# Text Description automatically generated

# Software Development Project: The Process of Designing, Programming, and Implementing ‘If You Like…’, A Multi-Media Matching Suggestion Tool

**Writing, Research, and Development Conducted By:**Elliott James Tomkings  
Student ID: 21067898  
Module Leader: Dr. Steve Battle  
Supervisor: Sami Abuezhayeh

*Submitted to The University of The West of England in partial fulfilment of the requirements for the degree of:*

Software Engineering for Business

May 2025

*Department of Environment and Technology - Computer Science and Creative Technologies*

## Abstract

## Foreword

Due to some life events, saliently moving out, parenting my young infant son, the cost of my initial project idea (and subsequent changing of projects), and some familial illnesses that have called me away from this piece of work, I have had considerably less available time to give to this project, than I would have liked; these factors caused the project shift, which in itself gave me less time for this final iteration.  
  
This is all to say, life significantly got in the way of this project, and the scale and implementation and methods are all different and smaller because of this.

## Introduction:

In many words, the intent of this section is to illuminate the details of this project’s background and context, the justificatory programming reasons for venturing into this project specifically, and the anecdotal or personal observations made that led to this project being the one chosen for this *Software Development Project*; in fewer, this is the problem statement.

The core aim of this project is to design and create an online Multi-Media Matching tool that suggests media from multiple mediums (be that music, games, books, TV shows, movies) based on user-inputted preferences on *those same* mediums, i.e., a user may input that they love ‘The Dark Tower’ book series by ‘Stephen King’ and be looking for video-games that convey a similar feeling or atmosphere; in this scenario, the website may output a game like “Fallout: New Vegas” by ‘Obsidian’, based on genre or theme similarities. This application is meant to be iterative and beginning with just books and movies, as a proof of concept around creating cross-recommendations, and then in later iterations (shown early through an experimental version of the page) start to include game and music recommendations.

The principal considerations that arose when making the final decision to endeavour with this project were as such: mostly, this is a purely functional project and artefact, a tool that works as way to broaden my, and any users horizons by complementing an already established palette, perhaps even bridging mediums that a user typically doesn’t indulge in by providing enticing recommendations.

Secondly, from a technical perspective, during the pre-planning of this project, I decided it would be beneficial to apply experience I already had, (like using Python, PHP, and SQL) and new topics that complement those already established areas of knowledge to deepen my awareness of their functionalities, like learning to integrate HTML, JavaScript, CSS, API Calls, and Flask to create both the interface and backend of this tool.

The anecdotal reason associated with this project stems most accurately to myself, and the prior listed IP’s (intellectual property’s); as an adult, very little has given me alluring reason enough to pick up a book and read it front to back, that was until I had the knowledge of “The Dark Tower” as a suggestion based on “Fallout: New Vegas”, and the free time I had around my son’s birth. This suggestion and moment fostered a love for reading that was until then, unknown to me. The hope with this project is to be able to create and provide recommendations strong enough to create a change in someone’s perspective, as I had in mine.

## Method:

### Development Approach - Tools:

The first and biggest considerations of this project, aside from the project topic itself, were the tools that I would use during the development process, it was paramount that I focused foremost on the scale of this app and what tools would cater to what is realistically, and currently, a small-scale application that is expected to function with less-than 10 concurrent users and minimal traffic. There is an argument to be made concerning the scalability of this project and developing for ‘Facebook-level’ traffic in the future but given the already accepted complexity of working with a recommendation system (https://www.mdpi.com/2076-3417/10/21/7748), the foci of this project ended up being pure basal functionality and the capacity for my own personal use to be uninhibited.

With these considerations, and the context discussed during the foreword, dissolving down my options, I decided upon ‘Flask’ as my web framework as it coincided with my needs of being relatively simple and the want of using something somewhat familiar to myself; Flask working with Python primarily was a bonus in this context too. Its lightweight design and architecture is bolstered by its most simple features being made simpler through extremely straightforward syntax, this is most demonstrated through its route-to-function mapping being as direct as possible “(@app.route('/search'))” followed by function code made for the least obfuscation possible which became necessary when large code-blocks ended up becoming tangled by the very nature of recommendation and filtering. Tying in Flask’s built in development server and debug mode, with a Github repository made for a constantly iterative and rapid to redeploy methodology and design process.

There were considerations for other frameworks but the issues, or more generally the shirking of extra functionality gained through these due to non-necessity made them less viable than Flask; Django was one avenue to consider and may be the next step for this application once scalability and size becomes of greater importance, but the scale of this project does not currently align with it.

The most core functionality of this project is designed around working with API pulls and manipulating the results to work alongside the recommendation system, while this project has changed in scope a few times regarding what mediums would be used in comparisons, the vast array of methods to obtain oceans of knowledge, be that books, movies, games, music etc. is heavily supported by the current options. OpenLibrary API bolsters over 28 million books [potential for reference of proof] that are all stored on typical webpages as well as in.json and YAML formats that are accessible through URL modifications, this conversion being a pre-completed task as well as the filtering options easily leveraged like language filtering, made this the obvious choice for deriving book information. It’s worth noting that the other API’s are not as comprehensive or “pre-worked”, but still contain essentially all the information needed, these were the Spotify API, TMDB (The Movie Database), Google Books API, and IGDB (Video Game Database API).

Other tools that were used but do not warrant extended discussion are addressed succinctly as: RESTful API, Python’s ‘request’ library, Github, jsonify, and using .gitignore and a .env file for securely storing API Key’s and Credentials.

Development Approach – Risk, Ethics:

The risks presented and the risks expected during preliminary assessments of this project were not *always* parallel which led to sudden developmental changes having to be implemented:

* Risk - Technical: The availability and quality of third-party API, alongside the impact of rate limiting could prove to dispel entire chunks of this projects expected outcome; if the information and data simply does not exist in a way that is acquirable, there is little on my end that can be done to counteract that.  
  + Mitigations and Considerations: While the architecture of the multiple API used in this project are not synonymous, using filtering and.json manipulation allowed for an, at least algorithmically consistent, way to compare the data derived from each service. Additionally, the application of “fall-backing” and creating hybrid API implementations allows for more consistent data information, where certain parts of one API call are merged with another to create a more full picture e.g. mixing OpenLibrary API with Google Books API.  
      
    There are measures and filters to be implemented that can check against non-standard characters, but so far implementing a method that fully negates non-English results, or low-quality results in a bulletproof manner has seemed unfeasible; the consideration of whether all non-English media should be removed from recommendations also comes into question, when it is not only western media that should be viewed and experienced.  
      
    For rate limiting, my initial assumption is that the service will never be used in such a manner that this would come into effect, but for coding quality purposes, caching and per-session information storing should allow for rates to not exceed what is expected in each API’s documentation. This is also an effective strategy for quicker search results within per-session user experiences, the efficiency of this caching and its effect on speed will be demonstrated in the presentation part of this assignment.
* Risk – Project, Fundamental: Scope creep, or even just the concept of scope itself in this project has been a prominent factor that has elucidated issues and necessary changes from day one of development, questions are asked e.g., “how many mediums do you compare?”, “how do you represent recommendations?”, all which stall development more than create burgeoning ideas. This, alongside my time constraints mentioned in the ‘Foreword’, and a general knowledge gap on certain fundamental tools and technologies that *must* be implemented for this app to work at all, have made for certain developmental pitfalls along the way.  
  + Mitigations and Considerations: At the earliest point I decided use a simpler framework and focus on useability over UI as is discussed in the tools section of this report, in the endeavour to fit within an already short timeframe. Additionally, I decided to use tools I was already at least partially familiar with, focusing on a web-development project, using Python, and decided *not* to store data in a database.  
      
    It was crucial to, early on, make decisions against incorporating ‘TV’ into the recommendations algorithm, while the API pull remains almost the same as that for movies, the broadness of television regarding pure content was too varied and granular to compare against more unique and singular pieces of content like individual video games.  
      
    As a point parallel to this, I decided to keep the core functionality of this application working with two mediums, and keeping other medium implementation as a part of this project that is on the outside of the assignment (and as parts of an ‘experimental feature’ section of the site).

Given the concept of this project, the ethical considerations are quite minimal as demonstrated in the ethical checklist, there is very little that needs to be handled in relation to user data since there is no username or user information stored, each session is per-session and cache based and only asks for the title of a piece of media.

The API keys are stored in a way that is inaccessible to any computer that isn’t the one that the application is locally ran on, through the use of .gitignore and .env files within the codebase, and given the brief or limited use of the API’s (regarding commercial use and rate limits), there are no licensing or attribution acknowledgements required in the documentation or code. This is to say, obfuscating measures have been implemented to protect against a system that is, even if exposed, not holding any personal data.

## Research: Primary:

As a part of the primary research for this application, I conducted informal usability testing with 4 anonymised participants at two different dates, these tests were conducted independently at the end of March, when the application was in a more infantile state, and at the end of April, when the application was in a near-completion state.

The prompting applied was as such: an explanation, one much like the ‘Introduction’ section, was delivered with the addition of, “*Provide notes on the searches you conducted during your testing, notes on your overall experience at the end of your testing, and provide a rating from “Poor-to-Excellent” based on these factors: “Usability and Functionality”, “Accuracy of Queries and Recommendations”, and “Design and UI Comfortability”.’*

To clearly summarise the findings of the two tests, each date of testing will be broken down into generalised points raised if combining both the “factors” of testing rated, and the notes provided by each participant.

* **March 30th:**
* *Usability and Functionality:*  
  + Participants generally noted that, (because of the extremely primitive design of the site and options provided) it was clear how to operate the website; the intuitiveness went somewhat hand in hand with what even the final presentation of the site would be, given both earliest and most recent interactable sections of the site are supposed to be very limited. However most noted that the ‘game’ and ‘music’ radio buttons had zero actual functionality attached to them, (which was true at the time).
  + Participants would note that there wasn’t a very clear way to intuit as to why certain search results were being provided or why recommendations were being provided. One participant mentioned, “it’s clear, since I know the two films, why one is being recommended based on the other, but if I didn’t know the films then there is nothing showing me how that’s been concluded…”
* *Accuracy of Queries and Recommendations:*
  + Participants noted that search and recommendation results generally ranged from ‘Poor’ to ‘Fine’ for reasons that trended similarly in their explanation.  
      
    The scale of this ranged from, the piece of media being searched for was pulled and displayed, but surrounded with many non-related or low-quality results, and not ordered in any particular way, e.g., having less relevant results or results that are extremely unknown (student movies or spam uploads on both TMDB and OpenLibrary) being shown first before exact title-matched results. This happening due to there being exactly *no* content filters on search and pull results at this point.  
      
    To, correct search results being displayed but recommendations being provided that were extremely shallow or entirely non-sensical, one participant noted, “I searched for a Stephen King book and got recommended a kids puzzle book called ‘101 Mazes’”.
* *Design and UI Comfortability:*
  + The criticisms levelled here were entirely synonymous, these being that the site design was “archaic” in some scenarios and “stilted” in others. This was due to factors like there being no loading indication, very plain layouts for all cards on the site, no indication for if an error had occurred or why, and mostly unformatted HTML elements being the core of the design.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Participant** | ***Usability and Functionality*** | ***Accuracy of Queries and Recommendations*** | ***Design and UI Comfortability*** | **Overall Experience** |
| 1 | *Great* | *Fine* | *Poor* | *Fine* |
| 2 | *Fine* | *Poor* | *Poor* | *Poor* |
| 3 | *Fine* | *Fine* | *Poor* | *Fine* |
| 4 | *Fine* | *Poor* | *Poor* | *Poor* |

Figure : March 30th feedback table

*Contextually, the site, at this time, looked as follows:*

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Original search layout, with redundant buttons and crude HTML and CSS Styling.

A screen shot of a computer

AI-generated content may be incorrect.

Figure : Original results card.

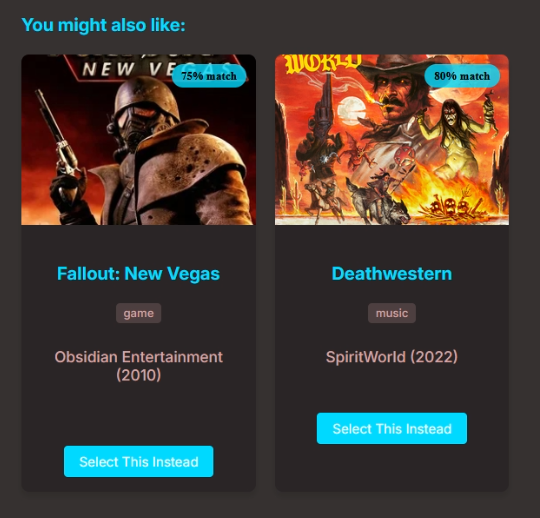


Figure : Original recommendations card, with hard-coded ‘Match Percentage’ based on no actual algorithm.

What the user feedback illustrated was immediately clear, regarding UI and especially the recommendation algorithm being both primitive in design and shallow in results, however this feedback was only useful if analysed, interpreted, and turned into actionable tasks and implementations to the site that could be demonstrated, at a secondary, later user testing cycle. For each “factor” that was rated, and the notes provided in the feedback, due to this, an immediate revision on the algorithm was put into work, focusing on removing generic thematic links like ‘sequel’, ‘based on a book’, making sure to focus instead on more substantial thematic links and genres.

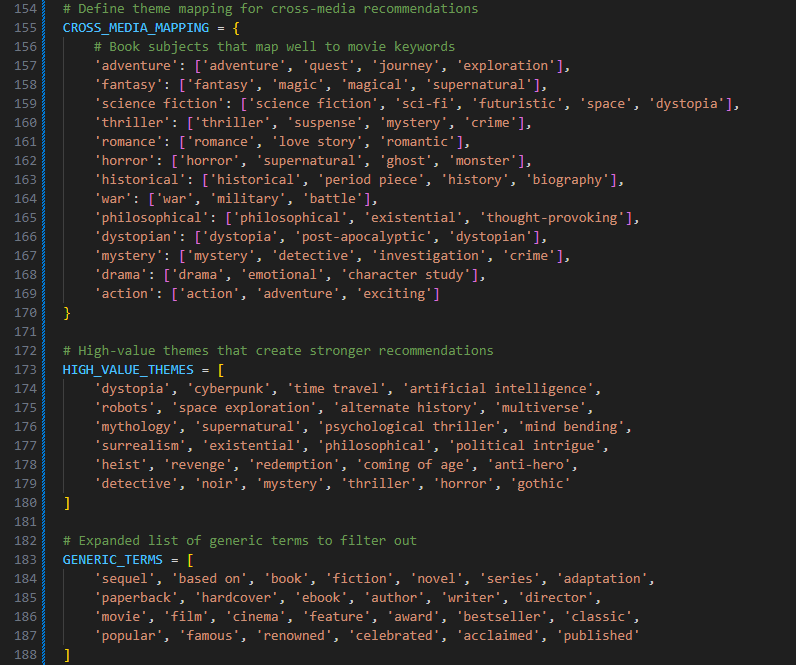


Figure : Utilisation of Python lists and dictionaries as a basis for a cross-recommendation and filtration system

* **April 29th:**
* *Usability and Functionality:*  
  + Participants generally noted that, the site’s presentation was still very straightforward with simple options, there hadn’t been a loss of intuitiveness while there had been an increase in visual aesthetic, and the redundant buttons, ‘game’ and ‘music’, had been removed. (*These are now features on the experimental version of the site).*
  + Participants would note that it had become much clearer to deduce why certain search results were being shown or why specific recommendations were being provided. Most noted that the addition of a section that showed the exact thematic link between their selection and the recommendations was a crucial change, however one did note “some thematic links are so hyper-specific that they go back around into not making any sense, like ‘aftercreditsstinger’”.
* *Accuracy of Queries and Recommendations:*
  + Most participants noted that the results had become noticeably more specific to their exact search terms and there were no longer ‘junk’ results being displayed.  
      
    In every scenario documented by participants, the piece of media being searched for was pulled and displayed, often shown first given exact title-matching, and if other results were shown, it was always the case that exact keywords were in those titles also.  
      
    All participants noted that recommendations had become much more substantive and sensible, with theme, author, rating, and age suitability being clearly taken into account. One participant did note that “comedic Shakespeare works were being recommended alongside ‘Dumb and Dumber’ which makes some sense but is also probably going to feel disparate to most people”.
* *Design and UI Comfortability:*
  + The points raised by all participants here were generally positive, being that the site felt modernised and smoother, with mentions of the new loading animations, error handling, and rounder, styled formatting to the entire site. One participant stated, “…and even if load times were long, it felt like something was happening given the loading animations.”