Logical AND Issues

Since the last report, the biggest issue we came across pertained to search queries, specifically those operating on the basis of logical AND. There were several issues that comprised the problem as a whole. First, the most pressing of them was the use of Firebase's Realtime Database as our primary database. The issue with this was that queries for both logical OR and AND searches became complex quickly due to the limited and restrictive qualifiers allowed for queries. While some simple workarounds allowed for logical OR searches, there was no simple, clear, and efficient method of performing AND searches without parsing the entire database client-side rather than through the server request.

To alleviate this issue, we migrated the database from Firebase's Realtime Database to Firebase's Firestore Database, which is built on top of the Realtime database and allows for more complex queries. The next issue was trying to implement a "full-text search" that would allow users to search not only the title for keywords but also the description. Ultimately, we chose to drop this feature based on cost grounds because there was no way of implementing such a style of search without paying for the upgraded "Blaze" plan from Firebase or paying for a secondary database that could handle those types of searches.

Instead of doing a "full-text search," we chose to implement a direct "title-keyword" pair search that operates similarly to a "full-text search." This method takes a list of search terms from the user and compares them to an array containing the keywords (determined by the curator) and title of the exhibit to check for existence in the array and returns the list of exhibits. Essentially, this method is a highly simplified version of a "full-text search."

Finally, the last issue that was encountered pertains more to the lack of knowledge of JavaScript syntax and features. This issue came to a head with the generation of logical AND queries. With different search options available, it led to roughly 30 or so AND-type queries. I took a naive approach to writing these queries, resulting in a code block of about 250 lines of if-else statements for each combination. Once I finished, I knew that there had to be a better method of generating AND queries. If my naive approach to the problem was the industry standard, then computer science has some big fish to fry.

After searching around to find a solution online with little success for our particular situation, I asked ChatGPT if it could improve my naive solution. Through it, the code was reduced to about 30 lines using an "Object/Array spread" operator ('...Object' or '...Array') in JavaScript, an operator that I didn't even know existed.