App Idea: The app will store analogies for computing concepts. These analogies will be searchable by the user. These analogies will serve to help explain a complex programming concept to someone who may not understand it otherwise.

Users:

- Computer science instructors (undergrad TAs, graduate TAs, professors, etc.)
- The users will have some level of computing expertise
- They may be older (close to retirement), so the app must be accessible

App Usage:

Finding Analogies (Requirement):
An instructor will open the app and search for an analogy. The app will display a list of analogies that match the search criteria. This list can be sorted in a variety of ways. The user can click on each analogy to see more information about it, including the different parts of the analogy.

Creating Analogies (Nice to have):
The user would select to create a new analogy, and the app would then walk them through the process of creating an analogy.

Comparing Analogies (Nice to have):
The user would be able to select one analogy, but still be able to find other analogies on the search page. Once a second analogy is selected, the user can compare the two analogies and how they benefit from the concept.

Analogy Popularity (Nice to have):
The app would display how often each analogy has been viewed, popular search terms, etc. This would let instructors know which analogies are popular and which ones people are looking for that don't currently exist.

Workflow:

The user (typically an instructor or GTA) will login to their account and browse through app analogies. They can search for an analogy that clarifies a topic, but also compare the analogies with others. They can "favorite" an analogy if they like it or it helps explain it to students. They can create new analogies, but the app will walk them through the components of the analogy.

Data:

- Text data associated with the actual analogies
- Analogy search data (measures of who's seeing what analogies and popular searches)