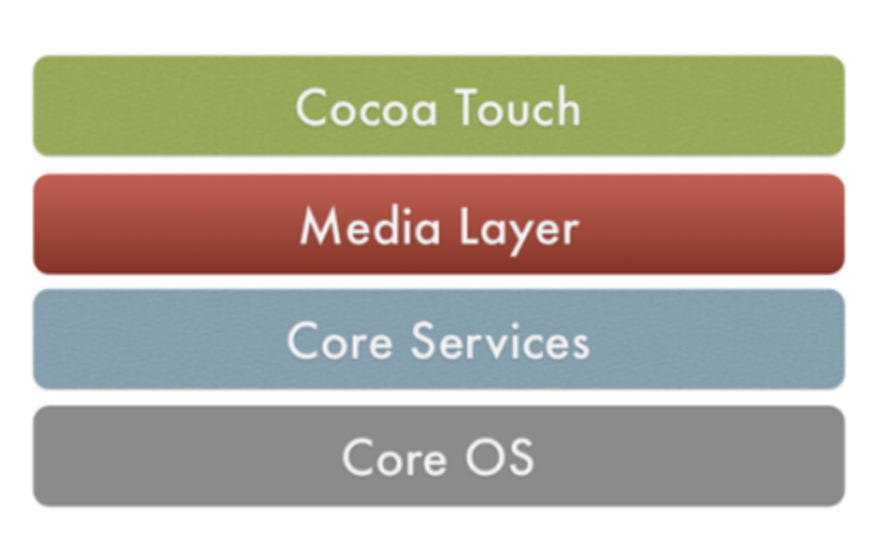
Public api: <https://github.com/public-apis/public-apis>

Ios patern designs: <https://viblo.asia/p/ios-architecture-patterns-mvc-mvp-mvvm-ban-da-hieu-ro-chua-bWrZnPyO5xw>

I. IOS application architecture:



**Core OS includes:**

* Core Bluetooth Framework.
* Accelerate Framework.
* External Accessory Framework.
* Security Services framework.
* Local Authentication framework

**Core Service includes:**

- ****Address book framework****– Gives programmatic access to a contacts database of user

- ****Cloud Kit framework****– Gives a medium for moving data between your app and iCloud

- ****Core data Framework –**** Technology for managing the data model of a Model View Controller app

- ****Core Foundation framework –**** Interfaces that give fundamental data management and service features for iOS apps

- ****Core Location framework –**** Gives location and heading information to apps

- ****Core Motion Framework –**** Access all motion-based data available on a device. Using this core motion framework Accelerometer based information can be accessed

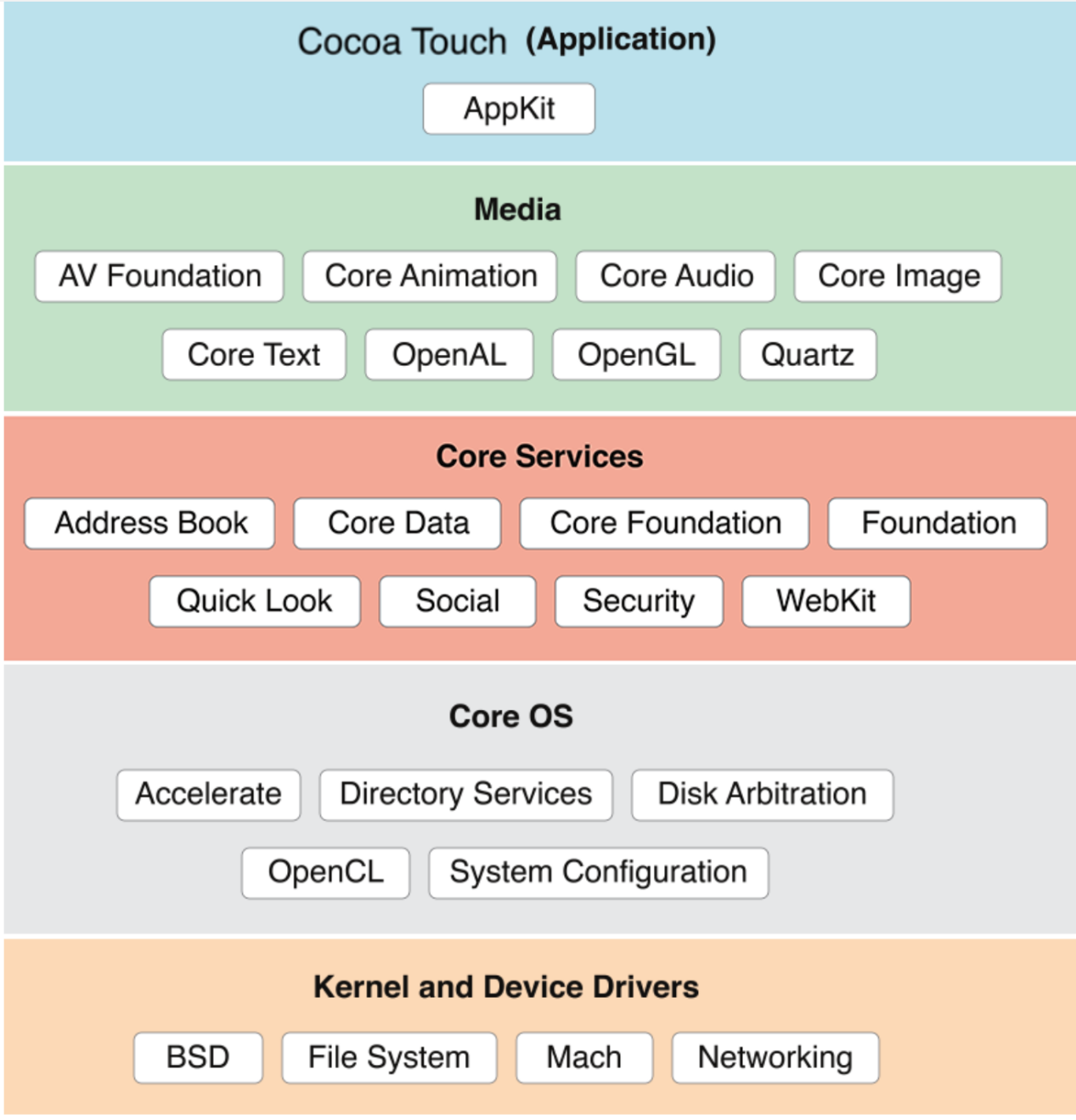
- ****Foundation Framework –**** Objective C covers too many of the features found in the Core Foundation framework

- ****Healthkit framework –**** New framework for handling health-related information of the user

- ****Homekit framework –**** A new framework for talking with and controlling connected devices in a user’s home

- ****Social framework –**** Simple interface for accessing the user’s social media accounts

- ****StoreKit framework –**** Gives support for the buying of content and services from inside your iOS apps, a feature is known as In-App Purchase



3. Media layer:

*Graphics Framework:*

- UIKit graphics: use for support designing images and use for animating contents of your view.

- Core graphic framework: is drawing engine for ios apps, customs 2D vector and image-based rendering.

- Core animation: technique for optimizing animation in ios apps.

- Core image: support for video and motionless image.

- OpenGL and GLKit: manager 2D, 3D rendering by hardware-accelerated interfaces

*Audio Framework:*

- Media player framework:

- AV Foundation: handling recording and playback of video and audio.

- Open AL: technology for audio.

*Video Framework:*

- AV Kit: presenting video

- AV Foundation: playback and record video

- Core media:

1. Cocoa Touch:

***II. Memory manager:***

<https://viblo.asia/p/co-phai-khi-nao-cung-can-dung-weak-self-trong-closures-gAm5y4bLldb>

## **1. Automatic Reference Counting**

- Bộ nhớ trong Swift được quản lí theo cơ chế ARC, nó hoạt động theo cơ chế tính số lượng reference đến vùng nhớ

Nó sẽ được giải phóng khi không có đối tượng nào tham chiếu đến nó. Vì thế trong một số trường hợp bạn không giải phóng được do vẫn có strong reference đến nó.

- Một số phương pháp để kiểm tra memory leak:

* Kiểm tra trong hàm deinit() khi ViewController bị đóng.
* Quan sát mức độ bộ nhớ tăng dần
* Sử dụng tool Allocations Instruments

## **2. Unowned, weak và Strong - weak**

- Closues có thể strong capture bất cứ biến hay hằng số trong context mà nó được định nghĩa. Ví dụ bạn sử dụng self trong closures thì nó sẽ được duy trì trong suốt vòng đời của scope đó. Và nếu self vẫn giữ strong reference sẽ dẫn đến strong reference cycle => phải sử dụng [weak self] và [unowned self]

- [unowned self]: nếu object bị nil => bị crash.

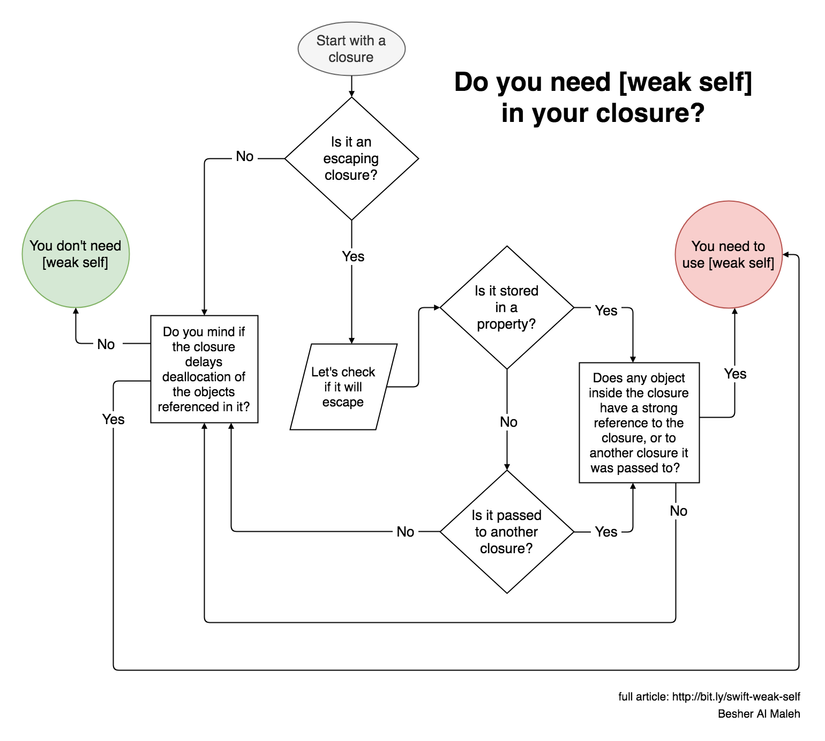
## **3. Escaping và non-escaping closures**

- Non-escaping closures: thực thi code trong scope của nó một cách tức thì và không có khả năng lưu trữ hay sử dụng sau đó => không gây ra reference cycle => ko cần sử dụng unowned và weak.

- Escaping closures: có thể lưu trữ nó vào 1 biến hoặc 1 closure khác và có thể thực thi nó trong tương lai

=> nếu closues đc  lưu trữ vào 1 biến hoặc 1 closures khác

hoặc có sử dụng self => fai sử dụng [unowned self] và [weak self]



### **6.1 Grand Central Dispatch**

- Dispatch thường sẽ ko gây ra reference cycle

- dispatch được lưu trữ vào 1 biến thì sẽ gây ra leak

func leakyDispatchQueue() { let workItem = DispatchWorkItem { self.view.backgroundColor = .red } DispatchQueue.main.asyncAfter(deadline: .now() + 1.0, execute: workItem) self.workItem = workItem // stored in a property }

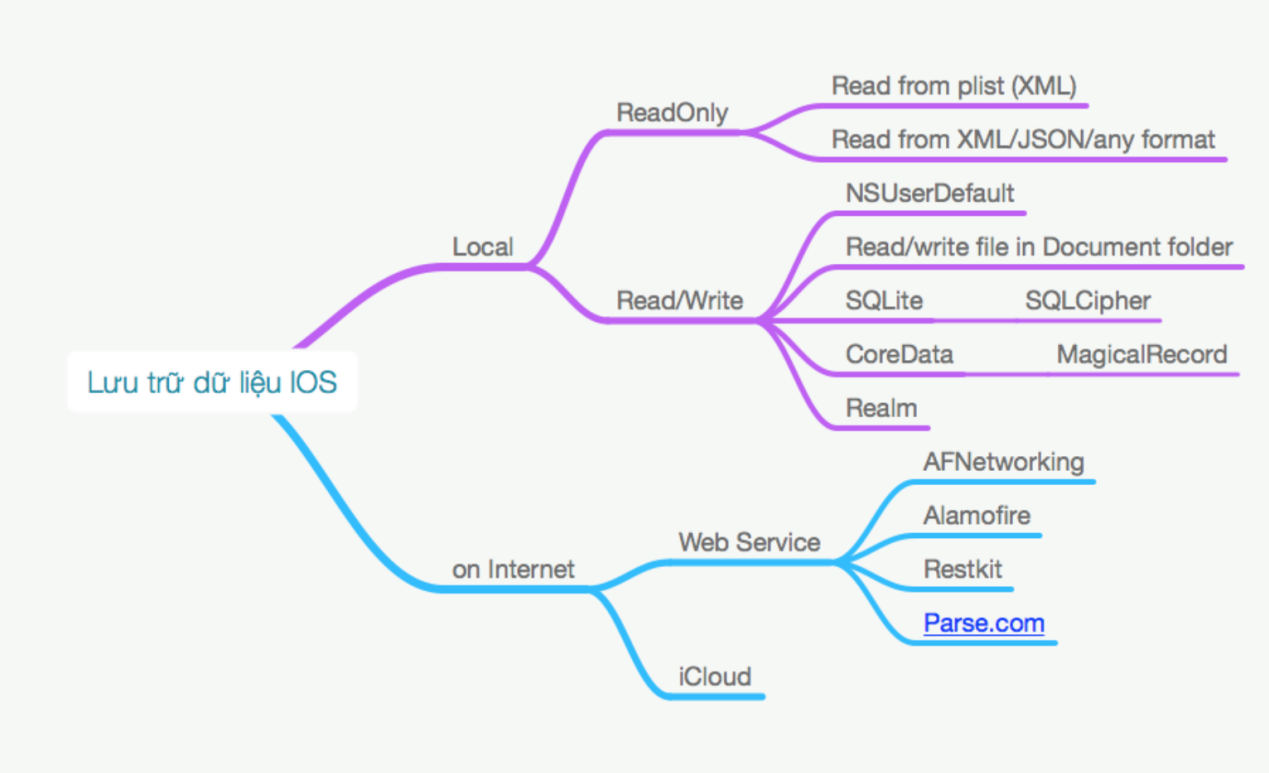
### **6.2 UIView.Animate and UIViewPropertyAnimator**

Tương tự như GCD nó cũng không gây ra memory leak nếu không lưu trữ nó vào thuộc tính.



**III. IOS Backend services:**

<https://blog.back4app.com/ios-backend-service/>



1. **RESTFull API:**

- RestFul la 1 chuẩn để thiết kế các api cho web service.

- API (Application Programing Interface): là 1 tập hợp các cơ chế để các ứng dụng tương tác với nhau. API trả về kiểu dữ liệu Json, xml.

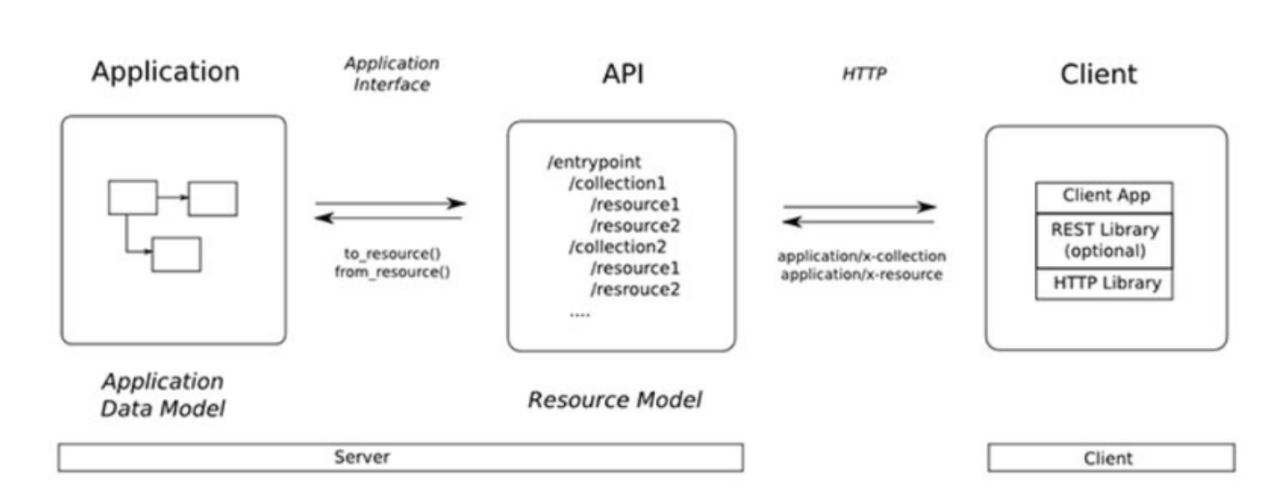
- REST: là 1 kiểu cấu trúc để viêts API

+ nó sử dụng phương thức HTTP để tạo giao tiếp giữa các => thay vì sử dụng 1 URL để sử lý thông tin người dùng, Rest gửi các yêu cầu http: get, post, delete... đến 1 URL để sử lý dữ liệu.

+ Rest quy định việc sử dụng các http method: get, post, , put...

+ Rest quy đinh cách định dạng URL cho ứng dụng web để xử lý resource.

- Nguyên lý hoạt động của restful api:



1. **WebSocket:**

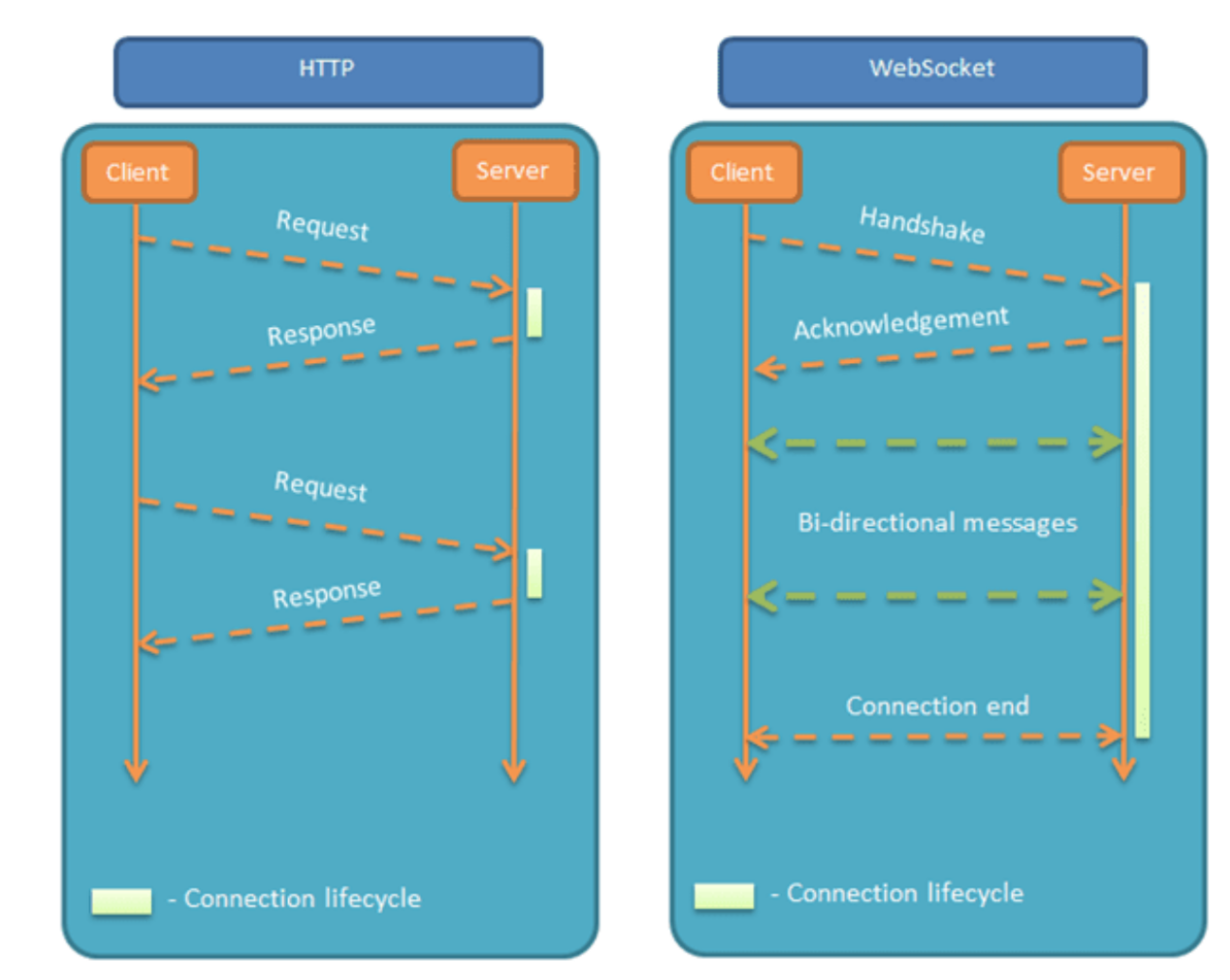
- là giao thức mạng, cho phép kết nối 2 chiều giữa client và web.

- Websocket rất hữu dụng cho các ứng dụng di động, cho phép các ứng dụng này tái sử dụng lại các công cụ, API được xây dựng cho ứng dụng web, như việc chúng ta tái sử dụng các API trên giao thức HTTP.

* WebSocket là một phương thức giúp máy trạm và máy chủ có thể giao tiếp thời gian thực 2 chiều với nhau.
* Khả năng giảm độ trễ của mạng đến mức tối đa và vượt trội hơn HTTP. Nếu bạn phát triển các ứng dụng thời gian thực như: sàn tiền ảo, Game, chat hay video call,… sử dụng WebSocket sẽ mang lại trải nghiệm tối ưu nhất cho người dùng.
* Khả năng giảm kích thước header HTTP lên đến 1000 lần! Nếu một request HTTP có header nặng 871 byte, khi sử dụng WebSocket, khối lượng header của request chỉ còn 2 byte.

- Sử dụng trong các app: theo real-time, game online, app chat.

- Thư viện cho ios app: **SocketRocket** sử dungj objective-



1. **HTTP/HTTPs:**

**- là giao thức 1 chiều**dựa theo giao thức TCP, bạn có thể tạo ra các kết nối dựa vào request HTTP, sau khi kết nối được thực hiện xong và được phản hồi lại, quá trình sẽ kết thúc và đóng lại.

- Ứng dụng: dùng trong các app Restful đơn giản, chỉ nhận thông tin 1 chiều về server để xử lý => phản hồi => đóng kết nối.

**IV: Push Notification:**

Có 2 loại :

- local notification:

- Remote notification (push notification): sử dụng các remote push như Firebase, SNS Amazon cloud service thông qua APNS.

1. **Local Notifications:**

-Registering UNUserNotificationCenter in file appdelegate.swift, UNUserNotificationCenterDelegate

- Sending a noti: using UNTimeIntervalNotificationTrigger

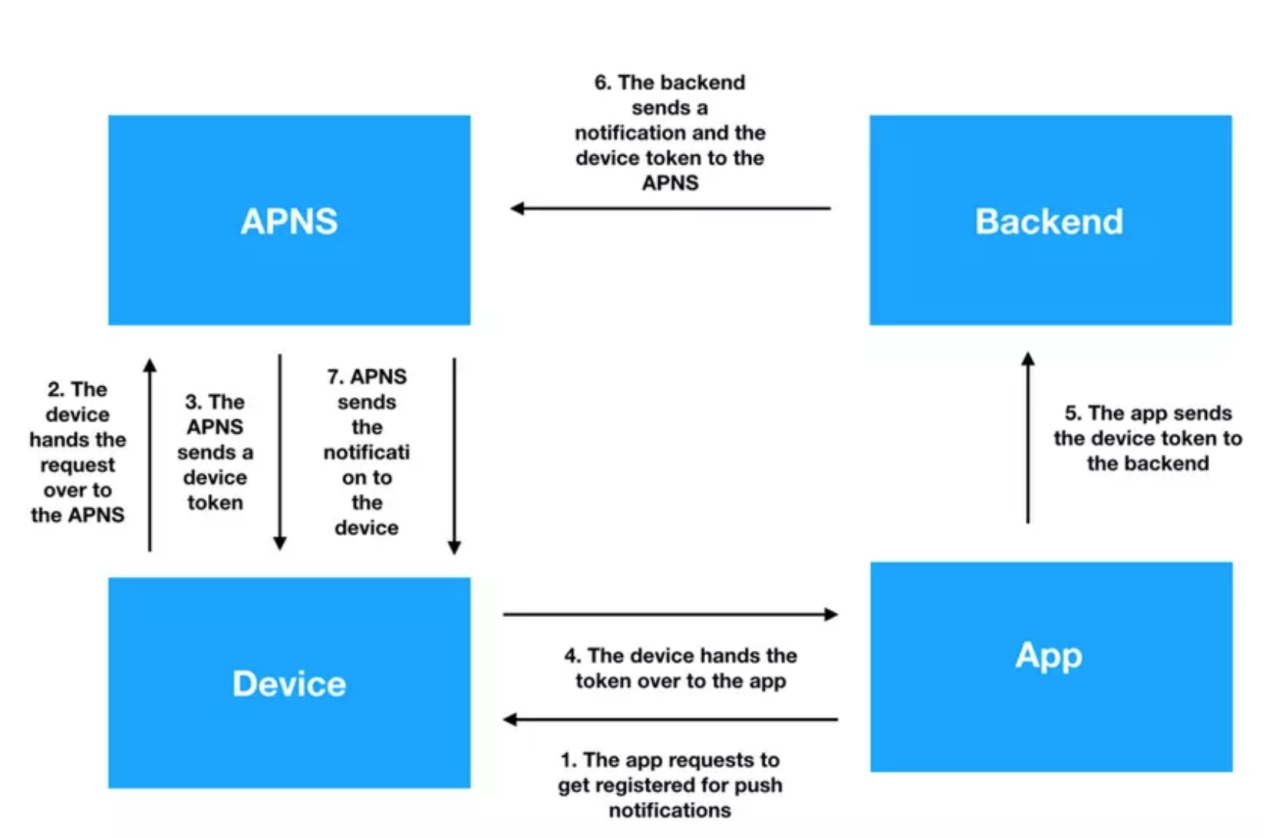
UNCalendarNotificationTrigger

UNLocationNotificationTrigger

1. Remote Notifications:

- Là dạng push được send bằng backend tức là bằng server.

1. APNS: Apple push notification service



**V. Biometrics:**

- là bảo mật bằng vân tay, Touch ID

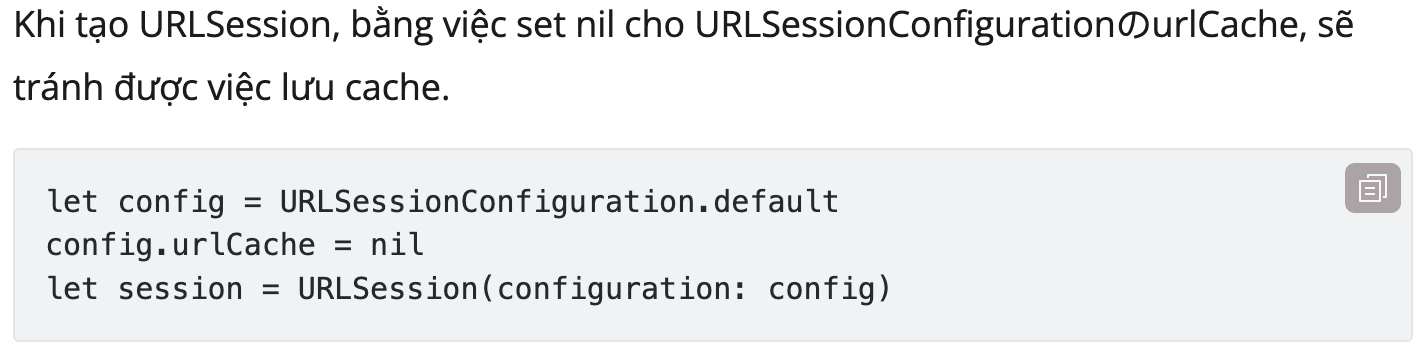
- sử dụng ***LocalAuthentication***

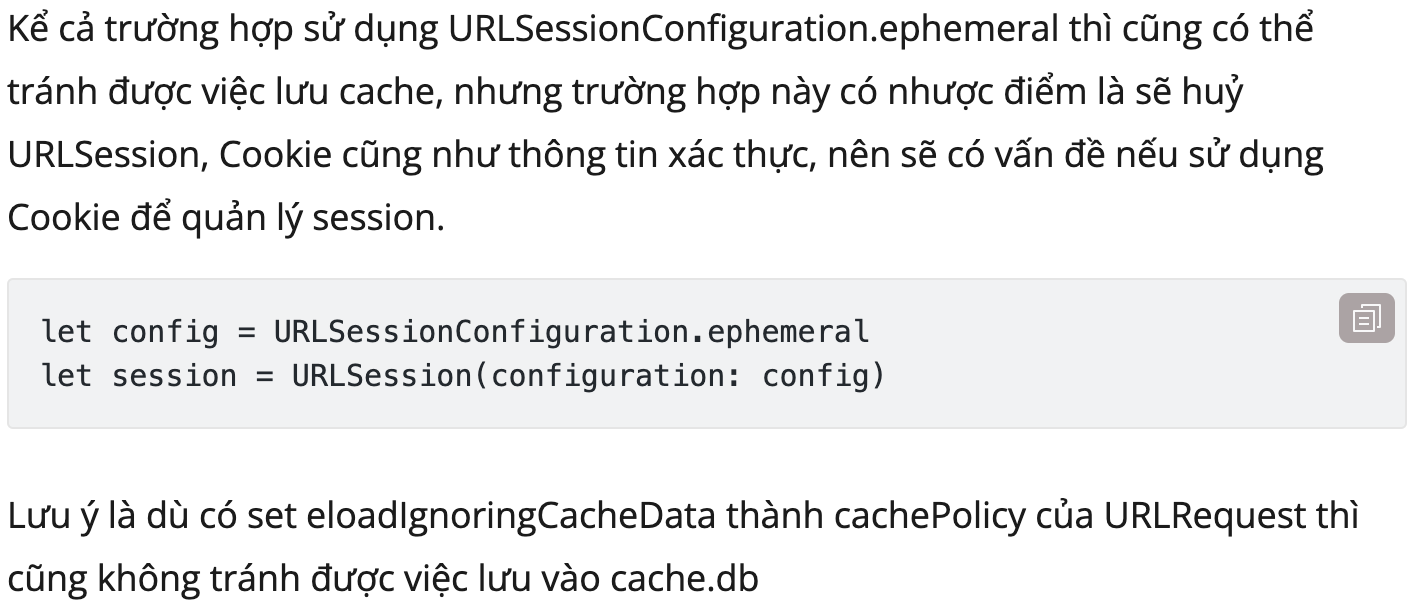
**VI: Security for ios app:**

1. **[iOS] Cache dữ liệu truyền:**

- Nếu thực hiện truyền dữ liệu bằng URLSession,..thì dữ liệu truyền bởi OS sẽ bị lưu tự động vào file (cache.db) trong thiết bị. Dữ liệu cache này do được lưu bằng plain text, kể cả truyền bằng HTTPS, nên có khả năng cung cấp thông tin có lợi cho kẻ tấn công.

- Giải phaps:





1. **[iOS] App Transport Security(ATS):**

Rủi ro

Nếu vô hiệu hoá App Transport Security (ATS) thì đồng nghĩa với việc chấp nhận truyền thông tin bằng phương thức không an toàn như truyền bằng plain text. Bên cạnh đó, có khả năng trong tương lai Apple sẽ bắt buộc hoá việc cài đặt ATS.

Giải pháp

Nếu phía server có khả năng đáp ứng thì nên cài đặt App Transport Security (ATS).

/////////////////////// need search ////////////////////

* Experience working with remote data via REST and JSON, WebSocket, HTTPS.
* Familiarity with RESTful APIs
* Experience with Apple Portals, App Store Connect Workflow, Generating and Exporting
* Knowledge of Apples's design principles, application interface guidelines, iOS back-end Services.
* Development and Distribution Profiles and Certificates.
* Familiarity with Remote and In App Push Notifications, Biometrics, Mobile App Security
* Principles, Encryption Algorithms, Keycloak.

Knowledge of Firebase services, especially Crashlytics and App Distribution

* Knowledge of C-based libraries, third-party libraries and APIs.
* Strong technical background, strong knowledge of xCode and GIT procedures.
* Strong knowledge over Objective C, Swift and Cocoa Touch, Object Oriented Concepts, Interface Builder, Storyboards and XIBs, **UIKit**
* Kinh nghiệm code review

Có kinh nghiệm lựa chọn kiến trúc

Knowledge of SwiftUI design principles, patterns

**reading and writing English (technical document)**

* experience in building **Custom UI, Autolayout** (NSLayoutConstraint, Visual Format Language, AutoresizingMask, NSLayoutAnchor), Animations.
* Strong working knowledge with the main iOS architectures (MVVM, Clean Swift/VIP, VIPER)

Familiar with Agile practices, SOLID principles

* Familiar with asynchronous programming
* Strong working knowledge of **View Life cycle** and **Custom UI**

Có kinh nghiệm sử dung và làm việc với các hệ quản trị CSDL MySQL, SQLite

Experience with RESTful programming with JSON and XML.

Deep understanding of XCode, iOS SDK.

Quy trinh phát triển phan mềm

Able to identify and recommend best practices

Có kinh nghiệm làm việc với các mô hình kiến trúc phổ biến như MVC, MVVM.

Kỹ năng tốt về xử lý nhiều khung màn hình khác nhau (trên cả smartphone và tablets)

MVC/MVVM architecture

/////// need search ///////////

1. DispatchQueue la gi?

1. Protocol, generic, extension la gi?
2. Lập trình POP (Protocol Oriented Programming)
3. swiftUI
4. Reference count