# Tech specifications

Using Swift in coding | Targeting iOS 11.0+ | Tested on iPhone 8 and iPhone X | Capable with rotations

Yet supporting iPad

# Highlights

Employee iOS App

Yaoli Ma

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Couple things I would like to bring up are design pattern, code structuring and couple small details which made app looks much better.   * Used MVVM design pattern in this project (I’m also familiar with other pattern like MVC, VIPER as well, we could discuss more when opportunity allows) * Asyncnise call with proper loading (a third party lib has been used – Shimmer, to handle the loading user experience. It blinks the dummy image tile the actual image loaded) * Data binding (applied observable to the data object, and link to the UI element, the benefit is that once data model gets updated, UI will automatically reflect the change) * Network (API serice call) (3rd party library Alamofire is used here to handle api calls, comparing to AFNetwork, it is very handy in terms of exception handling. I used this a lot in my projects that I have done in the past) * Caching (Used a simple way to caching data, could also use other mechanism, like Core Data, SqlLite, event NSUserDefaults (small))  TODOThere are always some more could be added into the project. Like:  1. Improve the UI 2. Paging 3. Increase test coverage 4. Better exception or edge case handling 5. iPad supporting |  | |  | | --- | |  | |  | |  | | How did time spend In total, I dedicated roughly 6 hours, effort would be saved, if less detailing or simplify the documentation.   * Understanding of the project requirement and planing the development – 30 mins * Built base structure (ex. Pods, stubing, configuration …) – 45 mins * API Service, data model – 30 mins * UI layout, data binding – 60 mins * Detailing, cosmetic polishing – 60 mins * Unit testing – 25 mins * Documentation – 60 mins * Else? (ex. debuging, self testing…) – 30 mins | |