Swift Assignment

This Assignment is intended to check your Swift knowledge and general development practices.

General Guidelines

- The output for this assignment should be a working Xcode project that compiles and runs out of the box on a recent iPhone.
- Create a new repo on Github, push the project, use incremental commits as you work on the exercise. Send us the link when completed.

How we review your work

The goal of this coding assignment is to help us identify what you consider clean code. Avoid using hacks and tricks that you would absolutely not use in production code. You should strive to deliver quality, production code that you are proud of, rather than just something that works. It is fine to leave some things aside provided you call them out in your project's README.We will review your work with the following topics in mind:

- Time-to-market: How much you can accomplish in the given deadline.
- App quality: Simple UI that fits the spirit of the platform, easy to use, and stable.
- Architecture: Use of proper architecture and design patterns.
- Clean code: Is the code simple and easy to understand? Is there any code smells or other red flags? Does the code follow general Swift guidelines and idioms? Does object-oriented code follow key principles such as the single responsibility principle? Is the coding style and naming generally consistent with the language's guidelines? Is the latest `Swift` used?In general, try to keep attention to details, care about the user experience and about writing elegant code.

The Assignment

Trending Repositories:

Create an iphone (iOS) application that displays the most trending repositories on GitHub that were created in the *last day*, *last week* and *last month*. The user should be able to choose which timeframe they want to view. In the same screen, the list of trending repositories sorted by the number of stars should be shown int a table.Each cell (repository) on the list should contain the following information:

- The username of the owner and the name of the repository (`owner->login` and `name` fields in the API response)
- The avatar of the owner as a small thumbnail (`owner->avatar_url` field). If no avatar exists, use a default "no avatar" image.
- The description for the repository (`description` field). if there is no description, add some default text to imply that.
- The number of stars (`stargazers_count` field)

Additional Features

- The list should allow for infinite scrolling, loading more items when the user reaches the end (or near it, for optimization)
- When a user taps on a cell, present a detail screen for the repository (could be full screen or modal, your choice), with all the former details and these additional ones:
    - Language, if available (`language` field)
    - Number of forks (`forks` field)
    - Creation date (`created_at` field)
    - a working link to the GitHub page of the repository (`html_url` field)
- The user should be able to add a repository to their own favourite list. The favourites repositories are saved locally and are available offline. There should be a Favorite Repositories screen that allows a user to view the favourite repositories, get their details and delete them. Favourited repositories should be shown as such in the main list.

Major Bonus Points

- The avatar images should be cached by you somehow, in order to avoid redownloading the same images over and over again.
- Create a UI that is better suitable for tablets (as well as the current one for the phone form factor), considering that there's some sort of list
- Implement search for each list.
- Provide a clear user experience when there is no internet connection.

GitHub API usage

The GitHub API allows for 60 unauthenticated requests per hour. If you need more than that, you can create a personal access token to be used with your GitHub account. You can configure the token so it won't have any private access. For more information: https://developer.github.com/v3/auth/#basic-authentication`curl` commands which you'll need to translate into `Swift`:

Created in the last month:

```
curl -G https://api.github.com/search/repositories --data-urlencode
"q=created:>`date -v-1m '+%Y-%m-%d'`" --data-urlencode "sort=stars"
--data-urlencode "order=desc" -H "Accept: application/json"
```

Created in the last week:

```
curl -G https://api.github.com/search/repositories --data-urlencode
"q=created:>`date -v-1w '+%Y-%m-%d'`" --data-urlencode "sort=stars"
--data-urlencode "order=desc" -H "Accept: application/json"
```

Created in the last day:

```
curl -G https://api.github.com/search/repositories --data-urlencode
"q=created:>`date -v-1d '+%Y-%m-%d'`" --data-urlencode "sort=stars"
--data-urlencode "order=desc" -H "Accept: application/json"
```

Each of these requests returns a maximum of 100 records. To get to the next page of records, you need to look at the `Link` header of the request which looks like this: `Link: <https://api.github.com/search/repositories?q=created%3A%3E2017-05-17&sort=stars&order=desc&page=2>; rel="next", <https://api.github.com/search/repositories?q=created%3A%3E2017-05-17&sort=stars&order=desc&page=34>; rel="last"`, and request the next URL that was crafted for you.

README

Write a README to accompany the code. It should address the following:

- General architecture of the application.
- Reasoning behind main technical choices.

- Features you didn't implement. This can also include details about how you would implement things differently if you were to spend more time on the assignment or if it was for production use.
- Anything else you feel should be included.

Deadline

Please send the complete assignment within 7 days.

Good luck, and we are here if you have any questions...