iOS Engineer Candidate Code Exercise

Write a sample Swift 4 app that fetches and displays data from a RESTful Web API. The app should be able to display the data as a text only, scrollable list of titles, and on phones, should be toggle-able from the list to a scrollable grid of item images.
The title and description for each item should each be parsed out of the data in the "Text" field. Images should be loaded from the URLs in the "Icon" field. For items with blank or missing image URLs, use a placeholder image of your choice.

Clicking on an item should load a Detail view, including the item's image, title, and description. You choose the layout of the Detail view.

On tablets, the app should show the list and detail views on the same screen. For phones, the list and detail should each be full screen.

The app should have an action bar that displays:

For Phone
The name of the app on the item list screen, and the title of the item on the detail screen
For Tablets
The name of the app

In addition, two versions of the app should be created. Each version has a different Name, package-name, and url that it pulls data from. (We're interested in your methodology for creating multiple apps from a shared codebase)
Version One

Name: Simpsons Character Viewer
Data API: http://api.duckduckgo.com/?q=simpsons+characters&format=json
Package name: com.xfinity.simpsonsviewer

Version Two

Name: The Wire Character Viewer
Data API: http://api.duckduckgo.com/?q=the+wire+characters&format=json
Package name: com.xfinity.wireviewer

Feel free to use open source libraries as you see fit, but we must be able to build and run your project in XCode. Before sending, consider building and running your project from a clean environment.
Please include a separate NOTES document (pdf, .doc, etc) outlining the major decisions you made while building the app and the motivations behind them, and, for any libraries you chose to use, explanations of what they do and why you chose them. (We're interested in the decision making process you use while developing)

As a bonus, implement one of the following (not required):
Local search functionality that filters the item list according to items whose titles or descriptions contain the query text
Functionality to favorite characters, and a drawer layout for navigating between a view of all characters, and a view of only favorited items
Animated transitions between the item list and detail screen (for phones), or animated detail-pane loading on tablets. You choose the complexity and character of the animations.

Please feel free to reach out with any questions you have, or if any part of the above is unclear or ambiguous.