­­­

Video Tag Portal

System Design Document | Current Version [0.3.0]

Prepared By:

Anudeep Potlapally

Travis Rous

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Authors** | **Version** | **Comments** |
| 9/20/12 | Travis Rous & Anudeep Potlapally | 1.0.0 | Initial version |
| 10/4/12 | Travis Rous & Anudeep Potlapally | 1.1.0 | Added prototype 2 |
| 11/2/12 | Travis Rous & Anudeep Potlapally | 1.2.0 | Added prototype 2 |
| 12/6/12 | Travis Rous & Anudeep Potlapally | 1.3.0 | Added prototype 3 |

Table of Contents

[1.0 Overview 5](#_Toc354068643)

[1.1 Scope 5](#_Toc354068644)

[1.2 Purpose 5](#_Toc354068645)

[1.2.1 Major System Component #1: Database 5](#_Toc354068646)

[1.2.2 Major System Component #2: APIs 5](#_Toc354068647)

[1.2.3 Major System Component #3: YouTube and Vimeo 5](#_Toc354068648)

[1.2.4 Major System Component #4: Popcorn.js 5](#_Toc354068649)

[1.3 Systems Goals 6](#_Toc354068650)

[1.4 System Overview and Diagram 6](#_Toc354068651)

[1.5 Technologies Overview 7](#_Toc354068652)

[2.0 Project Overview 8](#_Toc354068653)

[2.1 Team Members and Roles 8](#_Toc354068654)

[2.2 Project Management Approach 8](#_Toc354068655)

[2.3 Sprint Overview 8](#_Toc354068656)

[2.3.1 Sprint 1: Initial prototype 9](#_Toc354068657)

[2.3.2 Sprint 2: Database Design 9](#_Toc354068658)

[2.3.3 Sprint 3: Design and Connect pages 9](#_Toc354068659)

[2.3.4 Sprint 4: Giving the users an attractive interface 9](#_Toc354068660)

[3.0 Requirements 10](#_Toc354068661)

[4.0 Design and Implementation 10](#_Toc354068662)

[4.1 Popcorn.js 10](#_Toc354068663)

[4.1.1 Technologies Used 10](#_Toc354068664)

[4.1.2 Component Overview 10](#_Toc354068665)

[4.1.3 Sprint Overview 10](#_Toc354068666)

[4.1.4 Architecture Diagram 11](#_Toc354068667)

[4.1.5 Design Details 11](#_Toc354068668)

[4.2 Database Management 11](#_Toc354068669)

[4.2.1 Technologies Used 11](#_Toc354068670)

[4.2.2 Component Overview 11](#_Toc354068671)

[4.2.3 Phase Overview 12](#_Toc354068672)

[4.2.4 Database Schema 12](#_Toc354068673)

[4.2.5 Design Details 12](#_Toc354068674)

[4.3 User Authentication 12](#_Toc354068675)

[4.3.1 Technologies Used 12](#_Toc354068676)

[4.3.2 Component Overview 12](#_Toc354068677)

[4.3.3 Phase Overview 13](#_Toc354068678)

[4.3.4 Architecture Diagram 13](#_Toc354068679)

[4.3.5 Data Flow Diagram 13](#_Toc354068680)

[4.3.6 Design Details 13](#_Toc354068681)

[5.0 System and Unit Testing 14](#_Toc354068682)

[5.1.1 Popcorn 14](#_Toc354068683)

[5.1.2 Videos 14](#_Toc354068684)

[5.1.3 Database 14](#_Toc354068685)

[5.1.4 Host Server 14](#_Toc354068686)

[5.1.5 JavaScript 14](#_Toc354068687)

[5.1.6 Range of Browsers 14](#_Toc354068688)

[5.2 Overview 14](#_Toc354068689)

[5.3 Dependencies 14](#_Toc354068690)

[5.4 Test Setup and Execution 14](#_Toc354068691)

[6.0 Development Environment 14](#_Toc354068692)

[6.1 Development Tools 15](#_Toc354068693)

[6.2 Source Control 15](#_Toc354068694)

[6.3 Dependencies 15](#_Toc354068695)

[6.4 Build Environment 15](#_Toc354068696)

[6.5 Development Machine Setup 15](#_Toc354068697)

[7.0 Release | Setup | Deployment 15](#_Toc354068698)

[7.1 User Properties 15](#_Toc354068699)

[Appendix I: List of Figures 15](#_Toc354068700)

[Appendix II: Supporting Information and Details 17](#_Toc354068701)

# Overview

The goal of the document is to show the way that we have implemented the video tag portal. There will be details on the purpose, goals, as well as requirements and implementation.

## Scope

There will be a detailed discussion on how the project is implemented and what tools are necessary to create the project.

We have 4 major system components

* Database – this is where tags, users, and other locally needed objects are needed
* APIs – These APIs are need for a number of things such as login and video uploading
* YouTube and Vimeo – source of videos for the project
* Popcorn.js – used to implement the tags that our project is based around

## Purpose

We have created a video tagging system that allows users to tag points in videos with a various number of predefined tags. After tagging a video, the user then has the ability to share the video with his or her friends. The friends then will have the ability to also tag the shared video. The goal of the system is to make videos more socially connectable between people.

### Major System Component #1: Database

The Database holds all of the information that the system will need in order to function. The database contains all of the information to control the users, friends and the tags used by the player. The database also contains the tag timings and the list of the users tags, as well as the information on who tagged what videos. In the database there are separate tables for separate videos venders(e.g. YouTube and Vimeo.

### Major System Component #2: APIs

Here is a description of the APIs that we used for the project

Google data API

* + Allows us to use the videos that are hosted on YouTube
  + Allows us to upload videos to YouTube
  + Allows us to manage users that are logged into our site

Facebook API

* + Allows us to manage users that are logged into our site
  + Allows us to map friends to different accounts

### Major System Component #3: YouTube and Vimeo

We have support for the two major video hubs: YouTube and Vimeo. We are able to get the timings and embed both video sources which helps us get the tags to show up at the right points in the videos. Most of the functionality favors YouTube because it has a more complete API, so we use YouTube in searches and uploads. However, the actual tagging there is almost no difference to the user except for where the videos are from.

### Major System Component #4: Popcorn.js

The popcorn framework is what allows us to make the different tags appear and disappear based on the timing within the videos. There was support for the videos from YouTube and Vimeo. This framework has the ability to check an embedded video in a sub-second manor and be able to trigger events based on the timing.

## Systems Goals

System goals include giving the user the ability to use our application to make video part of the online social experience. We want the user to have a “Natural” user interface to make the tagging process simple for the user. Also one major goal is to make the product as much as possible a social networking site.

## System Overview and Diagram

Much of the project will depend on the various APIs that are needed to make the videos display and show up well on the web page. We will also need to use the APIs to manage the users that access the site using Facebook and Google. In addition to the APIs needed in the project there will be a database that is used to manage the different users and the tasks that they will be able to do. There are many different things that will be going on per user such as knowing there favorites to logging in to the site. The most significant system component is the popcorn.js framework that will actually allow us to make time based tags on the fly which is the main part of the project.



## Technologies Overview

PHP

* Stands for PHP: Hypertext Preprocessor(recursive acronym)
* It is a server side scripting language that is designed for web development.
* Supports cross platform development

HTML5:

* A markup language that is for structuring and presenting content for websites
* Contains many more features than the previous versions of HTML
* Supports cross platform development

JavaScript:

* An interpreted client-side scripting language
* Used to make user interfaces more interactive
* Used to communicate asynchronously
* Used to alter document content that the user sees dynamically

Popcorn.js:

* A JavaScript frame work for editing media in real time
* Able to edit media based on timing of the media
* We use it to control the main functionality of our site by displaying tags are specific times
* Support for YouTube and Vimeo
* Checks media in fractions of a second

Google APIs:

* Google Maps API v3
  + Using the maps API to display the map tags in our application
* Google+ API
  + Added to allow the users to login with google accounts
* YouTube Data API v3
  + Allows the access of the YouTube search
  + Allows users to upload videos to YouTube
  + Allows checks users uploads to YouTube

Facebook API:

* Allows users to login using their Facebook accounts
* Allows users to see the friends that they have on Facebook
* Allows users to view profile pages from facebook

jQuery:

* A feature rich JavaScript library
* Allows for simpler dynamic HTML traversal/modification
* Allows for simpler dynamic animation and Ajax coding
* Multi-browser support

Viemo:

* A website for hosting videos.

YouTube:

* A website for hosting videos.
* Contains many useful APIs

MySQL:

* Open source relational database management system
* Allows for multi-user access to a number of databases

# Project Overview

This section will provide some of the team roles and the way the project is managed and kept on track.

## Team Members and Roles

The two members of the project are Anudeep Potlapally and Travis Rous. Each member is responsible for attending the scrum meetings and doing what is assigned to them after each meeting. There are only two team members so there is much overlap in the roles, however Anudeep was the official contact to the client. Travis had more of a testing role as well.

## Project Management Approach

We are using the Scrum development cycle to manage the project. This consists of almost daily scrum meetings that are less than 5 minutes each (minimum bi-weekly). There are sprints that range from 2 to 3 weeks. We are using GitHub and Trello to manage our code and progress over time. Trello is like a digital board with task card that are able to be moved from to show the progress we have done in the current sprint and assign some cards to have a higher priority than others to show the significance of each task or what is left for each task at hand.

## Sprint Overview

The major phases in the project are as follows (We will explain in greater detail in the next few sections):

Sprint I:  
 Embed videos from 3rd party sources.  
Sprint II:  
 Implement feature to play videos from YouTube using Popcorn.js.

Link video player to HTML elements to display hard-coded tag information.

Use Database to save and retrieve tags.

Sprint III:  
 Implement feature to play videos from other possible sources.

Display multiple tag types simultaneously.

Sprint IV:

Started work on an upload page

YouTube search was implemented.

Sprint V:

Updated favorites page to include the users profile picture from Facebook.

Added a friends page.

CSS experimentation.

Vimeo integration.

Search and Loading of videos now in the same bar

Linked Facebook and Google accounts.

Sprint VI:

Finished Requirements and started testing

### Sprint 1: Initial prototype

The initial prototype was needed to get our idea out to the client and make sure that we were on the same page. We were also able to get the videos embedded into the site. There was also research for finding the many different APIs and other frameworks that would be useful to us in the future of the project.

Overview of sprint:

* Created prototype
* Finished the client agreement
* Prepared for client presentation
* Cut up the project (know when to do what)
* We defined some project boundaries
* Also defined a set of tools that are were going to be used

Some of the early goals

* Play videos in the application
* Play videos from major sites
* Create a database to store some video information
* Social network ready
* Selecting video tag points
* Create the list of tag types
* Have a simple way to make the tags editable
* Filters
* Possible mobile development

### Sprint 2: Database Design

**Summary:**

For this sprint there was mostly research and learning of how to implement some of the features that we will be trying to use in the project. Anudeep had discovered and Implemented the Popcorn.js which is designed create videos with an features such as what we are trying to implement.

**Sprint 2 Backlog:**

* Database Design (In Progress...)
* Creating a favorites page (In Progress...)

**Sprint 2 Burndown:**

RESEARCH

* Anudeep had researched the Youtube data API and found that we needed to get permission from Google to use it which allows VTP to search, upload videos. Travis sent google a request to use the data api and we are currently waiting for a response.

DATABASE DESIGN

* The current host location is just local on our personal machines. There is a database that we can both access that is being hosted on Travis’ desktop. We are using MySQL via the phpMyAdmin tool.
* We currently have a database with three tables; users,yttags,favorites. In the users table there an auto incrementing ID number, the user id that is a 50 char array for the userId, a dateRegistered date that is auto timestamped, and a hostSite variable that is going to tell us what site the user is logging in with(e.g. facebook, google account, ...etc). In the ytTags table we have an auto incrementing ID number, a videoId which is an the ID that the site that we get the video from uses to ID the video, a userId is used to associate the tag that is added with a specific user, and other video details used to create the tags. In the favorites table there is another auto incrementing ID, along with a userId to tell us which user’s favorites this is, an auto timestamp, and a videoId to tell us the video ID that is a favorite.
* Anudeep created a database connector class to interact with MySQL from PHP and as we go we will be adding functions for frequent DB queries.

INTEGRATION OF POPCORN.JS

* Popcorn.js is an open source library developed by Mozilla Corporation which allows audio, videos to interact with HTML elements. Anudeep researched and integrated popcorn.js into the system and also automated displaying image tag’s for videos with tag entries in database. Popcorn.js supports HTML5, Youtube, Viemo videos in future adding these supporting

CREATING A FAVORITES PAGE

* Travis started the favorites page, it currently gets the favorites for a dummy user named “123” who has five videos saved as favorites. Currently the page will only display the thumbnails and titles of the favorites of the videos.

LOGIN VIA GOOGLE AND FACEBOOK

* Anudeep Integrated facebook PHP SDK and Google PHP API to allow users to login via Facebook or Google. Google API does not provide access to user’s friends list. We might need to disable filter tags by friends option to google users or create additional feature to support managing friends.

ERROR LOG

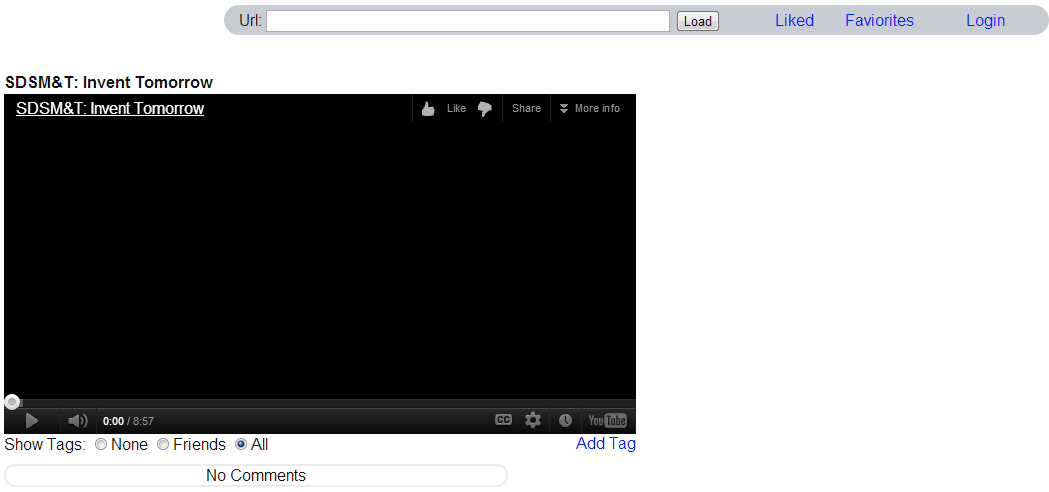
* Anudeep created error log function to keep track of all PHP generated errors. This function will create detailed entry for each error so that developers can attend the server-side bugs.

**Current Prototype**

The current prototype has gained a favorites page and is now using a database to get and store the information about users. We also can now login using facebook and google accounts. Anudeep has begun use of the popcorn.js.

**Prototype images**

Sprint 2 look:



### Sprint 3: Design and Connect pages

**Summary:**

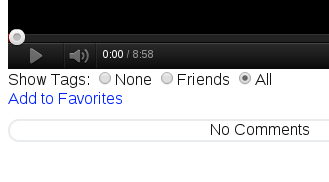
In this sprint we created features for logged in users to add videos into their favorites list and also they can go through the list of their favorite videos and just click on them to watch again. We created a form to add image tag to the video currently being watched and play the image tag automatically from the database. We hosted  the code on a server to see the product directly on the internet.

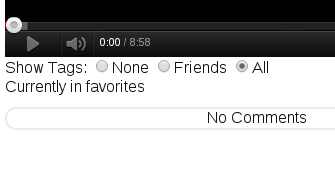
**Sprint 3 Remaining Backlog:**

* Youtube Data API (Permission pending...)

**Sprint 3 Burndown:**

ADD TO FAVORITES

* For logged in users Travis created option to add videos to their favorites list, and wrote required functions to add video information linking to user in the database. 
* Videos can be added to favorites only once so when the video is already in favorites list the text would change to “Currently in favorites” forcing users to not make duplicates videos in their list.



FAVORITES PAGE DESIGN:

* Travis created a page where logged in users can look at the list of their favorite videos. Favorites  page displays a thumbnail for the video and the title for the video and users can click on either video or the title and video and it would take user to the homepage to start watching the video again. To save memory on server  thumbnail and video titles are not saved on database but retrieved from Youtube server every time the page is loaded. 

ADD IMAGE TAG FEATURE

* Anudeep created a form for users to add Image tags and link it with a video and save information in the database. Wrote code to pull the saved tag information from database and show on the page at appropriate time. Tags should be played between appropriate timings of the video so linked the form to the player so that users can use seek bar on video to set exact time the tag should be played. Wrote JavaScript function to test the valid image link provided by the user and class to handle Image uploads to the server.



HOSTED ON SERVER:

* Anudeep found a website to host the code which can run PHP and use MySQL database. Made required changes in Google API and Database classes to make them work on the new server.

### Sprint 4: Giving the users an attractive interface

**Summary:**

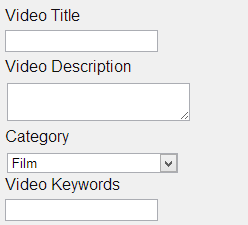
Travis started the upload page, however it took more time that what was anticipated and still needs work before moving on. On the upload page, originally the user needed to put their credentials into the YouTube upload page on the VTP site.  This is not done well so Travis is in the process of getting the user authentication to work with the upload. Anudeep had implemented the YouTube search with user authentication and also implemented other tags.

**Sprint 4 Remaining Backlog:**

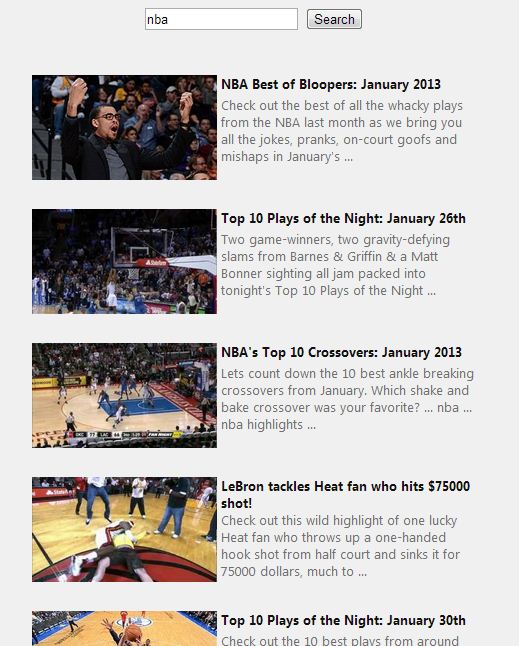
* Video Upload fix-it list
  + error checking
  + Graphics
  + Add the users Video to the database
* Friends Page
* Grabbing new users videos from YouTube
* Displaying Map and web link tags

**Sprint 4 Burndown:**

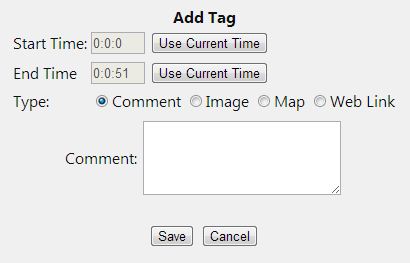
YOUTUBE UPLOAD

* Added Upload to the headed
* Once clicking you are brought to a page to input the Video details.
* Once the video fields are filled out then you will be able to authorize the upload to YouTube.  Then you can choose the file and select upload.  This will then take you to a page where the you get the details about how to find the video on YouTube and a disclaimer about the video taking a long time to get done processing. Also on this page there is a link put the user back to the player page with their newly uploaded video.

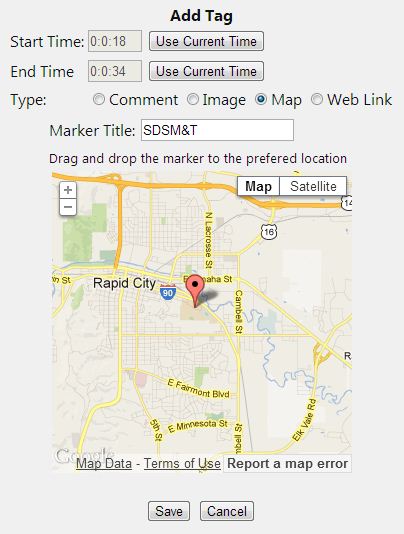
SEARCH YOUTUBE VIDEOS

* YouTube made the Data api V3 publicly available for all the developers Anudeep implemented the search functionality for the videos. Now users can search YouTube videos and they can also change the search filters for relevant content.  
  + - 

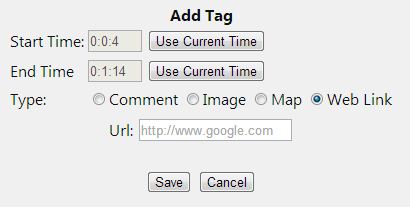
ADDING TAGS TO VIDEOS

* All tags are saved in database and for all of the following tags error handling has been implemented.
* Comments: Users can add comment and maximum text allowed has been set to 512 characters.When the video is being played these comments will be scrolled under the video.

MAPS

* Anudeep implemented map tags using Google Maps Api for users to drag and drop the marker to their preferred location. users can also set the title for markers.

WEB LINKS

* Users can add URL links to their websites.

### Sprint 5:

# Requirements

For the detailed list of requirements check the Software Requirements Document. Here is the undetailed list of requirements (User stories that were initially provided).

* Play the videos within a web application
* Make the application have the ability to play local videos
* Possible integration with sling.com -- *unimplemented*
* The first time viewing of the video should store all the information of the video that is being played – *unneeded*
* Design some kind of login system that allows us to know who one’s friends are
* We need to be able to pause a video at any one moment and place one of many kinds of tags
* We have a set of predefined Tags to choose from
* Need to have the ability to change the list of tags
* We need to set up a filter system to filter out tags placed on a video

# Design and Implementation

This section is used to describe the design details for each of the major components in the system. This section is not brief and requires the necessary detail that can be used by the reader to truly understand the architecture and implementation details without having to dig into the code.

## Popcorn.js

### Technologies Used

We are using open source library “Popcorn.js” developed by Mozilla Foundation. Popcorn.js allows VTP to interact with the limited 3rd party video sources like YouTube, Vimeo and any HTML5 videos either saved locally or from other 3rd party sources.

### Component Overview

Features:

Play local videos in HTML5 and interact with them

Play both HTML5 and flash videos from YouTube and Vimeo.

Triggers JavaScript functions when video is on a particular time.

### Sprint Overview

Sprint I:  
 Embed videos from 3rd party sources.  
Sprint II:  
 Implement feature to play videos from YouTube using Popcorn.js.

Link video player to HTML elements to display hard-coded tag information.

Use Database to save and retrieve tags.

Sprint III:  
 Implement feature to play videos from other possible sources.

Display multiple tag types simultaneously.

### Architecture Diagram



### Design Details

This is where the details are presented and may contain subsections.

## Database Management

### Technologies Used

Created DbConnector class in PHP to interact with MySQL Database. All queries are written as individual functions in DbConnector class.

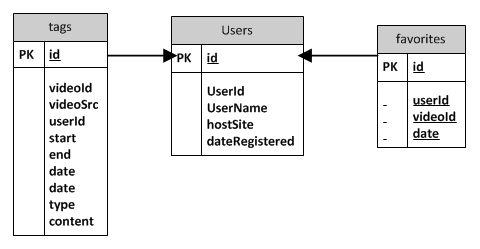
### Component Overview

Features:  
 Connect to database and authenticate  
 Execute frequent queries from functions  
 Disconnect when PHP execution is completed

### Phase Overview

Phase I:  
 Start database design  
Phase II:  
 Implement Database design.  
 Write DbConnector class   
Phase III:  
 Add queries as functions in DbConnector class

### Database Schema



### Design Details

This is where the details are presented and may contain subsections.

## User Authentication

### Technologies Used

Implement user authentications using Google API PHP Client and Facebook SDK.

### Component Overview

Features:  
 Facebook SDK:  
 Authenticate users; access basic information  
 Get friends list to compare with the VTP database for filtering video tags  
 Google API PHP Client:  
 Authenticate user accounts, access basic information  
 Upload videos into their YouTube account. (Requires YouTube Data API)

### Phase Overview

Phase I:  
 Research on both the API’s  
Phase II:  
 Implement both API and save basic information into Database.

### Architecture Diagram



### Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

### Design Details

This is where the details are presented and may contain subsections.

# System and Unit Testing

### Popcorn

We will need to make sure the popcorn is behaving as we expect, by checking the timings and making sure that they are consistent with what is happening on-screen.

### Videos

We need to make sure that the videos are being displayed correctly and are sized properly. The videos need to be able to be played while to tags are being updated.

### Database

We need to make sure the database is secure as well as able to be responsive for the users accessing it.

### Host Server

Security is the major concern that is associated with the Host Server.

### JavaScript

We will be using the JSLInt to test the quality of the JavaScript functions.

### Range of Browsers

We will need to test the functionality of the different browsers on our site.

## Overview

At this time we only have one framework for testing and that is the JSLint framework described below.

## Dependencies

JSLint for testing the individual JavaScript functions to make sure that they are living up to some standards.

http://www.jslint.com/ --- the webpage for the tester.

## Test Setup and Execution

The setup of the JSLint is go to the web sited and paste in the code that you want to test. (There are also command line versions for download)

# Development Environment

The basic purpose for this section is to give a developer all of the necessary information to setup their development environment to run, test, and/or develop.

## Development Tools

Development tools used in this project are Notepad++, PHPMyAdmin.

Notepad++ is used just to do the markups on the code to see where the keyword are used and manage brackets.

PHPMyAdmin was used in the database management.

## Source Control

For the source control we are using the GitHub repository that uses a windows interface to access the repository. GitHub is completely open to the public to see unless you pay for a private service.

## Dependencies

Describe all dependencies associated with developing the system.

## Build Environment

The project will need to be deployed on a PHP server and a MySQL server will need to have the tables setup using a .sql script that will be sent with the project. There will need to be a connection between the server and the PHP using a database connection in PHP.

## Development Machine Setup

In order to make a machine able to develop you will need to download Notepad++ and have access to a PHP server and MySQL server.

# Release | Setup | Deployment

Upon release the product will be hosted on a server that will have the bandwidth high enough to support thousands of users at a time. The project will need to be deployed on a PHP server and a MySQL server will need to have the tables setup using a .sql script that will be sent with the project. There will need to be a connection between the server and the PHP using a database connection in PHP.

## User Properties

There will be one final website that will be used for the hosting that the users will connect to.

* + - 1. List of Figures

**No table of figures entries found.**

* + - 1. Supporting Information and Details

This document will contain several appendices used as a way to separate out major component details, logic details, or tables of information. Use of this structure will help keep the document clean, readable, and organized.