image_source.substring(53,
keyframe_gall[position_gall].image_source.length());
                        String temp_video_id = temp.substring(0,
temp.indexOf("/"));
                        String temp_base_url = "http://
www.youtube.com/watch?v=";
                        String full_url = temp_base_url +
temp_video_id;
                        startActivity(new Intent(Intent.ACTION_VIEW,
Uri.parse(full_url)));
                    }
                });

                /**
                 * Setting Selected Video
                 */
                ImageProcessing processing_selected_img = new
ImageProcessing(keyframe_attr[position].image_source);

selected_img.setImageBitmap(processing_selected_img.decodeImage());
            }
        });

        /**
         * Sets the default content of the gallery
         */
        KeyFrame_attributes[] defaultList = new
KeyFrame_attributes[5];
        for(int i=0;i<5;i++){
            defaultList[i] = new KeyFrame_attributes(null,"",null);
        }
        crossbrowser_gallery.setAdapter(new
CrossBrowser_GalleryAdapter(this, defaultList));
        crossbrowser_gallery.setSelection(1);

    }
```

```
    /**
     * Sets the margins of top, bottom, right and left sides of the
galleryView
     */
    private void setMargins(MarginLayoutParams mlp){
        mlp.setMargins((metrics.widthPixels / 2 - 210), mlp.topMargin
+300,mlp.rightMargin, mlp.bottomMargin);
    }

    /**
     * Ends the current activity and returns to the previous activity
     */
    public void onListView_btn_click(View v){
        finish();
    }

    /**
     * Ends the current activity and returns to the main activity
     */
    public void onHomeButtonClick(View v){
        i = true;
        finish();
    }

    public String getValueFromBundle(){
        return cb_bundle.getString("video_id");
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.io.IOException;

import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

public class ImageProcessing {

    Bitmap bitmap_image;
    String imageUrl;

    public ImageProcessing(String imageUrl) {
        this.imageUrl = imageUrl;
        fetchImage();
    }

    /**
     * Fetches the image content from the provided URL
     *
     * @param bitmap    Stores the decoded image retrieved from the
provided url
     * @return bitmap
     * @see Bitmap Image
     * */
    private void fetchImage(){
        Bitmap bitmap = null;
        try {
              bitmap = BitmapFactory.decodeStream((InputStream)new
URL(imageUrl).getContent());
            } catch (MalformedURLException e) {
              e.printStackTrace();
            } catch (IOException e) {
              e.printStackTrace();
            }
        bitmap_image = bitmap;
    }
```

```
    public Bitmap decodeImage(){
        return bitmap_image;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.json;

import java.io.BufferedReader;

/**
 * Creates a JSON object from the data retrieved from the provided url
 * @author Anshul Saini
 * @version 5
 *
 * @param  is
 * @param  jObj     JSON Object which is finally created from the JSON
string
 * @param  json     JSON String which is used to create JSON Object

 * @return
 * @see
 */
public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    /**
     * Makes the HTTP request, parses the JSON String, creates and
JSON Object
     * @param httpClient
```

```
     * @param httpPost
     * @param httpResponse
     * @param httpEntity
     * @param reader
     * @param sb
     * @param line
     * @return jObj
     * @see JSONObject
     */
    public JSONObject getJSONFromUrl(String url) {

        DefaultHttpClient httpClient;
        HttpPost httpPost;
        HttpResponse httpResponse;
        HttpEntity httpEntity;
        BufferedReader reader;
        StringBuilder sb;
        String line;

        try {
            httpClient = new DefaultHttpClient();
            httpPost = new HttpPost(url);

            httpResponse = httpClient.execute(httpPost);
            httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            sb = new StringBuilder();
            line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
            is.close();
            json = sb.toString();
```

```
        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " +
e.toString());
        }

        try {
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " +
e.toString());
        }

        return jObj;
    }
}
```

```
package com.video_browser_thesis.elements;

import android.graphics.Bitmap;

/**
 * KeyFrame attributes
 * @author Anshul Saini
 * @version 5
 * @param image_source
 * @param frame_point
 * @param video_img
 *
 */
public class KeyFrame_attributes{
    public String image_source;
    public String frame_point;
    public Bitmap video_img;

    public KeyFrame_attributes(String image_source,String frame_point,
Bitmap video_img){
        this.image_source = image_source;
        this.frame_point = frame_point;
        this.video_img = video_img;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a Keyframe Video Adapter which extends a Custom List
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class Keyframe_ListAdapter extends
ArrayAdapter<Video_attributes>{

    Context context;
    static int layoutResourceId = R.layout.keyframe_list_layout;
    Video_attributes video_attr[] = null;

    public Keyframe_ListAdapter(Context context, Video_attributes[]
video_attr) {
        // TODO Auto-generated constructor stub
        super(context, layoutResourceId, video_attr);
        this.context = context;
        this.video_attr = video_attr;
    }

    /**
```

```
     * Inflates the gallery item with different types of views.
     */
    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        View row = convertView;
        VideoListHolder holder = null;

        if(row == null)
        {
            LayoutInflater inflater =
((Activity)context).getLayoutInflater();
            row = inflater.inflate(layoutResourceId, parent, false);

            holder = new VideoListHolder();
            holder.video_title =
(TextView)row.findViewById(R.id.video_title);
            holder.video_description =
(TextView)row.findViewById(R.id.video_description);
            holder.uploaded_time =
(TextView)row.findViewById(R.id.video_uploaded_timestamp);
            holder.view_count =
(TextView)row.findViewById(R.id.view_count);
            holder.video_author =
(TextView)row.findViewById(R.id.video_author);
            holder.video_length =
(TextView)row.findViewById(R.id.video_length);
            holder.video_img =
(ImageView)row.findViewById(R.id.video_image);

            row.setTag(holder);
        }
        else{
            holder = (VideoListHolder)row.getTag();
        }

        Video_attributes video_attributes = video_attr[position];
        holder.video_title.setText(video_attributes.video_title);
        holder.video_author.setText(video_attributes.video_author);
        holder.uploaded_time.setText(video_attributes.uploaded_time);
        holder.view_count.setText(video_attributes.view_count);
        holder.view_count.setText(video_attributes.view_count);
        holder.video_length.setText(video_attributes.length);

holder.video_description.setText(video_attributes.description);
```

```
        if(video_attributes.video_image!=null){

holder.video_img.setImageBitmap(video_attributes.video_image);
        }

        return row;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_title       Video title in TextView format
     * @param video_description Video description in TextView format
     * @param uploaded_time     Video image in TextView format
     * @param view_count        Video image in TextView format
     * @param video_author      Video image in TextView format
     * @param video_length      Video image in TextView format
     * @param video_img         Video image in ImageView format
     */
    static class VideoListHolder
    {
        TextView video_title;
        TextView video_description;
        TextView uploaded_time;
        TextView view_count;
        TextView video_author;
        TextView video_length;
        ImageView video_img;
    }
}
```

```
package com.video_browser_thesis.activities;

import android.app.Activity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;

import com.video_browser_thesis.adapters.Keyframe_ListAdapter;
import com.video_browser_thesis.elements.Video_attributes;
import com.video_browser_thesis.processing.ImageProcessing;

/**
 * Displays all the key frames of a video
 * @author Anshul Saini
 * @version 5
 */
public class KeyframeListActivity extends Activity{

    Bundle b;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_keyframe);

        /**
         * Test Data
         */
        Video_attributes test[] = new Video_attributes[10];
        for(int i=0; i<test.length;i++){
            test[i] = new Video_attributes("Anshul Bit My
Finger","Anshul",null,null,null,null,"10:15",null,"1 year","504","I am
Awesome. Watch me Roll !!!",null);
        }
        /**
         * End test data
         */
```

```
        ListView keyframe_list = (ListView)
findViewById(R.id.matching_keyframes_list);
        keyframe_list.setAdapter(new Keyframe_ListAdapter(this,test));

        b = getIntent().getExtras();
        TextView video_title =
(TextView)findViewById(R.id.video_title);
        video_title.setText(b.getString("video_title"));

        TextView video_description =
(TextView)findViewById(R.id.video_description);
        video_description.setText(b.getString("video_description"));

        TextView video_upload_time =
(TextView)findViewById(R.id.video_uploaded_timestamp);
        video_upload_time.setText(b.getString("video_upload_time"));

        TextView view_count = (TextView)findViewById(R.id.view_count);
        view_count.setText(b.getString("video_view_count"));

        TextView video_author = (TextView)findViewById(R.id.author);
        video_author.setText(b.getString("video_author"));

        TextView video_length =
(TextView)findViewById(R.id.video_length);
        video_length.setText(b.getString("video_length"));

        ImageView video_thumbnail =
(ImageView)findViewById(R.id.video_keyframe_image);
        ImageProcessing ip = new
ImageProcessing(b.getString("video_img"));
        video_thumbnail.setImageBitmap(ip.decodeImage());
    }

    /**
     * Ends the current activity and returns to the previous activity
     */
    public void onCrossBrowser_btn_click(View v){
        Intent cb_intent = new
Intent(this,com.video_browser_thesis.activities.CrossBrowserActivity.c
lass);
        cb_intent.putExtras(b);
        startActivity(cb_intent);
    }
```

```
    public void onResume(){
        super.onResume();
        CrossBrowserActivity cb_i = new CrossBrowserActivity();
        if (cb_i.i == true){
            cb_i.i = false;
            finish();
        }
    }

    /**
     * Ends the current activity and returns to the main activity
     */
    public void onHomeButtonClick(View v){
        finish();
    }
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a MainGallery Video Adapter which extends a Base Adapter
 * @author Anshul Saini
 * @version 5
 */
public class MainGallery_VideoAdapter extends BaseAdapter {

    private Activity activity;
    private static LayoutInflater inflater=null;
    Video_attributes video_attr[] = null;

    public MainGallery_VideoAdapter(Activity a,Video_attributes[]
v_attr) {
        activity = a;
        inflater =
(LayoutInflater)activity.getSystemService(Context.LAYOUT_INFLATER_SERV
ICE);
        this.video_attr = v_attr;
    }

    public int getCount() {
        return video_attr.length;
    }
```

```
    public Object getItem(int position) {
        return position;
    }

    public long getItemId(int position) {
        return position;
    }

    /**
     * Inflates the gallery item with different types of views.
     */
    public View getView(int position, View convertView, ViewGroup
parent) {
        View vi=convertView;
        ViewHolder holder;
        if(convertView==null){
            vi = inflater.inflate(R.layout.categorized_gallery_list,
null);
            holder=new ViewHolder();

holder.video_name=(TextView)vi.findViewById(R.id.video_name);

holder.video_author=(TextView)vi.findViewById(R.id.video_author);

holder.video_image=(ImageView)vi.findViewById(R.id.video_image);
            vi.setTag(holder);
        }
        else{
            holder=(ViewHolder)vi.getTag();
        }

        Video_attributes v_attr = video_attr[position];
        holder.video_name.setText(v_attr.video_title);
        holder.video_author.setText(v_attr.video_author);
        holder.video_image.setImageBitmap(v_attr.video_image);

        return vi;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_name    Video Frame point in TextView format
     * @param video_author  Video author in TextView format
     * @param video_author  Video image in ImageView format
```

```
     */
    public static class ViewHolder{
        public TextView video_name;
        public TextView video_author;
        public ImageView video_image;
    }
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini


 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.video_browser_thesis.activities;

import android.app.Activity;

/**
 * Displays all the videos in various categories
 * @author Anshul Saini
 * @version 5
 */
public class MainGalleryActivity extends Activity {
    private  Gallery galleryView1,galleryView2,galleryView3,
galleryView4,galleryView5;
    private DisplayMetrics metrics = new DisplayMetrics();

    private static final String TAG_CATEGORY01 =
"charlie_bit_my_finger";
    private static final String TAG_CATEGORY02 = "golden_voice";
    private static final String TAG_CATEGORY03 =
"samsung_galaxy_s3_ad";
    private static final String TAG_CATEGORY04 = "zinedine_zidane";
    private static final String TAG_CATEGORY05 =
"ray_william_johnson";

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_categorized_video_gallery);

        getWindowManager().getDefaultDisplay().getMetrics(metrics);

        galleryView1 = (Gallery)findViewById(R.id.galleryid1);
        galleryView2 = (Gallery)findViewById(R.id.galleryid2);
        galleryView3 = (Gallery)findViewById(R.id.galleryid3);
        galleryView4 = (Gallery)findViewById(R.id.galleryid4);
        galleryView5 = (Gallery)findViewById(R.id.galleryid5);

        TextView c1 = (TextView)findViewById(R.id.category01);
        TextView c2 = (TextView)findViewById(R.id.category02);
        TextView c3 = (TextView)findViewById(R.id.category03);
        TextView c4 = (TextView)findViewById(R.id.category04);
        TextView c5 = (TextView)findViewById(R.id.category05);

        MarginLayoutParams mlp1 = (MarginLayoutParams)
galleryView1.getLayoutParams();
        MarginLayoutParams mlp2 = (MarginLayoutParams)
galleryView2.getLayoutParams();
        MarginLayoutParams mlp3 = (MarginLayoutParams)
galleryView3.getLayoutParams();
        MarginLayoutParams mlp4 = (MarginLayoutParams)
galleryView4.getLayoutParams();
        MarginLayoutParams mlp5 = (MarginLayoutParams)
galleryView5.getLayoutParams();

        setMargins(mlp1);
        setMargins(mlp2);
        setMargins(mlp3);
        setMargins(mlp4);
        setMargins(mlp5);

        MainGalleryProcessing mg_p1 = new
MainGalleryProcessing(TAG_CATEGORY01);
        MainGalleryProcessing mg_p2 = new
MainGalleryProcessing(TAG_CATEGORY02);
        MainGalleryProcessing mg_p3 = new
MainGalleryProcessing(TAG_CATEGORY03);
        MainGalleryProcessing mg_p4 = new
MainGalleryProcessing(TAG_CATEGORY04);
        MainGalleryProcessing mg_p5 = new
MainGalleryProcessing(TAG_CATEGORY05);
```

```
        final Video_attributes[] v_attr_01 =
mg_p1.getCategoryVideos();
        final Video_attributes[] v_attr_02 =
mg_p2.getCategoryVideos();
        final Video_attributes[] v_attr_03 =
mg_p3.getCategoryVideos();
        final Video_attributes[] v_attr_04 =
mg_p4.getCategoryVideos();
        final Video_attributes[] v_attr_05 =
mg_p5.getCategoryVideos();

        galleryView1.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_01));
        galleryView2.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_02));
        galleryView3.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_03));
        galleryView4.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_04));
        galleryView5.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_05));

        galleryView1.setSelection(1);
        galleryView2.setSelection(1);
        galleryView3.setSelection(1);
        galleryView4.setSelection(1);
        galleryView5.setSelection(1);

        c1.setText(TAG_CATEGORY01.replace("_"," ").toUpperCase());
        c2.setText(TAG_CATEGORY02.replace("_"," ").toUpperCase());
        c3.setText(TAG_CATEGORY03.replace("_"," ").toUpperCase());
        c4.setText(TAG_CATEGORY04.replace("_"," ").toUpperCase());
        c5.setText(TAG_CATEGORY05.replace("_"," ").toUpperCase());

        galleryView1.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_01, position);
            }
```

```
        });

        galleryView2.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_02, position);
            }
        });

        galleryView3.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_03, position);
            }
        });

        galleryView4.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_04, position);
            }
        });

        galleryView5.setOnItemClickListener(new OnItemClickListener()
{

            @Override
```

```
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_05, position);
            }
        });
    }


    private void setMargins(MarginLayoutParams mlp){
        mlp.setMargins(-(metrics.widthPixels / 2 + 155),
mlp.topMargin,
                mlp.rightMargin, mlp.bottomMargin);
    }

    /**
     * Creates a bundle of data to pass it to the another activity
     *
     * @param galleryIntent New Intent
     * @param extras        The bundle which carries all data from the
galleryIntent
     * @return
     * */
    private void passingBundles(Video_attributes[] v_arr,int position)
{
        Intent galleryIntent = new
Intent(getApplicationContext(),com.video_browser_thesis.activities.Key
frameListActivity.class);
        Bundle extras = new Bundle();

        extras.putString("video_title", v_arr[position].video_title);
        extras.putString("video_description",
v_arr[position].description);
        extras.putString("video_upload_time",
v_arr[position].uploaded_time);
        extras.putString("video_view_count",
v_arr[position].view_count);
        extras.putString("video_author",
v_arr[position].video_author);
        extras.putString("video_img", v_arr[position].thumbnail);
        extras.putString("video_length", v_arr[position].length);
        extras.putString("video_id", v_arr[position].video_id);
        extras.putString("video_url",v_arr[position].player_url);
```

```
        galleryIntent.putExtras(extras);
        startActivity(galleryIntent);
    }

    /**
     * Does nothing onHomeButton click as it is already in the home
activity
     * @param v
     */
    public void onHomeButtonClick(View v){
        //Do Nothing
    }
}
```

```java
package com.video_browser_thesis.processing;

import java.util.ArrayList;

public class MainGalleryProcessing implements Processing{
    private static String url;
    private String category_name;

    private static final String TAG_TITLE = "Title";
    private static final String TAG_AUTHOR = "Author";
    private static final String TAG_THUMBNAIL = "Thumbnail0";
    private static final String TAG_ID = "ID";
    private static final String TAG_VIDEOID = "VideoID";
    private static final String TAG_PLAYER_URL = "PlayerUrl";
    private static final String TAG_LENGTH = "Length";
    private static final String TAG_TOPICS = "Topics";
    private static final String TAG_UPLOADED_TIME = "Uploadtime";
    private static final String TAG_VIEW_COUNT = "ViewCount";
    private static final String TAG_DESCRIPTION = "Description";

    Video_attributes[] v_attr;

    public MainGalleryProcessing(String category_name){
        url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
metadataJSONOutput.py";
        this.category_name = category_name;
        setArrayContents();
    }

    /**
     * Organizes the data received from the URL in JSON format into a
HashMap
     *
     * @param  categoryList     Hash Map which contains all the
segregated metadata under different tag names
     * @param  jParser          An instance of JSONParser which parses
data from the provided URL
     * @param  c                An instance JSONObject which stores
all data contained within that JSON Object
     * @param  id               A temporary array to store video ids
     * @param  image_source     A temporary array to store video
images
     * @param  video_url        A temporary array to store video url
     * @param  video_name       A temporary array to store video name
     * @return categoryList
```

```
     * @see    Image
     */
    public ArrayList<HashMap<String, String>> jsonFeed(){

        ArrayList<HashMap<String, String>> categoryList = new
ArrayList<HashMap<String, String>>();
        JSONArray category = null;
        /** Creating JSON Parser instance */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String id, title, thumbnail, video_id, player_url, length,
author, topics, upload_time,view_count,description;
        HashMap<String, String> map;

        /** Getting JSON string from URL */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /** Getting Array of Contacts */
            category = json.getJSONArray(category_name);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /** Storing each json item in variable */
                id = c.getString(TAG_ID);
                title = c.getString(TAG_TITLE);
                thumbnail = c.getString(TAG_THUMBNAIL);
                video_id = c.getString(TAG_VIDEOID);
                player_url = c.getString(TAG_PLAYER_URL);
                length = c.getString(TAG_LENGTH);
                author = c.getString(TAG_AUTHOR);
                topics = c.getString(TAG_TOPICS);
                upload_time = c.getString(TAG_UPLOADED_TIME);
                view_count = c.getString(TAG_VIEW_COUNT);
                description = c.getString(TAG_DESCRIPTION);

                /** Creating new HashMap */
                map = new HashMap<String, String>();

                /** adding each child node to HashMap key => value */
                map.put(TAG_ID, id);
                map.put(TAG_TITLE, title);
                map.put(TAG_THUMBNAIL, thumbnail);
```

```
                map.put(TAG_VIDEOID, video_id);
                map.put(TAG_PLAYER_URL, player_url);
                map.put(TAG_LENGTH, length);
                map.put(TAG_AUTHOR, author);
                map.put(TAG_TOPICS, topics);
                map.put(TAG_UPLOADED_TIME, upload_time);
                map.put(TAG_VIEW_COUNT, view_count);
                map.put(TAG_DESCRIPTION, description);

                /** Adding HashList to ArrayList */
                categoryList.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return categoryList;
    }


    /**
     * Segregates and organizes Metadata content
     *
     * @param
     * */
    public void setArrayContents(){

        ArrayList<HashMap<String, String>> category_list = jsonFeed();
        Video_attributes video_attr[] = new
Video_attributes[category_list.size()];

        for(int i=0; i<category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(category_list.get(i).get(TAG_THUMBNAIL).toString());

            video_attr[i] = new Video_attributes(
                    category_list.get(i).get(TAG_TITLE).toString(),
                    category_list.get(i).get(TAG_AUTHOR).toString(),
                    ip.decodeImage(),
                    category_list.get(i).get(TAG_ID).toString(),
                    category_list.get(i).get(TAG_VIDEOID).toString(),

category_list.get(i).get(TAG_PLAYER_URL).toString(),
                    category_list.get(i).get(TAG_LENGTH).toString(),
                    category_list.get(i).get(TAG_TOPICS).toString(),
```

```
category_list.get(i).get(TAG_UPLOADED_TIME).toString(),

category_list.get(i).get(TAG_VIEW_COUNT).toString(),

category_list.get(i).get(TAG_DESCRIPTION).toString(),

category_list.get(i).get(TAG_THUMBNAIL).toString());
        }
        v_attr = video_attr;
    }

    public Video_attributes[] getCategoryVideos(){
        return v_attr;
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

import java.util.HashMap;

public interface Processing {
    void setArrayContents();
    ArrayList<HashMap<String, String>> jsonFeed();
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.splashScreen;
/**
 * @author Anshul Saini
 * @version 5
 */
import android.app.Activity;
/**
 * Main Splash Screen when the application is initiated
 *
 * @param  _active      Indicates if the splash screen is present
 * @param  _splashTime  The display time of the splash screen image in
milliseconds
 * @return
 * @see    Image
 */
public class SplashScreen extends Activity {

    protected boolean _active = true;
    protected int _splashTime = 2000;

    /**
     * Called when the activity is first created.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.splash_screen);

        /**
         * Threads for displaying the splash screen
         */
        Thread splashTread = new Thread() {
            @Override
            public void run() {
                try {
                    int waited = 0;
                    while(_active && (waited < _splashTime)) {
                        sleep(100);
                        if(_active) {
                            waited += 100;
                        }
                    }
                } catch(InterruptedException e) {

                } finally {

                    startActivity(new Intent(SplashScreen.this,
MainGalleryActivity.class));
                    finish();
                }
            }
        };
        splashTread.start();
    }
}
```

```
package com.video_browser_thesis.elements;

import android.graphics.Bitmap;
/**
 * Video_attributes
 * @author Anshul Saini
 * @version 5
 */
public class Video_attributes{
    public String video_title;
    public String video_author;
    public Bitmap video_image;
    public String thumbnail;
    public String tag_id;
    public String video_id;
    public String player_url;
    public String length;
    public String topics;
    public String uploaded_time;
    public String view_count;
    public String description;

    public Video_attributes(){
        super();
    }

    public Video_attributes(
```

```
            String video_title,
            String video_author,
            Bitmap video_image,
            String tag_id,
            String video_id,
            String player_url,
            String length,
            String topics,
            String uploaded_time,
            String view_count,
            String description,
            String thumbnail){
        super();
        this.tag_id = tag_id;
        this.video_title = video_title;
        this.video_author = video_author;
        this.video_image = video_image;
        this.video_id = video_id;
        this.player_url = player_url;
        this.length = length;
        this.topics = topics;
        this.uploaded_time = uploaded_time;
        this.view_count = view_count;
        this.description = description;
        this.thumbnail = thumbnail;
    }
}
```

activity_categorized_video_gallery.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <include
        android:id="@+id/header_bar"
        android:layout_alignParentTop="true"
        layout="@layout/app_header_layout" />

    <ScrollView
        android:id="@+id/scroll_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_below="@+id/header_bar"
        android:background="@color/
categorized_video_gallery_activity_background" >

        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical" >

            <include android:id="@+id/category01" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid1" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category02" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid2" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category03" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid3" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category04" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid4" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category05" layout="@layout/
```

```
category_heading_text_layout"/>
            <include android:id="@+id/galleryid5" layout="@layout/
category_gallery_layout"/>

        </LinearLayout>
    </ScrollView>
</RelativeLayout>
```

activity_cross_browser.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/cross_browser_background" >

    <include
        android:id="@+id/third_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/third_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:onClick="onListView_btn_click"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/cross_browser_tab_txt" />
    </LinearLayout>
```

```
    <ListView
        android:id="@+id/cross_browser_listview"
        android:layout_width="94dp"
        android:layout_height="fill_parent"
        android:layout_below="@+id/button_area"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="5dp"
        android:background="@color/cross_browser_listview_background"
        android:paddingBottom="5dp" />

    <Gallery
        android:id="@+id/cross_browser_gallery"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button_area"
        android:background="@color/cross_browser_gallery_background" /
>

    <TextView
        android:id="@+id/matching_keyFrame_txt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/selection_border01"
        android:layout_alignLeft="@+id/selection_border04"
        android:layout_marginLeft="5dp"
        android:background="#AA000000"
        android:paddingBottom="7dp"
        android:paddingLeft="30dp"
        android:paddingRight="15dp"
        android:text="@string/matching_keyFrame_txt"
        android:textColor="#ffffff"
        android:textStyle="bold"
        android:typeface="sans" />

    <TextView
        android:id="@+id/selection_border01"
        android:layout_width="fill_parent"
        android:layout_height="4dp"
        android:layout_alignTop="@+id/cross_browser_gallery"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border02"
        android:layout_width="fill_parent"
        android:layout_height="4dp"
```

```
        android:layout_alignBottom="@+id/cross_browser_gallery"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border03"
        android:layout_width="5dp"
        android:layout_height="72dp"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="150dp"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border04"
        android:layout_width="5dp"
        android:layout_height="72dp"
        android:layout_alignRight="@+id/cross_browser_listview"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="150dp"
        android:background="@color/cross_browser_selection_border"
        android:paddingLeft="1dp" />

    <RelativeLayout
        android:id="@+id/selected_frame_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignLeft="@+id/selection_border04"
        android:layout_alignTop="@+id/selection_border02"
        android:layout_marginLeft="5dp"
        android:background="#000000"
        android:layout_marginTop="8dp">

        <TextView
            android:id="@+id/selected_matching_keyframe_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="55dp"
            android:layout_marginTop="30dp"
            android:background="#AA000000"
            android:paddingLeft="5dp"
            android:paddingRight="5dp"
            android:text="@string/selected_matching_keyframe_txt"
            android:textColor="#ffffff"
            android:textSize="16sp"
            android:textStyle="bold"
```

```
            android:typeface="sans" />

        <ImageView
            android:id="@+id/selected_matching_keyframe"
            android:layout_width="250dp"
            android:layout_height="250dp"
            android:scaleType="fitCenter"
            android:layout_below="@+id/selected_matching_keyframe_txt"
            android:contentDescription="@string/no_description"
            android:src="@drawable/splash_screen_img"
            android:layout_marginLeft="30dp" />
    </RelativeLayout>

</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

app_header_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="@color/app_header_background" >

        <ImageView
            android:id="@+id/app_header_homebutton"
            android:layout_width="@dimen/app_header_home_button"
            android:layout_height="@dimen/app_header_home_button"
            android:layout_alignParentLeft="true"
            android:layout_centerInParent="true"
            android:layout_marginLeft="5dp"
            android:contentDescription="@string/no_description"
            android:src="@drawable/home"
            android:onClick="onHomeButtonClick" />

        <ImageView
            android:id="@+id/app_header_img"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_centerInParent="true"
            android:layout_toRightOf="@+id/app_header_homebutton"
            android:contentDescription="@string/no_description"
            android:src="@drawable/app_header" />
    </RelativeLayout>

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="@dimen/
app_header_bottom_divider_thickness"
        android:background="@color/app_header_bottom_divider" />

</LinearLayout>
```

categorized_gallery_item_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/video_image"
        android:layout_width="130dp"
        android:layout_height="110dp"
        android:layout_gravity="center"
        android:contentDescription="@string/no_description"
        android:paddingRight="8dp"
        android:scaleType="fitXY"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/video_name"
        android:layout_width="112dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:layout_marginTop="70dp"
        android:background="@color/
video_attributes_overlayed_text_background"
        android:maxLength="17"
        android:maxLines="1"
        android:paddingLeft="1dp"
        android:text="@string/video_title_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textSize="12sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/video_author"
        android:layout_width="112dp"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_name"
        android:layout_below="@+id/video_name"
        android:background="@color/
video_attributes_overlayed_text_background"
        android:maxLength="17"
        android:maxLines="1"
        android:paddingLeft="1dp"
        android:text="@string/video_author_txt"
```

```
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textSize="10sp"
        android:textStyle="italic" />

</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

cross_browser_galleryitem.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <ImageView
        android:id="@+id/cross_browsers_gallery_img"
        android:layout_width="90dp"
        android:layout_height="71dp"
        android:contentDescription="@string/no_description"
        android:src="@drawable/ic_launcher"
        android:layout_marginTop="1dp"
        android:layout_marginBottom="1dp"
        android:paddingLeft="10dp"
        android:paddingRight="10dp" />

    <TextView
        android:id="@+id/cb_gallery_video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="45dp"
        android:background="#A0000000"
        android:text="@string/video_time_length"
        android:textColor="@android:color/white"
        android:textSize="8sp"/>

</RelativeLayout>
```

cross_browser_listitem.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <ImageView
        android:id="@+id/cross_browsers_img"
        android:layout_width="90dp"
        android:layout_height="71dp"
        android:contentDescription="@string/no_description"
        android:src="@drawable/ic_launcher"
        android:layout_marginTop="1dp"
        android:layout_marginBottom="1dp" />

    <TextView
        android:id="@+id/cb_video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="45dp"
        android:background="#A0000000"
        android:text="@string/video_time_length"
        android:textColor="@android:color/white" />

</RelativeLayout>
```

keyframe_list_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:id="@+id/LinearLayout01"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:padding="3sp" >

    <include layout="@layout/keyframe_listview" />

</LinearLayout>
```

keyframe_listview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:id="@+id/main_keyframe_area"
    android:layout_width="fill_parent"
    android:layout_height="80dp"
    android:layout_below="@+id/button_area"
    android:layout_marginTop="10dp"
    android:background="#484848" >

    <ImageView
        android:id="@+id/video_keyframe_image"
        android:layout_width="90dp"
        android:layout_height="70dp"
        android:layout_gravity="center"
        android:contentDescription="@string/no_description"
        android:paddingRight="8dp"
        android:scaleType="fitXY"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="50dp"
        android:background="@color/category_text_background"
        android:text="@string/video_time_length"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/video_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/video_keyframe_image"
        android:layout_marginLeft="15dp"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="1"
        android:text="@string/video_title_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textStyle="bold"
        android:typeface="normal" />
```

```
    <TextView
        android:id="@+id/video_description"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_title"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="2"
        android:text="@string/video_description_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:typeface="normal" />

    <TextView
        android:id="@+id/video_uploaded_timestamp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_description"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="2"
        android:text="@string/video_uploaded_timestamp_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:typeface="normal" />

    <TextView
        android:id="@+id/upload_viewcount_divider"
        android:layout_width="2dp"
        android:layout_height="13dp"
        android:layout_below="@+id/video_description"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="1dp"
        android:layout_toRightOf="@+id/video_uploaded_timestamp"
        android:background="#ffffff" />

    <TextView
        android:id="@+id/view_count"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/video_description"
        android:layout_marginLeft="3dp"
        android:layout_toRightOf="@+id/upload_viewcount_divider"
        android:maxLines="1"
```

```
        android:text="@string/video_view_count_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

    <TextView
        android:id="@+id/by_txt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_uploaded_timestamp"
        android:text="@string/by_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

    <TextView
        android:id="@+id/video_author"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/video_uploaded_timestamp"
        android:layout_marginLeft="3dp"
        android:layout_toRightOf="@+id/by_txt"
        android:text="@string/video_author_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

</RelativeLayout>
```

splash_screen.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/splash_background"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/splash_screen_img"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:contentDescription="@string/no_description"
        android:src="@drawable/splash_screen_img" />

</RelativeLayout>
```

color.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <!-- Application main header colors -->
    <color name="app_header_background">#000000</color>
    <color name="app_header_bottom_divider">#00ffff</color>

    <!-- Categorized video gallery activity -->
    <color
name="categorized_video_gallery_activity_background">#8b8b83</color>
    <color name="category_text_color">#fffafa</color>
    <color name="category_text_background">#A0000000</color>
    <color name="categorized_gallery_background">#484848</color>
    <color
name="video_attributes_overlayed_text_background">#AA000000</color>
    <color name="video_attributes_overlayed_text_color">#ffffff</
color>

    <!-- Keyframe Activity -->
    <color name="keyframe_listview">#696969</color>

    <!-- Cross Browser -->
    <color name="cross_browser_background">#8b8b83</color>
    <color name="cross_browser_listview_background">#484848</color>
    <color name="cross_browser_gallery_background">#484848</color>
    <color name="cross_browser_selection_border">#00bfff</color>
</resources>
```

dimens.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <!-- Application Header elements dimensions -->
    <dimen name="app_header_home_button">50dp</dimen>
    <dimen name="app_header_bottom_divider_thickness">2dp</dimen>
</resources>
```

strings.xml

```
<resources>

    <string name="app_name">Video_Browser</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string
name="title_activity_categorized_video_gallery">CategorizedVideoGaller
y</string>

    <!-- For all image descriptions -->
    <string name="no_description">No Description</string>

    <!-- Category default names on home activity -->
    <string name="category_txt">Category Name</string>

    <!-- For all video elements -->
    <string name="video_title_txt">Video Title</string>
    <string name="video_author_txt">Video Author</string>
    <string name="video_description_txt">Video Description</string>
    <string name="video_uploaded_timestamp_txt">Uploaded Time</string>
    <string name="video_view_count_txt">View Count</string>
    <string name="video_views_txt">views</string>
    <string name="by_txt">by</string>

    <!-- KeyFrame Activity -->
    <string name="listView_tab_txt">Listview</string>
    <string name="cross_browser_tab_txt">Cross Browser</string>
    <string name="video_time_length">Length</string>

    <!-- Matching keyframes -->
    <string name="matching_keyFrame_txt">Matching KeyFrames:</string>
    <string name="selected_matching_keyframe_txt">Selected KeyFrame:</
string>
</resources>
```

styles.xml

```
<resources>

    <style name="AppTheme" parent="android:Theme.Light" />

</resources>
```

project.properties

```
# This file is automatically generated by Android Tools.
# Do not modify this file -- YOUR CHANGES WILL BE ERASED!
#
# This file must be checked in Version Control Systems.
#
# To customize properties used by the Ant build system edit
# "ant.properties", and override values to adapt the script to your
# project structure.
#
# To enable ProGuard to shrink and obfuscate your code, uncomment this
(available properties: sdk.dir, user.home):
#proguard.config=${sdk.dir}/tools/proguard/proguard-
android.txt:proguard-project.txt

# Project target.
target=android-10
```

```
import sqlite3
import sqlite3 as lite
import os
import glob
import ConfigParser
import re
import sys
import json
import unicodedata
from xml.dom.minidom import parseString

global xmlperkeyword

try:
    cf = ConfigParser.ConfigParser()
        cf.read('./config.ini')
        xmlperkeyword = int(cf.get('NUMBERS', 'xml_per_keyword'))
except Exception,e:
        print e
        print 'Error: Invalid config file.'
        exit(1)

#print '-------------------------------------------------------------------------'
#print 'START CRAWLXML.PY'
#print '-------------------------------------------------------------------------'
#os.chdir(os.path.expanduser("~/data/crawling/"))
#os.system('python2.7 ./crawlxml.py -i')

#print '-------------------------------------------------------------------------'
#print 'START CRAWLVIDEO.PY'
#print '-------------------------------------------------------------------------'
#os.system('python2.7 ./crawlvideo.py -i')

#print '-------------------------------------------------------------------------'
#print 'START BATCHRUN.PY'
#print '-------------------------------------------------------------------------'
#os.chdir(os.path.expanduser("~/data/videosplitting/"))
#os.system('python2.7 ./batchrun.py -i')

# connect to database
os.chdir(os.path.expanduser("~/data/www/cgi-bin/"))
conn = sqlite3.connect('videos.sqlite3')
c = conn.cursor()


#c.execute ('''DROP TABLE IF EXISTS topics''')
#c.execute ('''DROP TABLE IF EXISTS videosMetaData''')
#c.execute ('''DROP TABLE IF EXISTS imageFrames''')
```

```
#c.execute ('''DROP TABLE IF EXISTS matchingFrames''')

c.execute('''CREATE TABLE topics (id INTEGER PRIMARY KEY  AUTOINCREMENT  NOT N

c.execute('''CREATE TABLE videosMetaData
(id INTEGER PRIMARY KEY  AUTOINCREMENT  NOT NULL , title TEXT NOT NULL , videc
query TEXT, rank INTEGER, uploadtime DATETIME, description TEXT, category TEXT
tags TEXT, playerurl TEXT, flashplayerurl TEXT, length DOUBLE, author TEXT, cc
viewcount INTEGER, ratingavg TEXT,thumbnail0 TEXT, thumbnail1 TEXT, thumbnail2
FOREIGN KEY(topics) REFERENCES topics(id))''')

c.execute('''CREATE TABLE imageFrames
(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, imageUrl TEXT, imageName TEXT,
FOREIGN KEY (videoRef) REFERENCES videoMetaData (videoID))''')

c.execute('''CREATE TABLE matchingFrames
(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, referenceFrameID TEXT, matchir


# inserts the topics into the topics table (excludes trash directory)

print '-------------------------------------------------------------------------
print 'ADDING ROWS TO TOPICS (CATEGORY) TABLE '
print '-------------------------------------------------------------------------
os.chdir(os.path.expanduser("~/data/crawling/topics/"))
for files in glob.glob("*"):
    if not files.startswith('trash') and not len(files) == 5 and not files.end
        c.execute ("INSERT INTO topics (name) VALUES(?)", (files,))

def count_letters(word):
    count = 0
    for c in word:
        count = count + 1
    return count

def listSubDir(filename, subdirectory=''):
    if subdirectory:
        path = subdirectory
    else:
        path = os.getcwd()
    for root, dirs, names in os.walk(path):
    if os.path.exists (os.path.join(root, filename)):
        return os.path.join(root, filename)


# inserts matching keyframes into the matchingFrame table

print '-------------------------------------------------------------------------
```

```
print 'ADDING ROWS TO MATCHING FRAME TABLE '
print '-------------------------------------------------------------------------
mKFramePath = os.path.expanduser("~/data/www/memexplore/matchingKeyFramesJSON/
baseUrl = "http://cantabile.anu.edu.au/memexplore/"
os.chdir(mKFramePath)
for path,subdirs,files in os.walk(mKFramePath):
    for file in files:
        filePath = listSubDir(file,mKFramePath)
        json_data=open(filePath).read()
        data = json.loads(json_data)
        ok = data["FileMatches"]
        referenceFileName = unicodedata.normalize('NFKD', data["FileName"]).er
        for i in range(len(ok)):
            c.execute ("INSERT INTO matchingFrames(referenceFrameID, matchingF
        #json_data.close();



# inserts images into imageFrames table in binary format

print '-------------------------------------------------------------------------
print 'ADDING IMAGES TO IMAGEFRAMES TABLE '
print '-------------------------------------------------------------------------
imageFrameDir = os.path.expanduser("~/data/www/memexplore/shot1Test/")
os.chdir(imageFrameDir)
for path,subdirs, files in os.walk(imageFrameDir):
    for subDir in subdirs:
        if (count_letters(subDir) > 1):
            storeVideoId = subDir
    for file in files:
        #print file
        filepath = listSubDir(file,imageFrameDir)
        fin = None
        try:
            fin = open (filepath,'rb')
            img = fin.read()
            binary = lite.Binary(img)
            c.execute ("INSERT INTO imageFrames(imageUrl,imageName,videoRef) \
            #print (filepath)
        except IOError, e:
            print "Error %d: %s" % (e.args[0],e.args[1])
            ok123 = "error"
            print ok123
        finally:
            if fin:
                fin.close()
```

```
# inserts metadata into the metadata table

print '-------------------------------------------------------------------------'
print 'ADDING ROWS TO XML METADATA TABLE '
print '-------------------------------------------------------------------------'
x = 0
previous = 0

root = os.path.expanduser("~/data/crawling/xml/raw/")
os.chdir(root)
for path,subdirs,files in os.walk(root):
    for file in files:
    xmlFilePath = listSubDir(file,root)
    if '/relevance/' not in xmlFilePath:
        xmlFilePath = "notValid"
    else:
        new = xmlFilePath.find('/relevance/')
    if previous != new:
        x = x + 1
        previous = new
    try:

        f=open(xmlFilePath, 'r')
        data = f.read()
        dom = parseString(data)
        try:
            xmlTitle = dom.getElementsByTagName('title')[0].toxml()
            titleData = xmlTitle.replace('<title>','').replace('</title>','')
        except IndexError:
            titleData = None

        try:
            xmlQuery = dom.getElementsByTagName('query')[0].toxml()
            queryData = xmlQuery.replace('<query>','').replace('</query>','')
        except IndexError:
            queryData = None

        try:
            xmlRank = dom.getElementsByTagName('rank')[0].toxml()
            rankData = xmlRank.replace('<rank>','').replace('</rank>','')
        except IndexError:
            rankData = None

        try:
            xmlUploadtime = dom.getElementsByTagName('uploadtime')[0].toxml()
```

```
        uploadtimeData = xmlUploadtime.replace('<uploadtime>','').replace(
    except IndexError:
        uploadtimeData = None

    try:
        xmlDescription = dom.getElementsByTagName('description')[0].toxml(
        descriptionData = xmlDescription.replace('<description>','').repla
    except IndexError:
        descriptionData = None

    try:
        xmlCategory = dom.getElementsByTagName('category')[0].toxml()
        categoryData = xmlCategory.replace('<category>','').replace('</cat
    except IndexError:
        categoryData = None

    try:
        xmlTags = dom.getElementsByTagName('tags')[0].toxml()
        tagsData = xmlTags.replace('<tags>','').replace('</tags>','')
    except IndexError:
        tagsData = None

    try:
        xmlPlayerurl = dom.getElementsByTagName('palyerurl')[0].toxml()
        playerurlData = xmlPlayerurl.replace('<palyerurl>','').replace('</
    except IndexError:
        playerurlData = None

    try:
        xmlFlashplayerurl = dom.getElementsByTagName('flashpalyerurl')[0].
        flashplayerurlData = xmlFlashplayerurl.replace('<flashpalyerurl>',
    except IndexError:
        flashplayerurl = None

    try:
        xmlLength = dom.getElementsByTagName('length')[0].toxml()
        lengthData = xmlLength.replace('<length>','').replace('</length>',
    except IndexError:
        lengthData = None

    try:
        xmlAuthor = dom.getElementsByTagName('author')[0].toxml()
        authorData = xmlAuthor.replace('<author>','').replace('</author>',
    except IndexError:
        authorData = None

    try:
        xmlComment_count = dom.getElementsByTagName('comment_count')[0].to
```

```
        comment_countData = xmlComment_count.replace('<comment_count>','')
    except IndexError:
        comment_countData = None

    try:
        xmlFavorite_count = dom.getElementsByTagName('favorite_count')[0].
        favorite_countData = xmlFavorite_count.replace('<favorite_count>',
    except IndexError:
        favorite_countData = None

    try:
        xmlViewcount = dom.getElementsByTagName('viewcount')[0].toxml()
        viewcountData = xmlViewcount.replace('<viewcount>','').replace('</
    except IndexError:
        viewcountData = None

    try:
        xmlRating_avg = dom.getElementsByTagName('rating_avg')[0].toxml()
        rating_avgData = xmlRating_avg.replace('<rating_avg>','').replace(
    except IndexError:
        rating_avgData = None

    try:
        xmlThumbnail0 = dom.getElementsByTagName('thumbnail')[0].toxml()
        thumbnailData0 = xmlThumbnail0.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData0 = None

    try:
        xmlThumbnail1 = dom.getElementsByTagName('thumbnail')[1].toxml()
        thumbnailData1 = xmlThumbnail1.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData1 = None

    try:
        xmlThumbnail2 = dom.getElementsByTagName('thumbnail')[2].toxml()
        thumbnailData2 = xmlThumbnail2.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData2 = None

    try:
        xmlThumbnail3 = dom.getElementsByTagName('thumbnail')[3].toxml()
        thumbnailData3 = xmlThumbnail3.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData3 = None

    try:
        videoIDData = playerurlData[31:42]
```

```
        except IndexError:
            videoIDData = None

        topic = x

        c.execute('''INSERT INTO videosMetaData (title,videoID,query,rank,uplo
        f.close()
    except IOError, e:
        ok = "testtt";

c.execute('pragma foreign_keys = on')

# save (commit) the changes
conn.commit()

# close connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json
import binascii
import base64

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')

# connect to Database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
ImageSource = '"%s"' % "ImageSource"
Name = '"%s"' % "ImageName"
VideoRef = '"%s"' % "VideoRef"
tempStr = "["

# perform SQL query and print JSON
cursor.execute ("SELECT MAX (imageName) FROM imageFrames WHERE videoRef = ?",(
lastEntryInEachCategory = cursor.fetchone()[0]
print "{\n\t"
print '"%s"' % "key_frame" + ": [\n"
for cg in cursor.execute('SELECT * FROM imageFrames WHERE videoRef = ? ORDER E
    print json.dumps({'ID':"%s" % cg[0],'ImageSource':"%s" % cg[1],'Name':cg[2
    if cg[2] != lastEntryInEachCategory:
        print ","
print "\n\t]\n}"

# close Database connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json
import string

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')
#last_name  = form.getvalue('last_name')

# connect to Database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
VideoTitle = '"%s"' % "Title"
VideoID = '"%s"' % "VideoID"
Query = '"%s"' % "Query"
Rank = '"%s"' % "Rank"
UploadTime = '"%s"' % "Uploadtime"
Description = '"%s"' % "Description"
Category = '"%s"' % "Category"
Tags = '"%s"' % "Tags"
PlayerUrl = '"%s"' % "PlayerUrl"
FlashPlayerUrl = '"%s"' % "FlashPlayerUrl"
Length = '"%s"' % "Length"
Author = '"%s"' % "Author"
CommentCount = '"%s"' % "CommentCount"
FavoriteCount = '"%s"' % "FavoriteCount"
ViewCount = '"%s"' % "ViewCount"
RatingAvg = '"%s"' % "RatingAvg"
Thumbnail0 = '"%s"' % "Thumbnail0"
Thumbnail1 = '"%s"' % "Thumbnail1"
Thumbnail2 = '"%s"' % "Thumbnail2"
```

```
Thumbnail3 = '"%s"' % "Thumbnail3"
Topics = '"%s"' % "Topics"

print "{"

# iterative query for creating JSON
def iterateSqlQuery (parameters,test,idd):
    for names in cursor.execute("SELECT * FROM topics WHERE id = ?", (paramete
        tempStr = "\t" + '"%s"' % names[1] + ": ["
        cursor.execute ("SELECT MAX (id) FROM videosMetaData WHERE topics =?",
        lastEntryInEachCategory = cursor.fetchone()[0]
        for cg in cursor.execute("SELECT * FROM videosMetaData WHERE topics =?
            id = Id + ": " + '"%s"' % str(cg[0]) + ", " + "\n"
            videoTitle = VideoTitle + ": " + '"%s"' % cg[1] + ", " + "\n"
            videoUniqueID = VideoID + ": " + '"%s"' % cg[2] + ", " + "\n"
            uploadTime = UploadTime + ": " + '"%s"' % cg[5] + ", " + "\n"
            description = Description + ": " + '"%s"' % cg[6] + ", " + "\n"
            playerurl = cg[9]
            playerurl = PlayerUrl + ": " + '"%s"' % playerurl[:-33] + ", " + '
            length = Length + ": " + '"%s"' % str(cg[11]) + ", " + "\n"
            author = Author + ": " + '"%s"' % cg[12] + ", " + "\n"
            viewCount = ViewCount + ": " + '"%s"' % cg[15] + ", " + "\n"
            thumbnail0 = Thumbnail0 + ": " + '"%s"' % cg[17] + ", " + "\n"
            topics = Topics + ": " + '"%s"' % str(cg[21]) + "\n"
            tempStr = tempStr + "\n\t\t" + "{" + "\n\t\t\t" + id + "\t\t\t" +
            if len(id) > 0 and cg[0] == lastEntryInEachCategory:
                tempStr = tempStr[:-2]
        if (test == idd):
            tempStr = tempStr + "\n\t]"
        else:
            tempStr = tempStr + "\n\t],"
    print filter(lambda x: x in string.printable, tempStr)  # filters out non-
    test = test + 1
    if test <= idd:
        iterateSqlQuery(parameters + 1,test,idd)
    return

cursor.execute ("SELECT MAX (id) FROM topics")
lastID = cursor.fetchone()[0]
iterateSqlQuery(1,1,lastID)

print "}"

# close database connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')

# Connect to the database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
ReferenceFrameID = '"%s"' % "ReferenceFrameID"
MatchingFrame = '"%s"' % "MatchingFrame"
MatchingFrameURL = '"%s"' % "MatchingFrameURL"
tempStr =  "{\n\t" + '"%s"' % "matching_key_frames" + ": [\n"

# Query the database
for cg in cursor.execute('SELECT * FROM matchingFrames WHERE referenceFrameID
    id = Id + ": " + '"%s"' % str(cg[0]) + ", " + "\n"
    referenceFrameID = ReferenceFrameID + ": " + '"%s"' % cg[1] + ", " + "\n"
    matchingFrame = MatchingFrame + ": " + '"%s"' % cg[2] + ", " + "\n"
    matchingFrameURL = MatchingFrameURL + ": " + '"%s"' % cg[3] + "\n"
    tempStr = tempStr + "\n" + "\t" + "\t" + "{" + "\n" + "\t" + "\t" + "\t" +

tempStr = tempStr[:-2] + "\n\t]\n}"

# output JSON
print tempStr

# Close connection to Database
conn.close()
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

import java.util.HashMap;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import com.video_browser_thesis.elements.KeyFrame_attributes;
import com.video_browser_thesis.json.JSONParser;

public class CB_Processing implements Processing{

    private String base_url;
    private String url;

    private static final String TAG_KEYFRAME = "key_frame";
    private static final String TAG_IMAGE_SOURCE= "ImageSource";
    private static final String TAG_NAME = "Name";

    KeyFrame_attributes[] k_attr;

    public CB_Processing(String attach_url){
        base_url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
keyFramesJSONOutput.py?q=";
        url = base_url + attach_url;
        setArrayContents();
    }

    /**
     * Organizes the data received from the URL in JSON format into a
HashMap
     *
     * @param  categoryList     Hash Map which contains all the
segregated metadata under different tag names
     * @param  jParser          An instance of JSONParser which parses
data from the provided URL
     * @param  c                An instance JSONObject which stores
all data contained within that JSON Object
```

```
     * @param  id               A temporary array to store video ids
     * @param  image_source     A temporary array to store video
images
     * @param  video_url        A temporary array to store video url
     * @param  video_name       A temporary array to store video name
     * @return categoryList
     * @see
     */
    public ArrayList<HashMap<String, String>> jsonFeed(){

        JSONArray category = null;
        ArrayList<HashMap<String, String>> keyframe_list = new
ArrayList<HashMap<String, String>>();

        /**
         * Creating JSON Parser instance
         */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String image_source,video_name;
        HashMap<String, String> map;

        /**
         * Getting JSON string from URL
         */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /**
             * Getting Array of Contacts
             */
            category = json.getJSONArray(TAG_KEYFRAME);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /**
                 * Storing each json item in variable
                 */
                image_source = c.getString(TAG_IMAGE_SOURCE);
                video_name = c.getString(TAG_NAME);

                /** Creating new HashMap */
                map = new HashMap<String, String>();
```

```
                /**
                 * Adding each child node to HashMap key => value
                 */
                map.put(TAG_IMAGE_SOURCE, image_source);
                map.put(TAG_NAME, video_name);

                /**
                 * Adding HashList to ArrayList
                 */
                keyframe_list.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return keyframe_list;
    }

    /**
     * Creates a KeyFrame_attributes object by extracting the relevant
data from the HashMap
     *
     * @param  kf_attr      An instance of KeyFrame_attributes object
     * @return kf_attr
     * @see
     */
    public void setArrayContents(){

        ArrayList<HashMap<String, String>> temp_category_list =
jsonFeed();
        KeyFrame_attributes kf_attr[] = new
KeyFrame_attributes[temp_category_list.size()];

        for(int i=0; i<temp_category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(temp_category_list.get(i).get(TAG_IMAGE_SOURCE).toStri
ng());

            kf_attr[i] = new KeyFrame_attributes(

temp_category_list.get(i).get(TAG_IMAGE_SOURCE).toString(),

temp_category_list.get(i).get(TAG_NAME).substring(32, 38),
                    ip.decodeImage());
```

```
            }

            k_attr = kf_attr;
        }

        public KeyFrame_attributes[] getKeyFrameArray(){
            return k_attr;
        }
}
```

```
/*
 * Copyright (C) 2012 Mobs and Geeks

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.view.View;
/**
 * Constructs a {@linkplain CircularListAdapter}.
 *
 * @param listAdapter A {@link Keyframe_ListAdapter} that has to
behave circular.
 */
public class CircularListAdapter extends BaseAdapter {

    static final boolean DEBUG = false;
    static final String TAG =
CircularListAdapter.class.getSimpleName();

    private BaseAdapter mListAdapter;
    private int mListAdapterCount;

    public CircularListAdapter(BaseAdapter listAdapter) {
        if(listAdapter == null) {
            throw new IllegalArgumentException("listAdapter cannot be
null.");
        }

        this.mListAdapter = listAdapter;
        this.mListAdapterCount = listAdapter.getCount();
```

```
    }

    @Override
    public int getCount() {
        return Integer.MAX_VALUE;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        return mListAdapter.getView(position % mListAdapterCount,
convertView, parent);
    }

    @Override
    public Object getItem(int position) {
        return mListAdapter.getItem(position % mListAdapterCount);
    }

    @Override
    public long getItemId(int position) {
        return mListAdapter.getItemId(position % mListAdapterCount);
    }

    @Override
    public boolean areAllItemsEnabled() {
        return mListAdapter.areAllItemsEnabled();
    }

    @Override
    public int getItemViewType(int position) {
        return mListAdapter.getItemViewType(position %
mListAdapterCount);
    }

    @Override
    public int getViewTypeCount() {
        return mListAdapter.getViewTypeCount();
    }

    @Override
    public boolean isEmpty() {
        return mListAdapter.isEmpty();
    }
```

```
    @Override
    public boolean isEnabled(int position) {
        return mListAdapter.isEnabled(position % mListAdapterCount);
    }

    @Override
    public void notifyDataSetChanged() {
        mListAdapter.notifyDataSetChanged();
        mListAdapterCount = mListAdapter.getCount();

        super.notifyDataSetChanged();
    }

    @Override
    public void notifyDataSetInvalidated() {
        mListAdapter.notifyDataSetInvalidated();
        super.notifyDataSetInvalidated();
    }

    @Override
    public View getDropDownView(int position, View convertView,
ViewGroup parent) {
        return mListAdapter.getDropDownView(position %
mListAdapterCount,
                convertView, parent);
    }

    @Override
    public boolean hasStableIds() {
        return mListAdapter.hasStableIds();
    }

}
```

```
/*
 * Copyright (C) 2012 Anshul Saini


 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a CrossBrowserGallery Video Adapter which extends a Base
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class CrossBrowser_GalleryAdapter extends BaseAdapter {

    private Activity activity;
    private static LayoutInflater inflater=null;
    KeyFrame_attributes keyframe_attr[] = null;

    public CrossBrowser_GalleryAdapter(Activity
a,KeyFrame_attributes[] keyframe_attr) {
        activity = a;
        inflater =
(LayoutInflater)activity.getSystemService(Context.LAYOUT_INFLATER_SERV
ICE);
        this.keyframe_attr = keyframe_attr;
    }
```

```
    public int getCount() {
        return keyframe_attr.length;
    }

    public Object getItem(int position) {
        return position;
    }

    public long getItemId(int position) {
        return position;
    }

    /**
     * Inflates the gallery item with different types of views.
     */
    public View getView(int position, View convertView, ViewGroup
parent) {
        View vi=convertView;
        ViewHolder holder;
        if(convertView==null){
            vi = inflater.inflate(R.layout.cross_browser_galleryitem,
null);
            holder=new ViewHolder();

holder.video_image=(ImageView)vi.findViewById(R.id.cross_browsers_gall
ery_img);

holder.video_length=(TextView)vi.findViewById(R.id.cb_gallery_video_le
ngth);

            vi.setTag(holder);
        }
        else{
            holder=(ViewHolder)vi.getTag();
        }

        KeyFrame_attributes v_attr = keyframe_attr[position];
        holder.video_length.setText(v_attr.frame_point);

        if(v_attr.video_img != null){
            holder.video_image.setImageBitmap(v_attr.video_img);
        }

        return vi;
    }
```

```
    /**
     * Holds all the elements of the view created in its super class
     * @param video_length  Video length in TextView format
     * @param video_image   Video image in ImageView format
     */
    public static class ViewHolder{
        public ImageView video_image;
        public TextView video_length;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a Keyframe KeyFrame Adapter which extends a Custom List
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class CrossBrowser_ListAdapter extends
ArrayAdapter<KeyFrame_attributes>{

    Context context;
    static int layoutResourceId = R.layout.cross_browser_listitem;
    KeyFrame_attributes keyframe_attr[] = null;

    public CrossBrowser_ListAdapter(Context context,
KeyFrame_attributes[] keyframe_attr) {
        super(context, layoutResourceId,keyframe_attr);
        // TODO Auto-generated constructor stub
        this.context = context;
        this.keyframe_attr = keyframe_attr;
    }

    /**
```

```
     * Inflates the gallery item with different types of views.
     */
    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        View row = convertView;
        CrossBrowserListHolder holder = null;

        if(row == null)
        {
            LayoutInflater inflater =
((Activity)context).getLayoutInflater();
            row = inflater.inflate(layoutResourceId, parent, false);

            holder = new CrossBrowserListHolder();
            holder.video_img =
(ImageView)row.findViewById(R.id.cross_browsers_img);
            holder.video_frame_point =
(TextView)row.findViewById(R.id.cb_video_length);

            row.setTag(holder);
        }
        else{
            holder = (CrossBrowserListHolder)row.getTag();
        }

        KeyFrame_attributes kf_attributes = keyframe_attr[position];
        holder.video_img.setImageBitmap(kf_attributes.video_img);
        holder.video_frame_point.setText(kf_attributes.frame_point);

        return row;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_frame_point Video frame point in TextView format
     * @param video_img         Video image in ImageView format
     */
    static class CrossBrowserListHolder{
        ImageView video_img;
        TextView video_frame_point;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.video_browser_thesis.activities;

import android.app.Activity;

/**
 * Displays all the matching key frames of a video
 * @author Anshul Saini
 * @version 5
 */

public class CrossBrowserActivity extends Activity{

    public static Boolean i = false;
    Bundle cb_bundle;

    private  Gallery crossbrowser_gallery;
    private DisplayMetrics metrics = new DisplayMetrics();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cross_browser);

        /**
         *  Retrieves data sent by the previous activity
```

```
         */
        cb_bundle = getIntent().getExtras();

        /**
         * Setting up the gallery view positioning
         * */
        getWindowManager().getDefaultDisplay().getMetrics(metrics);
        crossbrowser_gallery =
(Gallery)findViewById(R.id.cross_browser_gallery);
        MarginLayoutParams mlp1 = (MarginLayoutParams)
crossbrowser_gallery.getLayoutParams();
        setMargins(mlp1);

        CB_Processing cb_p = new
CB_Processing(cb_bundle.getString("video_id"));
        final KeyFrame_attributes[] keyframe_attr =
cb_p.getKeyFrameArray();

        final ListView crossbrowser_list = (ListView)
findViewById(R.id.cross_browser_listview);
        final ImageView selected_img =
(ImageView)findViewById(R.id.selected_matching_keyframe);

        CrossBrowser_ListAdapter cb_listAdapter = new
CrossBrowser_ListAdapter(this, keyframe_attr);
        final CircularListAdapter circularListAdapter = new
CircularListAdapter(cb_listAdapter);
        crossbrowser_list.setAdapter(circularListAdapter);
        crossbrowser_list.setOnItemClickListener(new
OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> arg0, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub
                CB_GalleryProcessing cb_gall = new
CB_GalleryProcessing(keyframe_attr[position].image_source);
                final KeyFrame_attributes[] keyframe_gall =
cb_gall.getKeyFrameArray();
                crossbrowser_gallery.setAdapter(new
CrossBrowser_GalleryAdapter(CrossBrowserActivity.this,keyframe_gall));
                crossbrowser_gallery.setSelection(1);

                crossbrowser_gallery.setOnItemClickListener(new
```

```
OnItemClickListener() {

                    @Override
                    public void onItemClick(AdapterView<?> arg0, View
arg1, int position_gall,
                            long arg3) {
                        // TODO Auto-generated method stub
                        String temp =
keyframe_gall[position_gall].image_source.substring(53,
keyframe_gall[position_gall].image_source.length());
                        String temp_video_id = temp.substring(0,
temp.indexOf("/"));
                        String temp_base_url = "http://
www.youtube.com/watch?v=";
                        String full_url = temp_base_url +
temp_video_id;
                        startActivity(new Intent(Intent.ACTION_VIEW,
Uri.parse(full_url)));
                    }
                });

                /**
                 * Setting Selected Video
                 */
                ImageProcessing processing_selected_img = new
ImageProcessing(keyframe_attr[position].image_source);

selected_img.setImageBitmap(processing_selected_img.decodeImage());
            }
        });

        /**
         * Sets the default content of the gallery
         */
        KeyFrame_attributes[] defaultList = new
KeyFrame_attributes[5];
        for(int i=0;i<5;i++){
            defaultList[i] = new KeyFrame_attributes(null,"",null);
        }
        crossbrowser_gallery.setAdapter(new
CrossBrowser_GalleryAdapter(this, defaultList));
        crossbrowser_gallery.setSelection(1);

    }
```

```
    /**
     * Sets the margins of top, bottom, right and left sides of the
galleryView
     */
    private void setMargins(MarginLayoutParams mlp){
        mlp.setMargins((metrics.widthPixels / 2 - 210), mlp.topMargin
+300,mlp.rightMargin, mlp.bottomMargin);
    }

    /**
     * Ends the current activity and returns to the previous activity
     */
    public void onListView_btn_click(View v){
        finish();
    }

    /**
     * Ends the current activity and returns to the main activity
     */
    public void onHomeButtonClick(View v){
        i = true;
        finish();
    }

    public String getValueFromBundle(){
        return cb_bundle.getString("video_id");
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.io.IOException;

import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

public class ImageProcessing {

    Bitmap bitmap_image;
    String imageUrl;

    public ImageProcessing(String imageUrl) {
        this.imageUrl = imageUrl;
        fetchImage();
    }

    /**
     * Fetches the image content from the provided URL
     *
     * @param bitmap    Stores the decoded image retrieved from the
provided url
     * @return bitmap
     * @see Bitmap Image
     * */
    private void fetchImage(){
        Bitmap bitmap = null;
        try {
              bitmap = BitmapFactory.decodeStream((InputStream)new
URL(imageUrl).getContent());
            } catch (MalformedURLException e) {
              e.printStackTrace();
            } catch (IOException e) {
              e.printStackTrace();
            }
        bitmap_image = bitmap;
    }
```

```
    public Bitmap decodeImage(){
        return bitmap_image;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.json;

import java.io.BufferedReader;

/**
 * Creates a JSON object from the data retrieved from the provided url
 * @author Anshul Saini
 * @version 5
 *
 * @param  is
 * @param  jObj     JSON Object which is finally created from the JSON
string
 * @param  json     JSON String which is used to create JSON Object

 * @return
 * @see
 */
public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    /**
     * Makes the HTTP request, parses the JSON String, creates and
JSON Object
     * @param httpClient
```

```
     * @param httpPost
     * @param httpResponse
     * @param httpEntity
     * @param reader
     * @param sb
     * @param line
     * @return jObj
     * @see JSONObject
     */
    public JSONObject getJSONFromUrl(String url) {

        DefaultHttpClient httpClient;
        HttpPost httpPost;
        HttpResponse httpResponse;
        HttpEntity httpEntity;
        BufferedReader reader;
        StringBuilder sb;
        String line;

        try {
            httpClient = new DefaultHttpClient();
            httpPost = new HttpPost(url);

            httpResponse = httpClient.execute(httpPost);
            httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            sb = new StringBuilder();
            line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
            is.close();
            json = sb.toString();
```

```
        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " +
e.toString());
        }

        try {
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " +
e.toString());
        }

        return jObj;
    }
}
```

```
package com.video_browser_thesis.elements;

import android.graphics.Bitmap;

/**
 * KeyFrame attributes
 * @author Anshul Saini
 * @version 5
 * @param image_source
 * @param frame_point
 * @param video_img
 *
 */
public class KeyFrame_attributes{
    public String image_source;
    public String frame_point;
    public Bitmap video_img;

    public KeyFrame_attributes(String image_source,String frame_point,
Bitmap video_img){
        this.image_source = image_source;
        this.frame_point = frame_point;
        this.video_img = video_img;
    }
}
```

```java
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a Keyframe Video Adapter which extends a Custom List
Adapter
 * @author Anshul Saini
 * @version 5
 */
public class Keyframe_ListAdapter extends
ArrayAdapter<Video_attributes>{

    Context context;
    static int layoutResourceId = R.layout.keyframe_list_layout;
    Video_attributes video_attr[] = null;

    public Keyframe_ListAdapter(Context context, Video_attributes[]
video_attr) {
        // TODO Auto-generated constructor stub
        super(context, layoutResourceId, video_attr);
        this.context = context;
        this.video_attr = video_attr;
    }

    /**
```

```
     * Inflates the gallery item with different types of views.
     */
    @Override
    public View getView(int position, View convertView, ViewGroup
parent) {
        View row = convertView;
        VideoListHolder holder = null;

        if(row == null)
        {
            LayoutInflater inflater =
((Activity)context).getLayoutInflater();
            row = inflater.inflate(layoutResourceId, parent, false);

            holder = new VideoListHolder();
            holder.video_title =
(TextView)row.findViewById(R.id.video_title);
            holder.video_description =
(TextView)row.findViewById(R.id.video_description);
            holder.uploaded_time =
(TextView)row.findViewById(R.id.video_uploaded_timestamp);
            holder.view_count =
(TextView)row.findViewById(R.id.view_count);
            holder.video_author =
(TextView)row.findViewById(R.id.video_author);
            holder.video_length =
(TextView)row.findViewById(R.id.video_length);
            holder.video_img =
(ImageView)row.findViewById(R.id.video_image);

            row.setTag(holder);
        }
        else{
            holder = (VideoListHolder)row.getTag();
        }

        Video_attributes video_attributes = video_attr[position];
        holder.video_title.setText(video_attributes.video_title);
        holder.video_author.setText(video_attributes.video_author);
        holder.uploaded_time.setText(video_attributes.uploaded_time);
        holder.view_count.setText(video_attributes.view_count);
        holder.view_count.setText(video_attributes.view_count);
        holder.video_length.setText(video_attributes.length);

holder.video_description.setText(video_attributes.description);
```

```
        if(video_attributes.video_image!=null){

holder.video_img.setImageBitmap(video_attributes.video_image);
        }

        return row;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_title       Video title in TextView format
     * @param video_description Video description in TextView format
     * @param uploaded_time     Video image in TextView format
     * @param view_count        Video image in TextView format
     * @param video_author      Video image in TextView format
     * @param video_length      Video image in TextView format
     * @param video_img         Video image in ImageView format
     */
    static class VideoListHolder
    {
        TextView video_title;
        TextView video_description;
        TextView uploaded_time;
        TextView view_count;
        TextView video_author;
        TextView video_length;
        ImageView video_img;
    }
}
```

```
package com.video_browser_thesis.activities;

import android.app.Activity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;

import com.video_browser_thesis.adapters.Keyframe_ListAdapter;
import com.video_browser_thesis.elements.Video_attributes;
import com.video_browser_thesis.processing.ImageProcessing;

/**
 * Displays all the key frames of a video
 * @author Anshul Saini
 * @version 5
 */
public class KeyframeListActivity extends Activity{

    Bundle b;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_keyframe);

        /**
         * Test Data
         */
        Video_attributes test[] = new Video_attributes[10];
        for(int i=0; i<test.length;i++){
            test[i] = new Video_attributes("Anshul Bit My
Finger","Anshul",null,null,null,null,"10:15",null,"1 year","504","I am
Awesome. Watch me Roll !!!",null);
        }
        /**
         * End test data
         */
```

```
        ListView keyframe_list = (ListView)
findViewById(R.id.matching_keyframes_list);
        keyframe_list.setAdapter(new Keyframe_ListAdapter(this,test));

        b = getIntent().getExtras();
        TextView video_title =
(TextView)findViewById(R.id.video_title);
        video_title.setText(b.getString("video_title"));

        TextView video_description =
(TextView)findViewById(R.id.video_description);
        video_description.setText(b.getString("video_description"));

        TextView video_upload_time =
(TextView)findViewById(R.id.video_uploaded_timestamp);
        video_upload_time.setText(b.getString("video_upload_time"));

        TextView view_count = (TextView)findViewById(R.id.view_count);
        view_count.setText(b.getString("video_view_count"));

        TextView video_author = (TextView)findViewById(R.id.author);
        video_author.setText(b.getString("video_author"));

        TextView video_length =
(TextView)findViewById(R.id.video_length);
        video_length.setText(b.getString("video_length"));

        ImageView video_thumbnail =
(ImageView)findViewById(R.id.video_keyframe_image);
        ImageProcessing ip = new
ImageProcessing(b.getString("video_img"));
        video_thumbnail.setImageBitmap(ip.decodeImage());
    }

    /**
     * Ends the current activity and returns to the previous activity
     */
    public void onCrossBrowser_btn_click(View v){
        Intent cb_intent = new
Intent(this,com.video_browser_thesis.activities.CrossBrowserActivity.c
lass);
        cb_intent.putExtras(b);
        startActivity(cb_intent);
    }
```

```
    public void onResume(){
        super.onResume();
        CrossBrowserActivity cb_i = new CrossBrowserActivity();
        if (cb_i.i == true){
            cb_i.i = false;
            finish();
        }
    }

    /**
     * Ends the current activity and returns to the main activity
     */
    public void onHomeButtonClick(View v){
        finish();
    }
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.adapters;

import android.app.Activity;

/**
 * Constructs a MainGallery Video Adapter which extends a Base Adapter
 * @author Anshul Saini
 * @version 5
 */
public class MainGallery_VideoAdapter extends BaseAdapter {

    private Activity activity;
    private static LayoutInflater inflater=null;
    Video_attributes video_attr[] = null;

    public MainGallery_VideoAdapter(Activity a,Video_attributes[]
v_attr) {
        activity = a;
        inflater =
(LayoutInflater)activity.getSystemService(Context.LAYOUT_INFLATER_SERV
ICE);
        this.video_attr = v_attr;
    }

    public int getCount() {
        return video_attr.length;
    }
```

```
    public Object getItem(int position) {
        return position;
    }

    public long getItemId(int position) {
        return position;
    }

    /**
     * Inflates the gallery item with different types of views.
     */
    public View getView(int position, View convertView, ViewGroup
parent) {
        View vi=convertView;
        ViewHolder holder;
        if(convertView==null){
            vi = inflater.inflate(R.layout.categorized_gallery_list,
null);
            holder=new ViewHolder();

holder.video_name=(TextView)vi.findViewById(R.id.video_name);

holder.video_author=(TextView)vi.findViewById(R.id.video_author);

holder.video_image=(ImageView)vi.findViewById(R.id.video_image);
            vi.setTag(holder);
        }
        else{
            holder=(ViewHolder)vi.getTag();
        }

        Video_attributes v_attr = video_attr[position];
        holder.video_name.setText(v_attr.video_title);
        holder.video_author.setText(v_attr.video_author);
        holder.video_image.setImageBitmap(v_attr.video_image);

        return vi;
    }

    /**
     * Holds all the elements of the view created in its super class
     * @param video_name    Video Frame point in TextView format
     * @param video_author  Video author in TextView format
     * @param video_author  Video image in ImageView format
```

```
     */
    public static class ViewHolder{
        public TextView video_name;
        public TextView video_author;
        public ImageView video_image;
    }
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini


 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.video_browser_thesis.activities;

import android.app.Activity;

/**
 * Displays all the videos in various categories
 * @author Anshul Saini
 * @version 5
 */
public class MainGalleryActivity extends Activity {
    private  Gallery galleryView1,galleryView2,galleryView3,
galleryView4,galleryView5;
    private DisplayMetrics metrics = new DisplayMetrics();

    private static final String TAG_CATEGORY01 =
"charlie_bit_my_finger";
    private static final String TAG_CATEGORY02 = "golden_voice";
    private static final String TAG_CATEGORY03 =
"samsung_galaxy_s3_ad";
    private static final String TAG_CATEGORY04 = "zinedine_zidane";
    private static final String TAG_CATEGORY05 =
"ray_william_johnson";

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_categorized_video_gallery);

        getWindowManager().getDefaultDisplay().getMetrics(metrics);

        galleryView1 = (Gallery)findViewById(R.id.galleryid1);
        galleryView2 = (Gallery)findViewById(R.id.galleryid2);
        galleryView3 = (Gallery)findViewById(R.id.galleryid3);
        galleryView4 = (Gallery)findViewById(R.id.galleryid4);
        galleryView5 = (Gallery)findViewById(R.id.galleryid5);

        TextView c1 = (TextView)findViewById(R.id.category01);
        TextView c2 = (TextView)findViewById(R.id.category02);
        TextView c3 = (TextView)findViewById(R.id.category03);
        TextView c4 = (TextView)findViewById(R.id.category04);
        TextView c5 = (TextView)findViewById(R.id.category05);

        MarginLayoutParams mlp1 = (MarginLayoutParams)
galleryView1.getLayoutParams();
        MarginLayoutParams mlp2 = (MarginLayoutParams)
galleryView2.getLayoutParams();
        MarginLayoutParams mlp3 = (MarginLayoutParams)
galleryView3.getLayoutParams();
        MarginLayoutParams mlp4 = (MarginLayoutParams)
galleryView4.getLayoutParams();
        MarginLayoutParams mlp5 = (MarginLayoutParams)
galleryView5.getLayoutParams();

        setMargins(mlp1);
        setMargins(mlp2);
        setMargins(mlp3);
        setMargins(mlp4);
        setMargins(mlp5);

        MainGalleryProcessing mg_p1 = new
MainGalleryProcessing(TAG_CATEGORY01);
        MainGalleryProcessing mg_p2 = new
MainGalleryProcessing(TAG_CATEGORY02);
        MainGalleryProcessing mg_p3 = new
MainGalleryProcessing(TAG_CATEGORY03);
        MainGalleryProcessing mg_p4 = new
MainGalleryProcessing(TAG_CATEGORY04);
        MainGalleryProcessing mg_p5 = new
MainGalleryProcessing(TAG_CATEGORY05);
```

```
        final Video_attributes[] v_attr_01 =
mg_p1.getCategoryVideos();
        final Video_attributes[] v_attr_02 =
mg_p2.getCategoryVideos();
        final Video_attributes[] v_attr_03 =
mg_p3.getCategoryVideos();
        final Video_attributes[] v_attr_04 =
mg_p4.getCategoryVideos();
        final Video_attributes[] v_attr_05 =
mg_p5.getCategoryVideos();

        galleryView1.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_01));
        galleryView2.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_02));
        galleryView3.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_03));
        galleryView4.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_04));
        galleryView5.setAdapter(new
MainGallery_VideoAdapter(this,v_attr_05));

        galleryView1.setSelection(1);
        galleryView2.setSelection(1);
        galleryView3.setSelection(1);
        galleryView4.setSelection(1);
        galleryView5.setSelection(1);

        c1.setText(TAG_CATEGORY01.replace("_"," ").toUpperCase());
        c2.setText(TAG_CATEGORY02.replace("_"," ").toUpperCase());
        c3.setText(TAG_CATEGORY03.replace("_"," ").toUpperCase());
        c4.setText(TAG_CATEGORY04.replace("_"," ").toUpperCase());
        c5.setText(TAG_CATEGORY05.replace("_"," ").toUpperCase());

        galleryView1.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_01, position);
            }
```

```
        });

        galleryView2.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_02, position);
            }
        });

        galleryView3.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_03, position);
            }
        });

        galleryView4.setOnItemClickListener(new OnItemClickListener()
{

            @Override
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_04, position);
            }
        });

        galleryView5.setOnItemClickListener(new OnItemClickListener()
{

            @Override
```

```
            public void onItemClick(AdapterView<?> parent, View arg1,
int position,
                    long arg3) {
                // TODO Auto-generated method stub

                passingBundles(v_attr_05, position);
            }
        });
    }


    private void setMargins(MarginLayoutParams mlp){
        mlp.setMargins(-(metrics.widthPixels / 2 + 155),
mlp.topMargin,
                mlp.rightMargin, mlp.bottomMargin);
    }

    /**
     * Creates a bundle of data to pass it to the another activity
     *
     * @param galleryIntent New Intent
     * @param extras        The bundle which carries all data from the
galleryIntent
     * @return
     * */
    private void passingBundles(Video_attributes[] v_arr,int position)
{
        Intent galleryIntent = new
Intent(getApplicationContext(),com.video_browser_thesis.activities.Key
frameListActivity.class);
        Bundle extras = new Bundle();

        extras.putString("video_title", v_arr[position].video_title);
        extras.putString("video_description",
v_arr[position].description);
        extras.putString("video_upload_time",
v_arr[position].uploaded_time);
        extras.putString("video_view_count",
v_arr[position].view_count);
        extras.putString("video_author",
v_arr[position].video_author);
        extras.putString("video_img", v_arr[position].thumbnail);
        extras.putString("video_length", v_arr[position].length);
        extras.putString("video_id", v_arr[position].video_id);
        extras.putString("video_url",v_arr[position].player_url);
```

```
        galleryIntent.putExtras(extras);
        startActivity(galleryIntent);
    }

    /**
     * Does nothing onHomeButton click as it is already in the home
activity
     * @param v
     */
    public void onHomeButtonClick(View v){
        //Do Nothing
    }
}
```

```java
package com.video_browser_thesis.processing;

import java.util.ArrayList;

public class MainGalleryProcessing implements Processing{
    private static String url;
    private String category_name;

    private static final String TAG_TITLE = "Title";
    private static final String TAG_AUTHOR = "Author";
    private static final String TAG_THUMBNAIL = "Thumbnail0";
    private static final String TAG_ID = "ID";
    private static final String TAG_VIDEOID = "VideoID";
    private static final String TAG_PLAYER_URL = "PlayerUrl";
    private static final String TAG_LENGTH = "Length";
    private static final String TAG_TOPICS = "Topics";
    private static final String TAG_UPLOADED_TIME = "Uploadtime";
    private static final String TAG_VIEW_COUNT = "ViewCount";
    private static final String TAG_DESCRIPTION = "Description";

    Video_attributes[] v_attr;

    public MainGalleryProcessing(String category_name){
        url = "http://cantabile.anu.edu.au/memeapp/cgi-bin/
metadataJSONOutput.py";
        this.category_name = category_name;
        setArrayContents();
    }

    /**
     * Organizes the data received from the URL in JSON format into a
HashMap
     *
     * @param  categoryList     Hash Map which contains all the
segregated metadata under different tag names
     * @param  jParser          An instance of JSONParser which parses
data from the provided URL
     * @param  c                An instance JSONObject which stores
all data contained within that JSON Object
     * @param  id               A temporary array to store video ids
     * @param  image_source     A temporary array to store video
images
     * @param  video_url        A temporary array to store video url
     * @param  video_name       A temporary array to store video name
     * @return categoryList
```

```
     * @see    Image
     */
    public ArrayList<HashMap<String, String>> jsonFeed(){

        ArrayList<HashMap<String, String>> categoryList = new
ArrayList<HashMap<String, String>>();
        JSONArray category = null;
        /** Creating JSON Parser instance */
        JSONParser jParser = new JSONParser();
        JSONObject c;
        String id, title, thumbnail, video_id, player_url, length,
author, topics, upload_time,view_count,description;
        HashMap<String, String> map;

        /** Getting JSON string from URL */
        JSONObject json = jParser.getJSONFromUrl(url);

        try {
            /** Getting Array of Contacts */
            category = json.getJSONArray(category_name);

            /** looping through All Contacts */
            for(int i = 0; i < category.length(); i++){
                c = category.getJSONObject(i);

                /** Storing each json item in variable */
                id = c.getString(TAG_ID);
                title = c.getString(TAG_TITLE);
                thumbnail = c.getString(TAG_THUMBNAIL);
                video_id = c.getString(TAG_VIDEOID);
                player_url = c.getString(TAG_PLAYER_URL);
                length = c.getString(TAG_LENGTH);
                author = c.getString(TAG_AUTHOR);
                topics = c.getString(TAG_TOPICS);
                upload_time = c.getString(TAG_UPLOADED_TIME);
                view_count = c.getString(TAG_VIEW_COUNT);
                description = c.getString(TAG_DESCRIPTION);

                /** Creating new HashMap */
                map = new HashMap<String, String>();

                /** adding each child node to HashMap key => value */
                map.put(TAG_ID, id);
                map.put(TAG_TITLE, title);
                map.put(TAG_THUMBNAIL, thumbnail);
```

```
                map.put(TAG_VIDEOID, video_id);
                map.put(TAG_PLAYER_URL, player_url);
                map.put(TAG_LENGTH, length);
                map.put(TAG_AUTHOR, author);
                map.put(TAG_TOPICS, topics);
                map.put(TAG_UPLOADED_TIME, upload_time);
                map.put(TAG_VIEW_COUNT, view_count);
                map.put(TAG_DESCRIPTION, description);

                /** Adding HashList to ArrayList */
                categoryList.add(map);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return categoryList;
    }


    /**
     * Segregates and organizes Metadata content
     *
     * @param
     * */
    public void setArrayContents(){

        ArrayList<HashMap<String, String>> category_list = jsonFeed();
        Video_attributes video_attr[] = new
Video_attributes[category_list.size()];

        for(int i=0; i<category_list.size();i++){

            ImageProcessing ip = new
ImageProcessing(category_list.get(i).get(TAG_THUMBNAIL).toString());

            video_attr[i] = new Video_attributes(
                    category_list.get(i).get(TAG_TITLE).toString(),
                    category_list.get(i).get(TAG_AUTHOR).toString(),
                    ip.decodeImage(),
                    category_list.get(i).get(TAG_ID).toString(),
                    category_list.get(i).get(TAG_VIDEOID).toString(),

category_list.get(i).get(TAG_PLAYER_URL).toString(),
                    category_list.get(i).get(TAG_LENGTH).toString(),
                    category_list.get(i).get(TAG_TOPICS).toString(),
```

```
category_list.get(i).get(TAG_UPLOADED_TIME).toString(),

category_list.get(i).get(TAG_VIEW_COUNT).toString(),

category_list.get(i).get(TAG_DESCRIPTION).toString(),

category_list.get(i).get(TAG_THUMBNAIL).toString());
        }
        v_attr = video_attr;
    }

    public Video_attributes[] getCategoryVideos(){
        return v_attr;
    }
}
```

```java
package com.video_browser_thesis.processing;

/**
 * @author Anshul Saini
 * @version 5
 */
import java.util.ArrayList;

import java.util.HashMap;

public interface Processing {
    void setArrayContents();
    ArrayList<HashMap<String, String>> jsonFeed();
}
```

```
/*
 * Copyright (C) 2012 Anshul Saini

 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */


package com.video_browser_thesis.splashScreen;
/**
 * @author Anshul Saini
 * @version 5
 */
import android.app.Activity;
/**
 * Main Splash Screen when the application is initiated
 *
 * @param  _active      Indicates if the splash screen is present
 * @param  _splashTime  The display time of the splash screen image in
milliseconds
 * @return
 * @see    Image
 */
public class SplashScreen extends Activity {

    protected boolean _active = true;
    protected int _splashTime = 2000;

    /**
     * Called when the activity is first created.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.splash_screen);

        /**
         * Threads for displaying the splash screen
         */
        Thread splashTread = new Thread() {
            @Override
            public void run() {
                try {
                    int waited = 0;
                    while(_active && (waited < _splashTime)) {
                        sleep(100);
                        if(_active) {
                            waited += 100;
                        }
                    }
                } catch(InterruptedException e) {

                } finally {

                    startActivity(new Intent(SplashScreen.this,
MainGalleryActivity.class));
                    finish();
                }
            }
        };
        splashTread.start();
    }
}
```

```
package com.video_browser_thesis.elements;

import android.graphics.Bitmap;
/**
 * Video_attributes
 * @author Anshul Saini
 * @version 5
 */
public class Video_attributes{
    public String video_title;
    public String video_author;
    public Bitmap video_image;
    public String thumbnail;
    public String tag_id;
    public String video_id;
    public String player_url;
    public String length;
    public String topics;
    public String uploaded_time;
    public String view_count;
    public String description;

    public Video_attributes(){
        super();
    }

    public Video_attributes(
```

```
            String video_title,
            String video_author,
            Bitmap video_image,
            String tag_id,
            String video_id,
            String player_url,
            String length,
            String topics,
            String uploaded_time,
            String view_count,
            String description,
            String thumbnail){
        super();
        this.tag_id = tag_id;
        this.video_title = video_title;
        this.video_author = video_author;
        this.video_image = video_image;
        this.video_id = video_id;
        this.player_url = player_url;
        this.length = length;
        this.topics = topics;
        this.uploaded_time = uploaded_time;
        this.view_count = view_count;
        this.description = description;
        this.thumbnail = thumbnail;
    }
}
```

activity_categorized_video_gallery.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <include
        android:id="@+id/header_bar"
        android:layout_alignParentTop="true"
        layout="@layout/app_header_layout" />

    <ScrollView
        android:id="@+id/scroll_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_below="@+id/header_bar"
        android:background="@color/
categorized_video_gallery_activity_background" >

        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical" >

            <include android:id="@+id/category01" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid1" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category02" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid2" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category03" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid3" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category04" layout="@layout/
category_heading_text_layout"/>
            <include android:id="@+id/galleryid4" layout="@layout/
category_gallery_layout"/>

            <include android:id="@+id/category05" layout="@layout/
```

```
category_heading_text_layout"/>
            <include android:id="@+id/galleryid5" layout="@layout/
category_gallery_layout"/>

        </LinearLayout>
    </ScrollView>
</RelativeLayout>
```

activity_cross_browser.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/cross_browser_background" >

    <include
        android:id="@+id/third_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/third_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:onClick="onListView_btn_click"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/cross_browser_tab_txt" />
    </LinearLayout>
```

```
    <ListView
        android:id="@+id/cross_browser_listview"
        android:layout_width="94dp"
        android:layout_height="fill_parent"
        android:layout_below="@+id/button_area"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="5dp"
        android:background="@color/cross_browser_listview_background"
        android:paddingBottom="5dp" />

    <Gallery
        android:id="@+id/cross_browser_gallery"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button_area"
        android:background="@color/cross_browser_gallery_background" /
>

    <TextView
        android:id="@+id/matching_keyFrame_txt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/selection_border01"
        android:layout_alignLeft="@+id/selection_border04"
        android:layout_marginLeft="5dp"
        android:background="#AA000000"
        android:paddingBottom="7dp"
        android:paddingLeft="30dp"
        android:paddingRight="15dp"
        android:text="@string/matching_keyFrame_txt"
        android:textColor="#ffffff"
        android:textStyle="bold"
        android:typeface="sans" />

    <TextView
        android:id="@+id/selection_border01"
        android:layout_width="fill_parent"
        android:layout_height="4dp"
        android:layout_alignTop="@+id/cross_browser_gallery"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border02"
        android:layout_width="fill_parent"
        android:layout_height="4dp"
```

```
        android:layout_alignBottom="@+id/cross_browser_gallery"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border03"
        android:layout_width="5dp"
        android:layout_height="72dp"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="150dp"
        android:background="@color/cross_browser_selection_border" />

    <TextView
        android:id="@+id/selection_border04"
        android:layout_width="5dp"
        android:layout_height="72dp"
        android:layout_alignRight="@+id/cross_browser_listview"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="150dp"
        android:background="@color/cross_browser_selection_border"
        android:paddingLeft="1dp" />

    <RelativeLayout
        android:id="@+id/selected_frame_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignLeft="@+id/selection_border04"
        android:layout_alignTop="@+id/selection_border02"
        android:layout_marginLeft="5dp"
        android:background="#000000"
        android:layout_marginTop="8dp">

        <TextView
            android:id="@+id/selected_matching_keyframe_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="55dp"
            android:layout_marginTop="30dp"
            android:background="#AA000000"
            android:paddingLeft="5dp"
            android:paddingRight="5dp"
            android:text="@string/selected_matching_keyframe_txt"
            android:textColor="#ffffff"
            android:textSize="16sp"
            android:textStyle="bold"
```

```
            android:typeface="sans" />

        <ImageView
            android:id="@+id/selected_matching_keyframe"
            android:layout_width="250dp"
            android:layout_height="250dp"
            android:scaleType="fitCenter"
            android:layout_below="@+id/selected_matching_keyframe_txt"
            android:contentDescription="@string/no_description"
            android:src="@drawable/splash_screen_img"
            android:layout_marginLeft="30dp" />
    </RelativeLayout>

</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

app_header_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="@color/app_header_background" >

        <ImageView
            android:id="@+id/app_header_homebutton"
            android:layout_width="@dimen/app_header_home_button"
            android:layout_height="@dimen/app_header_home_button"
            android:layout_alignParentLeft="true"
            android:layout_centerInParent="true"
            android:layout_marginLeft="5dp"
            android:contentDescription="@string/no_description"
            android:src="@drawable/home"
            android:onClick="onHomeButtonClick" />

        <ImageView
            android:id="@+id/app_header_img"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_centerInParent="true"
            android:layout_toRightOf="@+id/app_header_homebutton"
            android:contentDescription="@string/no_description"
            android:src="@drawable/app_header" />
    </RelativeLayout>

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="@dimen/
app_header_bottom_divider_thickness"
        android:background="@color/app_header_bottom_divider" />

</LinearLayout>
```

categorized_gallery_item_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/video_image"
        android:layout_width="130dp"
        android:layout_height="110dp"
        android:layout_gravity="center"
        android:contentDescription="@string/no_description"
        android:paddingRight="8dp"
        android:scaleType="fitXY"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/video_name"
        android:layout_width="112dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:layout_marginTop="70dp"
        android:background="@color/
video_attributes_overlayed_text_background"
        android:maxLength="17"
        android:maxLines="1"
        android:paddingLeft="1dp"
        android:text="@string/video_title_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textSize="12sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/video_author"
        android:layout_width="112dp"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_name"
        android:layout_below="@+id/video_name"
        android:background="@color/
video_attributes_overlayed_text_background"
        android:maxLength="17"
        android:maxLines="1"
        android:paddingLeft="1dp"
        android:text="@string/video_author_txt"
```

```
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textSize="10sp"
        android:textStyle="italic" />

</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

activity_keyframe.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#8b8b83" >

    <include
        android:id="@+id/second_activity_header"
        layout="@layout/app_header_layout" />

    <LinearLayout
        android:id="@+id/button_area"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/second_activity_header"
        android:layout_marginTop="3dp"
        android:orientation="horizontal" >

        <Button
            android:id="@+id/keyframe_listview_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginLeft="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/darker_gray"
            android:text="@string/listView_tab_txt" />

        <View
            android:id="@+id/button_divider"
            android:layout_width="0dp"
            android:layout_height="5dp"
            android:layout_weight="0.01" />

        <Button
            android:id="@+id/key_frame_crossbrowser_btn"
            android:layout_width="0dp"
            android:layout_height="40dp"
            android:layout_marginRight="10dp"
            android:layout_weight="0.5"
            android:background="@android:color/background_light"
            android:text="@string/cross_browser_tab_txt"
            android:onClick="onCrossBrowser_btn_click" />
    </LinearLayout>
```

```
    <RelativeLayout
        android:id="@+id/main_keyframe_area"
        android:layout_width="fill_parent"
        android:layout_height="120dp"
        android:layout_below="@+id/button_area"
        android:layout_marginTop="5dp"
        android:background="#000000" >

        <ImageView
            android:id="@+id/video_keyframe_image"
            android:layout_width="130dp"
            android:layout_height="110dp"
            android:layout_gravity="center"
            android:contentDescription="@string/no_description"
            android:paddingRight="8dp"
            android:scaleType="fitXY"
            android:src="@drawable/ic_launcher" />

        <TextView
            android:id="@+id/video_length"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="75dp"
            android:layout_marginTop="90dp"
            android:background="@color/category_text_background"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:text="@string/video_time_length" />

        <TextView
            android:id="@+id/video_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="13dp"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="1"
            android:text="@string/video_title_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:textStyle="bold"
            android:typeface="normal" />

        <TextView
```

```
            android:id="@+id/video_description"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_title"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_description_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal" />

        <TextView
            android:id="@+id/video_uploaded_timestamp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_description"
            android:layout_toRightOf="@+id/video_keyframe_image"
            android:maxLines="2"
            android:text="@string/video_uploaded_timestamp_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color"
            android:typeface="normal"
            android:maxLength="10" />

        <TextView
            android:id="@+id/upload_viewcount_divider"
            android:layout_width="2dp"
            android:layout_height="13dp"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_marginTop="1dp"
            android:layout_toRightOf="@+id/video_uploaded_timestamp"
            android:background="#ffffff" />

        <TextView
            android:id="@+id/view_count"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/upload_viewcount_divider"
            android:maxLines="1"
            android:text="@string/video_view_count_txt"
```

```
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/views_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_description"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/view_count"
            android:maxLines="1"
            android:text="@string/video_views_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/by_txt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignLeft="@+id/video_title"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:text="@string/by_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />

        <TextView
            android:id="@+id/author"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/video_uploaded_timestamp"
            android:layout_marginLeft="3dp"
            android:layout_toRightOf="@+id/by_txt"
            android:text="@string/video_author_txt"
            android:textColor="@color/
video_attributes_overlayed_text_color" />
    </RelativeLayout>

    <ListView
        android:id="@+id/matching_keyframes_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/main_keyframe_area"
        android:layout_marginTop="5dp"
        android:background="@color/keyframe_listview" />
```

```
</RelativeLayout>
```

cross_browser_galleryitem.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <ImageView
        android:id="@+id/cross_browsers_gallery_img"
        android:layout_width="90dp"
        android:layout_height="71dp"
        android:contentDescription="@string/no_description"
        android:src="@drawable/ic_launcher"
        android:layout_marginTop="1dp"
        android:layout_marginBottom="1dp"
        android:paddingLeft="10dp"
        android:paddingRight="10dp" />

    <TextView
        android:id="@+id/cb_gallery_video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="45dp"
        android:background="#A0000000"
        android:text="@string/video_time_length"
        android:textColor="@android:color/white"
        android:textSize="8sp"/>

</RelativeLayout>
```

cross_browser_listitem.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <ImageView
        android:id="@+id/cross_browsers_img"
        android:layout_width="90dp"
        android:layout_height="71dp"
        android:contentDescription="@string/no_description"
        android:src="@drawable/ic_launcher"
        android:layout_marginTop="1dp"
        android:layout_marginBottom="1dp" />

    <TextView
        android:id="@+id/cb_video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="45dp"
        android:background="#A0000000"
        android:text="@string/video_time_length"
        android:textColor="@android:color/white" />

</RelativeLayout>
```

keyframe_list_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:id="@+id/LinearLayout01"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:padding="3sp" >

    <include layout="@layout/keyframe_listview" />

</LinearLayout>
```

keyframe_listview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:id="@+id/main_keyframe_area"
    android:layout_width="fill_parent"
    android:layout_height="80dp"
    android:layout_below="@+id/button_area"
    android:layout_marginTop="10dp"
    android:background="#484848" >

    <ImageView
        android:id="@+id/video_keyframe_image"
        android:layout_width="90dp"
        android:layout_height="70dp"
        android:layout_gravity="center"
        android:contentDescription="@string/no_description"
        android:paddingRight="8dp"
        android:scaleType="fitXY"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/video_length"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="35dp"
        android:layout_marginTop="50dp"
        android:background="@color/category_text_background"
        android:text="@string/video_time_length"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/video_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/video_keyframe_image"
        android:layout_marginLeft="15dp"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="1"
        android:text="@string/video_title_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:textStyle="bold"
        android:typeface="normal" />
```

```
    <TextView
        android:id="@+id/video_description"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_title"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="2"
        android:text="@string/video_description_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:typeface="normal" />

    <TextView
        android:id="@+id/video_uploaded_timestamp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_description"
        android:layout_toRightOf="@+id/video_keyframe_image"
        android:maxLines="2"
        android:text="@string/video_uploaded_timestamp_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color"
        android:typeface="normal" />

    <TextView
        android:id="@+id/upload_viewcount_divider"
        android:layout_width="2dp"
        android:layout_height="13dp"
        android:layout_below="@+id/video_description"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="1dp"
        android:layout_toRightOf="@+id/video_uploaded_timestamp"
        android:background="#ffffff" />

    <TextView
        android:id="@+id/view_count"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/video_description"
        android:layout_marginLeft="3dp"
        android:layout_toRightOf="@+id/upload_viewcount_divider"
        android:maxLines="1"
```

```
        android:text="@string/video_view_count_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

    <TextView
        android:id="@+id/by_txt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/video_title"
        android:layout_below="@+id/video_uploaded_timestamp"
        android:text="@string/by_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

    <TextView
        android:id="@+id/video_author"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/video_uploaded_timestamp"
        android:layout_marginLeft="3dp"
        android:layout_toRightOf="@+id/by_txt"
        android:text="@string/video_author_txt"
        android:textColor="@color/
video_attributes_overlayed_text_color" />

</RelativeLayout>
```

splash_screen.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/splash_background"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/splash_screen_img"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:contentDescription="@string/no_description"
        android:src="@drawable/splash_screen_img" />

</RelativeLayout>
```

color.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <!-- Application main header colors -->
    <color name="app_header_background">#000000</color>
    <color name="app_header_bottom_divider">#00ffff</color>

    <!-- Categorized video gallery activity -->
    <color
name="categorized_video_gallery_activity_background">#8b8b83</color>
    <color name="category_text_color">#fffafa</color>
    <color name="category_text_background">#A0000000</color>
    <color name="categorized_gallery_background">#484848</color>
    <color
name="video_attributes_overlayed_text_background">#AA000000</color>
    <color name="video_attributes_overlayed_text_color">#ffffff</
color>

    <!-- Keyframe Activity -->
    <color name="keyframe_listview">#696969</color>

    <!-- Cross Browser -->
    <color name="cross_browser_background">#8b8b83</color>
    <color name="cross_browser_listview_background">#484848</color>
    <color name="cross_browser_gallery_background">#484848</color>
    <color name="cross_browser_selection_border">#00bfff</color>
</resources>
```

dimens.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <!-- Application Header elements dimensions -->
    <dimen name="app_header_home_button">50dp</dimen>
    <dimen name="app_header_bottom_divider_thickness">2dp</dimen>
</resources>
```

strings.xml

```
<resources>

    <string name="app_name">Video_Browser</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string
name="title_activity_categorized_video_gallery">CategorizedVideoGaller
y</string>

    <!-- For all image descriptions -->
    <string name="no_description">No Description</string>

    <!-- Category default names on home activity -->
    <string name="category_txt">Category Name</string>

    <!-- For all video elements -->
    <string name="video_title_txt">Video Title</string>
    <string name="video_author_txt">Video Author</string>
    <string name="video_description_txt">Video Description</string>
    <string name="video_uploaded_timestamp_txt">Uploaded Time</string>
    <string name="video_view_count_txt">View Count</string>
    <string name="video_views_txt">views</string>
    <string name="by_txt">by</string>

    <!-- KeyFrame Activity -->
    <string name="listView_tab_txt">Listview</string>
    <string name="cross_browser_tab_txt">Cross Browser</string>
    <string name="video_time_length">Length</string>

    <!-- Matching keyframes -->
    <string name="matching_keyFrame_txt">Matching KeyFrames:</string>
    <string name="selected_matching_keyframe_txt">Selected KeyFrame:</
string>
</resources>
```

styles.xml

```
<resources>

    <style name="AppTheme" parent="android:Theme.Light" />

</resources>
```

project.properties

```
# This file is automatically generated by Android Tools.
# Do not modify this file -- YOUR CHANGES WILL BE ERASED!
#
# This file must be checked in Version Control Systems.
#
# To customize properties used by the Ant build system edit
# "ant.properties", and override values to adapt the script to your
# project structure.
#
# To enable ProGuard to shrink and obfuscate your code, uncomment this
(available properties: sdk.dir, user.home):
#proguard.config=${sdk.dir}/tools/proguard/proguard-
android.txt:proguard-project.txt

# Project target.
target=android-10
```

```
import sqlite3
import sqlite3 as lite
import os
import glob
import ConfigParser
import re
import sys
import json
import unicodedata
from xml.dom.minidom import parseString

global xmlperkeyword

try:
    cf = ConfigParser.ConfigParser()
        cf.read('./config.ini')
        xmlperkeyword = int(cf.get('NUMBERS', 'xml_per_keyword'))
except Exception,e:
        print e
        print 'Error: Invalid config file.'
        exit(1)

#print '-------------------------------------------------------------------------'
#print 'START CRAWLXML.PY'
#print '-------------------------------------------------------------------------'
#os.chdir(os.path.expanduser("~/data/crawling/"))
#os.system('python2.7 ./crawlxml.py -i')

#print '-------------------------------------------------------------------------'
#print 'START CRAWLVIDEO.PY'
#print '-------------------------------------------------------------------------'
#os.system('python2.7 ./crawlvideo.py -i')

#print '-------------------------------------------------------------------------'
#print 'START BATCHRUN.PY'
#print '-------------------------------------------------------------------------'
#os.chdir(os.path.expanduser("~/data/videosplitting/"))
#os.system('python2.7 ./batchrun.py -i')

# connect to database
os.chdir(os.path.expanduser("~/data/www/cgi-bin/"))
conn = sqlite3.connect('videos.sqlite3')
c = conn.cursor()


#c.execute ('''DROP TABLE IF EXISTS topics''')
#c.execute ('''DROP TABLE IF EXISTS videosMetaData''')
#c.execute ('''DROP TABLE IF EXISTS imageFrames''')
```

```
#c.execute ('''DROP TABLE IF EXISTS matchingFrames''')

c.execute('''CREATE TABLE topics (id INTEGER PRIMARY KEY  AUTOINCREMENT  NOT N

c.execute('''CREATE TABLE videosMetaData
(id INTEGER PRIMARY KEY  AUTOINCREMENT  NOT NULL , title TEXT NOT NULL , videc
query TEXT, rank INTEGER, uploadtime DATETIME, description TEXT, category TEXT
tags TEXT, playerurl TEXT, flashplayerurl TEXT, length DOUBLE, author TEXT, cc
viewcount INTEGER, ratingavg TEXT,thumbnail0 TEXT, thumbnail1 TEXT, thumbnail2
FOREIGN KEY(topics) REFERENCES topics(id))''')

c.execute('''CREATE TABLE imageFrames
(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, imageUrl TEXT, imageName TEXT,
FOREIGN KEY (videoRef) REFERENCES videoMetaData (videoID))''')

c.execute('''CREATE TABLE matchingFrames
(id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, referenceFrameID TEXT, matchir


# inserts the topics into the topics table (excludes trash directory)

print '-------------------------------------------------------------------------
print 'ADDING ROWS TO TOPICS (CATEGORY) TABLE '
print '-------------------------------------------------------------------------
os.chdir(os.path.expanduser("~/data/crawling/topics/"))
for files in glob.glob("*"):
    if not files.startswith('trash') and not len(files) == 5 and not files.end
        c.execute ("INSERT INTO topics (name) VALUES(?)", (files,))

def count_letters(word):
    count = 0
    for c in word:
        count = count + 1
    return count

def listSubDir(filename, subdirectory=''):
    if subdirectory:
        path = subdirectory
    else:
        path = os.getcwd()
    for root, dirs, names in os.walk(path):
    if os.path.exists (os.path.join(root, filename)):
        return os.path.join(root, filename)


# inserts matching keyframes into the matchingFrame table

print '-------------------------------------------------------------------------
```

```
print 'ADDING ROWS TO MATCHING FRAME TABLE '
print '-------------------------------------------------------------------------
mKFramePath = os.path.expanduser("~/data/www/memexplore/matchingKeyFramesJSON/
baseUrl = "http://cantabile.anu.edu.au/memexplore/"
os.chdir(mKFramePath)
for path,subdirs,files in os.walk(mKFramePath):
    for file in files:
        filePath = listSubDir(file,mKFramePath)
        json_data=open(filePath).read()
        data = json.loads(json_data)
        ok = data["FileMatches"]
        referenceFileName = unicodedata.normalize('NFKD', data["FileName"]).er
        for i in range(len(ok)):
            c.execute ("INSERT INTO matchingFrames(referenceFrameID, matchingF
        #json_data.close();



# inserts images into imageFrames table in binary format

print '-------------------------------------------------------------------------
print 'ADDING IMAGES TO IMAGEFRAMES TABLE '
print '-------------------------------------------------------------------------
imageFrameDir = os.path.expanduser("~/data/www/memexplore/shot1Test/")
os.chdir(imageFrameDir)
for path,subdirs, files in os.walk(imageFrameDir):
    for subDir in subdirs:
        if (count_letters(subDir) > 1):
            storeVideoId = subDir
    for file in files:
        #print file
        filepath = listSubDir(file,imageFrameDir)
        fin = None
        try:
            fin = open (filepath,'rb')
            img = fin.read()
            binary = lite.Binary(img)
            c.execute ("INSERT INTO imageFrames(imageUrl,imageName,videoRef) \
            #print (filepath)
        except IOError, e:
            print "Error %d: %s" % (e.args[0],e.args[1])
            ok123 = "error"
            print ok123
        finally:
            if fin:
                fin.close()
```

```
# inserts metadata into the metadata table

print '-------------------------------------------------------------------------'
print 'ADDING ROWS TO XML METADATA TABLE '
print '-------------------------------------------------------------------------'
x = 0
previous = 0

root = os.path.expanduser("~/data/crawling/xml/raw/")
os.chdir(root)
for path,subdirs,files in os.walk(root):
    for file in files:
    xmlFilePath = listSubDir(file,root)
    if '/relevance/' not in xmlFilePath:
        xmlFilePath = "notValid"
    else:
        new = xmlFilePath.find('/relevance/')
    if previous != new:
        x = x + 1
        previous = new
    try:

        f=open(xmlFilePath, 'r')
        data = f.read()
        dom = parseString(data)
        try:
            xmlTitle = dom.getElementsByTagName('title')[0].toxml()
            titleData = xmlTitle.replace('<title>','').replace('</title>','')
        except IndexError:
            titleData = None

        try:
            xmlQuery = dom.getElementsByTagName('query')[0].toxml()
            queryData = xmlQuery.replace('<query>','').replace('</query>','')
        except IndexError:
            queryData = None

        try:
            xmlRank = dom.getElementsByTagName('rank')[0].toxml()
            rankData = xmlRank.replace('<rank>','').replace('</rank>','')
        except IndexError:
            rankData = None

        try:
            xmlUploadtime = dom.getElementsByTagName('uploadtime')[0].toxml()
```

```
        uploadtimeData = xmlUploadtime.replace('<uploadtime>','').replace(
    except IndexError:
        uploadtimeData = None

    try:
        xmlDescription = dom.getElementsByTagName('description')[0].toxml(
        descriptionData = xmlDescription.replace('<description>','').repla
    except IndexError:
        descriptionData = None

    try:
        xmlCategory = dom.getElementsByTagName('category')[0].toxml()
        categoryData = xmlCategory.replace('<category>','').replace('</cat
    except IndexError:
        categoryData = None

    try:
        xmlTags = dom.getElementsByTagName('tags')[0].toxml()
        tagsData = xmlTags.replace('<tags>','').replace('</tags>','')
    except IndexError:
        tagsData = None

    try:
        xmlPlayerurl = dom.getElementsByTagName('palyerurl')[0].toxml()
        playerurlData = xmlPlayerurl.replace('<palyerurl>','').replace('</
    except IndexError:
        playerurlData = None

    try:
        xmlFlashplayerurl = dom.getElementsByTagName('flashpalyerurl')[0].
        flashplayerurlData = xmlFlashplayerurl.replace('<flashpalyerurl>',
    except IndexError:
        flashplayerurl = None

    try:
        xmlLength = dom.getElementsByTagName('length')[0].toxml()
        lengthData = xmlLength.replace('<length>','').replace('</length>',
    except IndexError:
        lengthData = None

    try:
        xmlAuthor = dom.getElementsByTagName('author')[0].toxml()
        authorData = xmlAuthor.replace('<author>','').replace('</author>',
    except IndexError:
        authorData = None

    try:
        xmlComment_count = dom.getElementsByTagName('comment_count')[0].to
```

```
        comment_countData = xmlComment_count.replace('<comment_count>','')
    except IndexError:
        comment_countData = None

    try:
        xmlFavorite_count = dom.getElementsByTagName('favorite_count')[0].
        favorite_countData = xmlFavorite_count.replace('<favorite_count>',
    except IndexError:
        favorite_countData = None

    try:
        xmlViewcount = dom.getElementsByTagName('viewcount')[0].toxml()
        viewcountData = xmlViewcount.replace('<viewcount>','').replace('</
    except IndexError:
        viewcountData = None

    try:
        xmlRating_avg = dom.getElementsByTagName('rating_avg')[0].toxml()
        rating_avgData = xmlRating_avg.replace('<rating_avg>','').replace(
    except IndexError:
        rating_avgData = None

    try:
        xmlThumbnail0 = dom.getElementsByTagName('thumbnail')[0].toxml()
        thumbnailData0 = xmlThumbnail0.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData0 = None

    try:
        xmlThumbnail1 = dom.getElementsByTagName('thumbnail')[1].toxml()
        thumbnailData1 = xmlThumbnail1.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData1 = None

    try:
        xmlThumbnail2 = dom.getElementsByTagName('thumbnail')[2].toxml()
        thumbnailData2 = xmlThumbnail2.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData2 = None

    try:
        xmlThumbnail3 = dom.getElementsByTagName('thumbnail')[3].toxml()
        thumbnailData3 = xmlThumbnail3.replace('<thumbnail>','').replace('
    except IndexError:
        thumbnailData3 = None

    try:
        videoIDData = playerurlData[31:42]
```

```
        except IndexError:
            videoIDData = None

        topic = x

        c.execute('''INSERT INTO videosMetaData (title,videoID,query,rank,uplo
        f.close()
    except IOError, e:
        ok = "testtt";

c.execute('pragma foreign_keys = on')

# save (commit) the changes
conn.commit()

# close connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json
import binascii
import base64

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')

# connect to Database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
ImageSource = '"%s"' % "ImageSource"
Name = '"%s"' % "ImageName"
VideoRef = '"%s"' % "VideoRef"
tempStr = "["

# perform SQL query and print JSON
cursor.execute ("SELECT MAX (imageName) FROM imageFrames WHERE videoRef = ?",(
lastEntryInEachCategory = cursor.fetchone()[0]
print "{\n\t"
print '"%s"' % "key_frame" + ": [\n"
for cg in cursor.execute('SELECT * FROM imageFrames WHERE videoRef = ? ORDER E
    print json.dumps({'ID':"%s" % cg[0],'ImageSource':"%s" % cg[1],'Name':cg[2
    if cg[2] != lastEntryInEachCategory:
        print ","
print "\n\t]\n}"

# close Database connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json
import string

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')
#last_name  = form.getvalue('last_name')

# connect to Database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
VideoTitle = '"%s"' % "Title"
VideoID = '"%s"' % "VideoID"
Query = '"%s"' % "Query"
Rank = '"%s"' % "Rank"
UploadTime = '"%s"' % "Uploadtime"
Description = '"%s"' % "Description"
Category = '"%s"' % "Category"
Tags = '"%s"' % "Tags"
PlayerUrl = '"%s"' % "PlayerUrl"
FlashPlayerUrl = '"%s"' % "FlashPlayerUrl"
Length = '"%s"' % "Length"
Author = '"%s"' % "Author"
CommentCount = '"%s"' % "CommentCount"
FavoriteCount = '"%s"' % "FavoriteCount"
ViewCount = '"%s"' % "ViewCount"
RatingAvg = '"%s"' % "RatingAvg"
Thumbnail0 = '"%s"' % "Thumbnail0"
Thumbnail1 = '"%s"' % "Thumbnail1"
Thumbnail2 = '"%s"' % "Thumbnail2"
```

```
Thumbnail3 = '"%s"' % "Thumbnail3"
Topics = '"%s"' % "Topics"

print "{"

# iterative query for creating JSON
def iterateSqlQuery (parameters,test,idd):
    for names in cursor.execute("SELECT * FROM topics WHERE id = ?", (paramete
        tempStr = "\t" + '"%s"' % names[1] + ": ["
        cursor.execute ("SELECT MAX (id) FROM videosMetaData WHERE topics =?",
        lastEntryInEachCategory = cursor.fetchone()[0]
        for cg in cursor.execute("SELECT * FROM videosMetaData WHERE topics =?
            id = Id + ": " + '"%s"' % str(cg[0]) + ", " + "\n"
            videoTitle = VideoTitle + ": " + '"%s"' % cg[1] + ", " + "\n"
            videoUniqueID = VideoID + ": " + '"%s"' % cg[2] + ", " + "\n"
            uploadTime = UploadTime + ": " + '"%s"' % cg[5] + ", " + "\n"
            description = Description + ": " + '"%s"' % cg[6] + ", " + "\n"
            playerurl = cg[9]
            playerurl = PlayerUrl + ": " + '"%s"' % playerurl[:-33] + ", " + '
            length = Length + ": " + '"%s"' % str(cg[11]) + ", " + "\n"
            author = Author + ": " + '"%s"' % cg[12] + ", " + "\n"
            viewCount = ViewCount + ": " + '"%s"' % cg[15] + ", " + "\n"
            thumbnail0 = Thumbnail0 + ": " + '"%s"' % cg[17] + ", " + "\n"
            topics = Topics + ": " + '"%s"' % str(cg[21]) + "\n"
            tempStr = tempStr + "\n\t\t" + "{" + "\n\t\t\t" + id + "\t\t\t" +
            if len(id) > 0 and cg[0] == lastEntryInEachCategory:
                tempStr = tempStr[:-2]
        if (test == idd):
            tempStr = tempStr + "\n\t]"
        else:
            tempStr = tempStr + "\n\t],"
    print filter(lambda x: x in string.printable, tempStr)  # filters out non-
    test = test + 1
    if test <= idd:
        iterateSqlQuery(parameters + 1,test,idd)
    return

cursor.execute ("SELECT MAX (id) FROM topics")
lastID = cursor.fetchone()[0]
iterateSqlQuery(1,1,lastID)

print "}"

# close database connection
conn.close()
```

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Import modules for CGI handling
import cgi
import sqlite3
import json

# enable debugging
import cgitb
cgitb.enable()

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
qword = form.getvalue('q')

# Connect to the database
conn = sqlite3.connect('videos.sqlite3')
cursor = conn.cursor()

print "Content-Type: text/plain;charset=utf-8"
print

# variables
Id = '"%s"' % "ID"
ReferenceFrameID = '"%s"' % "ReferenceFrameID"
MatchingFrame = '"%s"' % "MatchingFrame"
MatchingFrameURL = '"%s"' % "MatchingFrameURL"
tempStr =  "{\n\t" + '"%s"' % "matching_key_frames" + ": [\n"

# Query the database
for cg in cursor.execute('SELECT * FROM matchingFrames WHERE referenceFrameID
    id = Id + ": " + '"%s"' % str(cg[0]) + ", " + "\n"
    referenceFrameID = ReferenceFrameID + ": " + '"%s"' % cg[1] + ", " + "\n"
    matchingFrame = MatchingFrame + ": " + '"%s"' % cg[2] + ", " + "\n"
    matchingFrameURL = MatchingFrameURL + ": " + '"%s"' % cg[3] + "\n"
    tempStr = tempStr + "\n" + "\t" + "\t" + "{" + "\n" + "\t" + "\t" + "\t" +

tempStr = tempStr[:-2] + "\n\t]\n}"

# output JSON
print tempStr

# Close connection to Database
conn.close()
```