Packeta iOS Developer Task

Overview

This exercise is designed to test your coding skills and approach to app development. Your task is to take the provided sample app, get it running, and improve it based on the description in this document, along with your expectations of code and app quality. How you approach this task is up to you, and you should make your own judgments about what's most important to fix or implement first, making the best guess if unsure about something.
You should approach this app as if it will be expanded and maintained by you and other developers for many years, potentially beyond your involvement.

What We Will Be Looking for in Your Solution

Concentrate on the most critical improvements to the app's logic and any changes that would benefit its architecture, tests, code quality, programming paradigms, tooling, etc. Use this time to showcase what you believe is essential for a maintainable product and teamwork.
Key aspects we are interested in:

- Code that is **maintainable** long-term (easy to change or replace in the future).
- **Pragmatic technical choices** (choosing the right tool for the job).
- Code that is **clear and understandable** for teammates and your future self.

We use **Git**. If you are familiar with good Git practices, please commit your changes with meaningful messages. However, this is not a strict requirement.

Time Constraints

Please aim to complete the task within **3 days**. We recommend spending no more than **5-6 hours**. However, if you can't dedicate this much time, do as much as you comfortably can. You are not expected to fix or change everything. If you identify areas that would take a long time to fix, you can leave them as they are and add notes in code comments or the README with explanations or future recommendations.

Task

The purpose of this app is to retrieve basic **Pokémon information** from a RESTful API (PokéAPI: https://pokeapi.co) and present that information in **master** and **detail** screens. Pokémon data should be downloaded automatically at app launch. Each Pokémon's image should be downloadable on request by the user in the detail screen. Additionally, if the app could fall back on a cache when the API is unavailable, that would be a good (but not required) feature.

Master Screen

The master screen should display “Pokémon” in the navigation bar. Below the navigation bar should be a segmented control with the options “All,” “Male,” “Female,” and “Genderless” (based on Pokémon genders returned by the PokéAPI). Below the segmented control, there should be a table view displaying the relevant Pokémon according to the selected gender filter, in the appropriate order.

- **All**: Shows all Pokémon retrieved from the API in their default order.
- **Male**: Shows only Pokémon that have male gender data.
- **Female**: Shows only Pokémon that have female gender data.
- **Genderless**: Shows only Pokémon without male or female gender data.

Each row in the table should display the Pokémon’s name. If an image for a Pokémon has been downloaded (initiated by the user on the detail screen), the row in the master view should visually distinguish this by changing the row’s color or adding a thumbnail of the Pokémon’s image.

Detail Screen

The navigation bar title should be the Pokémon’s name. Below the navigation bar, display the Pokémon’s data in a consistent layout of your choice, showing relevant attributes such as **height**, **weight**, and **type(s)**.
Below this data, display the Pokémon’s image (if downloaded). A **“Download Image”** button should be aligned to the bottom of the screen. This button should disappear upon successful image download.

Notes

This task requires adapting the app’s functionality and UI from the weather API to work with the PokéAPI. Feel free to structure the app according to your own standards for readability, maintainability, and team collaboration.