Proton iOS developer task

Overview

This exercise is meant to test your coding skills as well as how you approach the app development process.

Your task is to take the provided sample app, get it running and improve it based on the description in this document, as well as your own expectation of code and app quality. How you choose to approach this task is up to you and you should make your own judgements about what's most important to fix or implement first and take a best guess if unsure about something.

You should approach this app with the attitude that you and other developers may have to expand and maintain it for an unknown number of years and will likely outlive you working on it.

What we will be looking for in your solution

You can concentrate on the most important fixes to the app logic and on whatever you feel the app will most benefit from: architecture, tests, code smells, code style, different programming paradigm, tooling, anything else.

Please use this time to show what you believe is important for you as the developer and what would be most important for the product when working in a team.

A few characteristics we are especially interested in are:

- code that is maintainable long-term (easy to be changed or replaced in the future),
- pragmatic technical choices (choosing the right tool for the job),
- code that is easy to follow and understand for both your teammates and your future self.

One note — we use Git. If you are familiar with good git practices, please feel free to commit your changes with meaningful messages. However, it's not a requirement — if you prefer to not use git, it's fine too.

Time constraints

Please send us back the task within 3 days.

There is no hard time limit for how much time you need to spend on the task.

However, we recommend spending no more than 5-6 hours.

If you cannot dedicate this much time for the task, it's ok — spend as little as you can comfortably fit into your life. You are not expected to fix or change everything. If you identify something as taking a long time to fix, just leave it.

If you identify some things you would like to do but run out of time, please leave us some notes with your reasoning and explanations. Add them either as code comments, or attach them to the README file. We appreciate it.

Task

The purpose of this app is to retrieve basic weather information from a RESTful API (already in the source code) and present that information in master and detail screens. Basic data should be downloaded automatically at app launch and images downloaded on request by the user. It would also be good (not required though) if the app is able to fall back on a cache in case the API is unavailable.

Master Screen

The master screen should have "Forecast" in the navigation bar. Below the navigation bar should be a pair of tabs with the options "Upcoming" and "Hottest". Below these buttons should be a table view displaying the relevant forecasts and in the correct order.

- Upcoming: All upcoming days returned by the API in order of day
- Hottest: Any days that have less than a 50% chance of rain, ordered hottest to coldest

The rows should be in the format "Day {day}: {description}" where day and description are keys returned by the API. If the image for a particular day is downloaded (initiated by the user in the detail screen), then the row in the master view should distinguish this visually by changing the row's color or using a thumbnail.

Detail Screen

The navigation bar title should be in the form "Day {day}". Below the navigation bar the weather data should be laid out in a consistent manner as you see fit using appropriate units and values. Below the text data you should display the image (once downloaded). A "Download Image" button should be aligned to the bottom of the screen and disappear upon successful download.