# READ ME

# Starling Bank iOS Technical Task.

I implemented 3 different services: AccountsService, TransactionService, SavingsService, all of them are contained inside a CoreService.
I created a ProfileFacade to perform every api call using these services without exposing them to the ViewController.
ProfileFacade's method getTransaction is responsible for generating a ProfileViewModel, which is used to update the ProfileViewController.
ProfileViewModel contains an array of TransactionViewModel,  used to populate ProfileViewController's tableViewCells, and a RoundUpViewModel which is used to update ProfileViewController's RoundUpView.
ProfileViewController is RoundUpView's delegate and it's responsible for calling ProfileFacade to transfer RoundUpViewModel's Amount back to the API by calling ProfileFacade's method createSavingsGoal.
I prepared a coordinator for the ProfileViewController, just for completeness, even though it's not used.
Every service and ProfileFacade conform to a specific protocol that is used to create mocks and flexible dependency injection.
There's a basic unit test coverage for viewModels and services.
Services are tested using JSON files.
Due to time constraints I didn't write any UI tests, but I did prepare a CoreServiceMock class, containing every other mock service, so both ProfileViewController and ProfileFacade can be used for UI test as well, by injecting mock classes available in the project.
If I had more time, I would have tested the UI with snapshot tests.
No third party framework is used.

Thanks,
Alessandro Loi