

JavaCups’

Project group #1

Fall 2019

CEN 4010 Principles of Software Engineering

CritiqueCafé

**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**

**“** This section serves as the “dictionary” of your document. It defines main terms, data structures and “items” or “entities” at high or logical (not implementation) level (e.g. name, meaning, usage, and NOT how the data is stored in memory) so it is easier to refer to them in the document. Focus on key terms (main data elements, actors, types of users etc.) specific for your application and not on general well know terms.”

**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
2. Database Functionality for forums posts (initial posts and replies)
3. Search Capabilities

**List of non-functional requirements**

1. Available at all times of the day
2. Ease of us
3. Search should be immediate, as well as comments/replies

**High-level system architecture**

“List of main software products, tools, languages and systems to be used, list of core APIs available at this point, supported browsers, etc. Decide on a framework”

Languages

* CSS
* HTML
* PHP

Supported Browsers

* Google Chrome

Framework

**Team JavaCup**

* Enmanuel Ortiz – Back-end dev (Github master)

**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: