

Teri-Ann Barber [tbarber2016@fau.edu](mailto:tbarber2016@fau.edu)

Enmanuel Ortiz eortiz2018@fau.edu

Richard Santiago rsantiago2018@fau.edu

Sadiki Brown sbrown2017@fau.edu

**November 04, 2019**

**Milestone 3 More Detailed Description**

JavaCups’

Project group #1

Fall 2019

CEN 4010 Principles of Software Engineering

CritiqueCafé

|  |
| --- |
| **Revision Dates** |
| **1st Submission Date: 09/23/2019** |
| **2nd Submission Date: 11/04/2019** |
|  |
|  |
|  |
|  |
|  |

**M1 Project Proposal:** **CritiqueCafé**

In the entertainment world there are plenty of ways to get your reviews to be small and concise for the end-users who need a quick summary/opinion on what they’re buying. There are sites like Rotten tomatoes which gives you a rating not only based on critics but also a rating based on the opinions of the consumer, quickly giving the end-user gratification on whether they enjoyed or hated the product or ignored it because of the reviews. Another site like IGN which gives you a rating on Videogames, doesn’t allow the consumer to report opinions, but gives you a quick and concise summary on the product. Our website wants to combine multiple media review sites into one, one that allows critics and consumers to join in conversation quickly and easily, either through username or anonymity.

We want our website to not only go over one type of entertainment, but multiple genres of entertainment decided by the end-users who visit the website. We give reign to the end-users so they can create their subjects and post their personal reviews of the product, mostly like an imageboard or forum, where people can freely speak their mind on whether it was exceptional, mediocre, or unpleasant (of course rules will have to be implemented and if enough end-users visit a moderator/administrator would need to watch recent posts). We would want the opinions of music-enthusiasts, gamers, all movie-goers and book addicts to be shared on one specific website for quick access.

**Competitive Analysis**

| CritiqueCafe | PluggedIn |
| --- | --- |
| * User Login | * No User Login |
| * Quick access to subjects through navbar * Ability to reply | * Quick access to subjects through navbar * No reply ability |
| * Ability to create initial post * Search Function | * No ability to create initial post * Search Function |

**Data Definition**

**PHP** – Hypertext Preprocessor, Web scripting language (going to be mainly used in this project for databasing our information.)

**SQL Server** – Structured Query Language Server, going to be where our data is held and easily accessed by database managers. Allows easy changes to information/entities using SQL.

**End-user** – Main consumer; Anyone that uses our product.

**IP –** Internet Protocol, usually given an address which is mainly used for communication, can be used to identify a specific network.

**Database Tables** – A table with a set of data elements/values that are related. (For example: Posts in our database are separated by vertical columns that keep information such as poster\_id and post\_text. These can hold either integers or strings, post\_text can hold a text value which can either be held as text or hold a formatted embedded video. (425x344))

**Overview**

This website is mainly going to be used for a streamlined approach of reading reviews. Giving the end-users quick availability to the specific review they want to read about, either by users of the website or critics from the website, also allowing them to reply to reviews to show a differing opinion or agreeing with the initial review.

* Imagine the user going out to the movie theaters to watch a movie that was inspired by a book and he has read reviews of the movie being not great. He needs a quick source on whether the movie would be worth watching over the book. He then goes to CritiqueCafé and either goes through the subjects or searches for the specific movie threads. He then reads the movie is nowhere near as good as the book, he then has the ability to (on the same website) go to the book section and search for the book for reviews and whether or not to buy it later to read, rather than watching the movie. Saved time on searching for other websites for book reviews and not wasting money on the movie.

**List of high-level functional requirements**

1. Ability to Login/User access (1 – must have)
   1. Saving the user information through a PHP file to an SQL server for quick access to their account. Allows them to be recognized by other posters/commenters.
   2. Don’t need to immediately have an account to post a reply, allows the user to post Anonymously with a guest name.
   3. Moderator capabilities, allowing bans and deletion of posts not following guidelines.
2. Database Functionality for forums posts (initial posts and replies) (1 – must have)
   1. Initial posts should be available to access by Database managers which can choose to accept or delete the post. End-user will also be able to delete post that they posted.
   2. Ability to post images/videos to accompany replies/initial posts.
3. Search Capabilities (2 – desired)
   1. Quick search through subject, should search through all subjects; Allows end-user to save time by finding his/her item immediately.
   2. Able to set boundaries for search, such as using date created or searching by username.

**List of non-functional requirements**

1. Available at all times of the day
   1. Downtime should be very minimal; website should be active for the whole day unless server is shut down for an unknown reason.
2. Ease of us
   1. Quick access to all subjects given to the end-user; Easy and accessible for any skill level.
3. Search should be immediate, as well as comments/replies
   1. Comments and replies should not take over a few seconds to process, no delay to posting an initial comment or replying to a comment.

**High-level system architecture**

Languages

* CSS
* HTML
* PHP

Supported Browsers

* Google Chrome
* Mozilla Firefox
* Opera Browser

Framework

* Scrum

**Database organization**

DB Tables:

* banned
  + ID
  + Banip
  + Banreason
* Forums
  + Music
  + Books
  + Movies
  + Games
* Posts
* Topics
  + Live on Ice
* Users
  + Manny

Media Storage:

* Image links will be stored in the DB under posts (mainly storing the link to the image), same with any hyperlinks or videos.

Search architecture:

* Look for any characters that need to be cut
* Setup search for each array, such as looking at posterName and then $\_GET[‘posterName’] (Can limit search with Topic\_ids or dates (Listed in the PHP as where, searchtype, forum, etc.))
* Make sure variables cannot be empty arrays
* Output any relevant DB information pertaining to the search boundaries

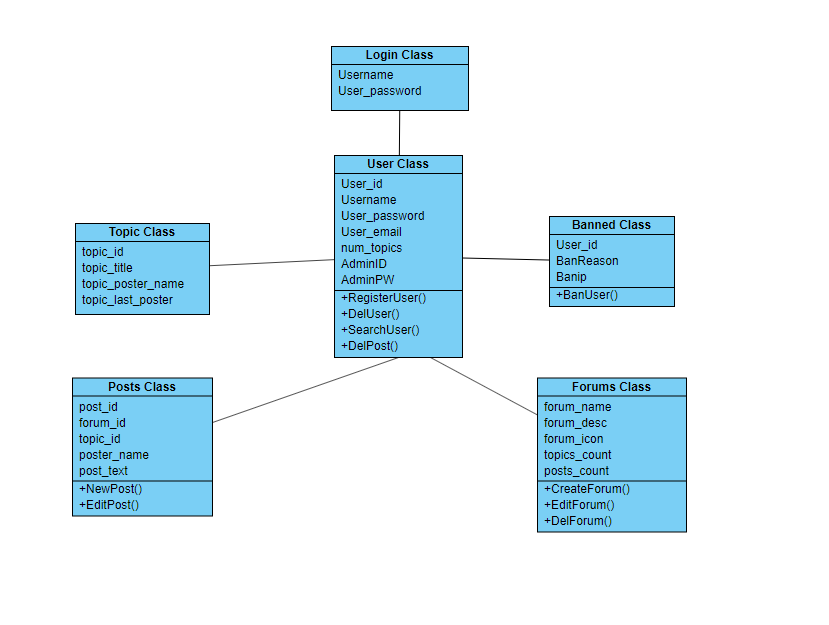
APIs:

* Applying any playable video APIs, allowing content to be hyperlinked into a forum and be playable. (Most likely a YouTube API)

Non-trivial algorithms:

* Admin Capabilities: Setting up an admin login .html page that can only be logged into with data kept within a .php file. Example of deleting a forum, search by forum\_id and using a function db\_delete() to delete the accompanying forum from the database (works the same for deleting other database data such as using user\_id). Using other .php files that contain functions such as banip which saves the IP of the banned user to the database, so they can have no access to the website.

**High-Level UML diagrams**

****

**Actual key risks**

* Time/schedule restraints
  + Due to some of the team having a job or having full time classes or living far away there are sometimes when we can’t fully communicate with the team when needed through a group meeting. To alleviate this, we created a group on WhatsApp to discuss any problems that came up during development or if any help is needed. We can now discuss either through text or through group phone calls.

**Team JavaCup**

* Enmanuel Ortiz – Back-end dev (Github master) eortiz2018@fau.edu
* Richard Santiago – Front-end dev/back-end back-up? (Product Owner) rsantiago2018@fau.edu
* Sadiki Brown – Front-end dev (Development team) sbrown2017@fau.edu
* Teri-Ann Barber – Back-end dev (Development Team)[tbarber2016@fau.edu](mailto:tbarber2016@fau.edu)

**Checklist**

1. Team decided on means of communications: ✓
2. Team found a time slot to meet outside of class: ✓
3. Front and back end team leads chosen: ✓
4. Github master chosen: ✓
5. Team ready and able to use the chosen back and front-end frameworks: ✓
6. Skills of each team member defined and known to all: ✓
7. Team lead ensured that all team members read the final M1 and agree: ✓