ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

§Æng H÷u Anh – Tin ph¸p – K50 ★ M¹ng metro ethernet (ME) vµ triÓn khai dÞch vô trªn m¹ng ME ★ Hµ Néi, 6-2010

Nguyễn Viết Hạnh

Phạm Mạnh Hưng

Nguyễn Hà Thanh

Hà Thị Thúy

Ionic Mobile Framework

Ngành: Công nghệ thông tin

Chuyên ngành: Kỹ Thuật Phần Mềm

Giáo viên hướng dẫn : **TS** **Ngô Thị Huyền Châu**

Hà Nội, 3/2016

Mục Lục

Contents

[**Danh Mục Hình Vẽ** 4](#_Toc446968012)

[**Danh Mục Bảng và Biểu Đồ** 5](#_Toc446968013)

[Chương 1. Cơ sở lý thuyết 6](#_Toc446968014)

[1.1. Các hình thức phát triển ứng dụng di động 6](#_Toc446968015)

[1.1.1. Các loại ứng dụng di động 6](#_Toc446968016)

[1.1.2. Đặc điểm của ứng dụng di động 6](#_Toc446968017)

[1.1.3. Cách thức hoạt động chung của ứng dụng di động 8](#_Toc446968018)

[1.2. Ionic Framework 9](#_Toc446968019)

[1.2.1. Tổng quan về nền tảng Ionic 9](#_Toc446968020)

[1.2.2. Kiến trúc của Ionic 10](#_Toc446968021)

[1.2.3. Các đặc điểm của Ionic 14](#_Toc446968022)

[1.2.4. Các thành phần chính của Ionic 16](#_Toc446968023)

[1.3. Một số nền tảng tương tự 19](#_Toc446968024)

[1.3.1. Famous 19](#_Toc446968025)

[1.3.2. OnsenUI 22](#_Toc446968026)

[1.3.3. Sencha Touch 23](#_Toc446968027)

[1.3.4. Kendo UI 24](#_Toc446968028)

[1.3.5. F7 Framework 25](#_Toc446968029)

[Chương 2. Phân tích thiết kế phần mềm 27](#_Toc446968030)

[2. Phân tích và đặc tả yêu cầu 27](#_Toc446968031)

[2.1. Giới thiệu 27](#_Toc446968032)

[2.1.1. MỤC ĐÍCH 27](#_Toc446968033)

[*2.1.2. PHẠM VI* 27](#_Toc446968034)

[2.1.3. NGOÀI PHẠM VI 27](#_Toc446968035)

[2.2. Mô tả chung 27](#_Toc446968036)

[2.2.1. TỔNG QUAN VỀ SẢN PHẨM 28](#_Toc446968037)

[2.2.2. CHỨC NĂNG SẢN PHẨM 28](#_Toc446968038)

[2.2.3. ĐỐI TƯỢNG NGƯỜI DÙNG 28](#_Toc446968039)

[2.2.4. CÁC RÀNG BUỘC PHÁT TRIỂN 28](#_Toc446968040)

[2.2.5. GIẢ THIẾT 29](#_Toc446968041)

[2.3. Yêu cầu chi tiết 29](#_Toc446968042)

[2.3.3. YÊU CẦU HIỆU SUẤT 39](#_Toc446968043)

[2.3.4. RÀNG BUỘC THIẾT KẾ 39](#_Toc446968044)

[Chương 3. CÀI ĐẶT VÀ TRIỂN KHAI 40](#_Toc446968045)

[3.1. CÀI ĐẶT NODEJS 40](#_Toc446968046)

[3.2. CÀI ĐẶT CORDOVA 42](#_Toc446968047)

[3.3. CÀI ĐẶT IONIC FRAMEWORK 43](#_Toc446968048)

[3.4. KHỞI TẠO ỨNG DỤNG 45](#_Toc446968049)

[3.5. TRIỂN KHAI CODE TRÊN KHUNG CỦA ỨNG DỤNG 46](#_Toc446968050)

[3.5.1. CẤU TRÚC THƯ MỤC 46](#_Toc446968051)

[3.5.2. CẤU TRÚC MÃ NGUỒN 47](#_Toc446968052)

[3.6. CHẠY ỨNG DỤNG 49](#_Toc446968053)

[3.7. So sánh cài đặt ứng dụng bằng Ionic và Android 57](#_Toc446968054)

[3.8 Đánh giá 59](#_Toc446968055)

[Kết Luận 59](#_Toc446968056)

[Tài liệu tham khảo 61](#_Toc446968057)

**Danh Mục Hình Vẽ**

[Hình 1.1 Đặc điểm của ứng dụng di động 7](#_Toc446885136)

[Hình 1.2 Cách thức hoạt động của ứng dụng di động 9](#_Toc446885137)

[Hình 1.3 Mô hình cảm quan Ionic 9](#_Toc446885138)

[Hình 1.4 Kiến trúc Ionic 11](#_Toc446885139)

[Hình 1.5 Angular’s operation 12](#_Toc446885140)

[Hình 1.6 Kiến trúc Cordova 13](#_Toc446885141)

[Hình 1.7 Kiến trúc Plugin Cordova 13](#_Toc446885142)

[Hình 1.8 Cấu trúc Ionic 14](#_Toc446885143)

[Hình 1.9 Ionic Navigation 16](#_Toc446885144)

[Hình 1.10 Ionic content container 17](#_Toc446885145)

[Hình 1.11 Side menu 19](#_Toc446885146)

[Hình 1.12 Kiến trúc tổng quan Famous 20](#_Toc446885147)

[Hình 1.13 Kiến trúc BEST 20](#_Toc446885148)

[Hình 1.14 So sánh BEST và MVC 21](#_Toc446885149)

[Hình 1.15 Mô hình của Sencha Touch 23](#_Toc446885150)

[Hình 1.16 Kiến trúc của Kendo UI 24](#_Toc446885151)

[Hình 1.17 Các UI elements của Framework7 25](#_Toc446885152)

[Hình 3.18. Trang Tải chương trình Nodejs 40](#_Toc446885153)

[Hình 3.19. Kiểm tra phiên bản Nodejs 41](#_Toc446885154)

[Hình 3.20. Cài đặt cordova trên Windows 10 42](#_Toc446885155)

[Hình 3.21. Kiểm tra phiên bản Cordova 43](#_Toc446885156)

[Hình 3.22. Cài đặt Ionic Framework trên Windows 10 44](#_Toc446885157)

[Hình 3.23. Kiểm tra phiên bản Ionic Framework 44](#_Toc446885158)

[Hình 3.24. Tạo ứng dụng mới với Ionic Framework 45](#_Toc446885159)

[Hình 3.25. Cấu trúc trong thư mục www 46](#_Toc446885160)

[Hình 3.26. Ionic Framework hỗ trợ kiểm thử bằng chính trình duyệt 49](#_Toc446885161)

[Hình 3.27. Hình ảnh ứng dụng trên trình duyệt 50](#_Toc446885162)

[Hình 3.28. Thêm nền tảng Android 51](#_Toc446885163)

[Hình 3.29. Hình ảnh ứng dụng trên Bluestack 52](#_Toc446885164)

[Hình 3.30 adb install 52](#_Toc446885165)

[Hình 3.31 Cài ứng dụng vào điện thoại 53](#_Toc446885166)

[Hình 3.32 Upload ứng dụng lên kho 53](#_Toc446885167)

[Hình 3.33 Lưu app trên server 54](#_Toc446885168)

[Hình 3.34 Share app qua email 54](#_Toc446885169)

[Hình 3.35 Tạo app trực tiếp trên web 55](#_Toc446885170)

[Hình 3.36 Tải source code từ web 56](#_Toc446885171)

[Hình 3.37 Chạy app 56](#_Toc446885172)

[Hình 3.38 Cấu trúc Android project 58](#_Toc446885173)

**Danh Mục Bảng và Biểu Đồ**

[Bảng 2.1 Chức năng sản phẩm 26](#_Toc446542424)

# Chương 1. Cơ sở lý thuyết

* 1. Các hình thức phát triển ứng dụng di động

Với sự gia tăng mạnh mẽ cả về số lượng thiết bị di động và đa dạng về nền tảng di động, các nhà phát triển ứng dụng di động tùy theo mục tiêu của ứng dụng mà chọn lựa các nền tảng tương ứng khác nhau, như Apple iOS, Google Android, Microsoft Windows Phone hay Samsung Tizen…

Số lượng các ứng dụng cho các thiết bị di động gia tăng nhanh chóng nhờ cộng đồng đông đảo của các nhà phát triển ứng dụng di động. Các ứng dụng này dựa trên các đặc trưng nhất định mà được phân nhóm thành ba loại ứng dụng di động: ứng dụng gốc (native app), ứng dụng web (web app) và ứng dụng lai (hybrid app). Trong phần này, báo cáo sẽ giới thiệu đặc trưng của các loại ứng dụng di động này và kiến trúc chung của chúng.

* + 1. Các loại ứng dụng di động
       1. Native Apps

Native app được xây dựng bằng ngôn ngữ lập trình tương ứng được hỗ trợ bởi các nền tảng mà nó hướng tới, như ứng dụng native cho Google Android viết bằng Java, ứng dụng native cho Apple iOS viết bằng Objective C hay ứng dụng C# cho Microsoft Windows Phone. Mỗi native app chỉ chạy được trên một nền tảng nhất định mà không thể chạy trên nền tảng khác. Các ứng dụng loại này được phân phối thông qua các kho ứng dụng tương ứng và được cài trực tiếp lên thiết bị di động.

* + - 1. Web Apps

Ứng dụng web là các ứng dụng chạy trên nền web, được viết bằng các ngôn ngữ web như HTML5, CSS và JavaScript. Các ứng dụng này chạy trên trình duyệt web của thiết bị và có thể hoạt động trên các nền tảng khác nhau. Khác với các ứng dụng native, ứng dụng web di động được phân phối thông qua web và không phải cài đặt trực tiếp lên thiết bị.

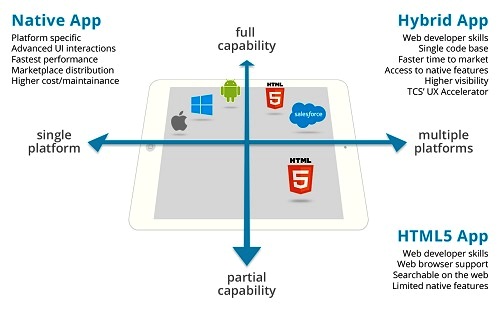
Về cơ bản, ứng dụng web di dộng có phần khác với trang web cho di động. Cụ thể là, các trang web cho di động là các trang tĩnh được định dạng theo chuẩn HTML/CSS/JavaScript để sử dụng cho thiết bị di động, còn ứng dụng web di động kết nói tới một cơ sở dữ liệu back-end và yêu cầu khối xử lý logic (business logic) để dáp ứng các yêu cầu chức năng cần thiết.

* + - 1. Hybrid Apps

Hybrid app là ứng dụng gồm có các đặc điểm của cả ứng dụng native và ứng dụng web di động. Cụ thể là, các phần cơ bản của ứng dụng vẫn được viết bằng ngôn ngữ web nhưng được đóng gói lại trong một vỏ bọc của nền tảng cụ thể. Lớp vỏ bọc này cung cấp những phẩm chất của một ứng dụng native như hoàn toàn tích hợp thiết bị, cài đặt trực tiếp lên thiết bị và được phân phối thông qua các kho ứng dụng.

* + 1. Đặc điểm của ứng dụng di động

Một ứng dụng di động, tùy theo loại mà ứng dụng đó thuộc về, sẽ có những đặc trưng nhất định của nó.



Hình 1. Đặc điểm của ứng dụng di động

Hình 1.1. thể hiện các đặc điểm của mỗi loại ứng dụng di động xét trên 2 phương diện chính là khả năng tương thích và đa nền tảng. Từ hình 1.1 có thể thấy rằng, một ứng dụng native app chỉ hỗ trợ một nền tảng nhất định nên chi phí để phát triển và bảo trì thường đắt đỏ, và việc phân phối các ứng dụng thường thông qua chợ ứng dụng hoặc một kho chia sẻ ứng dụng do bên thứ 3 cung cấp. Điều này dẫn tới việc nhà phát triển phải lệ thuộc vào nhà phân phối ứng dụng, và khi vì một lý do nào đó, ứng dụng được phát triển theo hướng này không được chấp nhận trên chợ ứng dụng, thì việc thu hồi chi phí phát triển ứng dụng có thể không bao giờ thực hiện được. Một ví dụ minh họa cho luận điểm này chính là việc rất nhiều ứng dụng Android bị Google loại bỏ khỏi chợ ứng dụng Google Play vì lý do sử dụng một số APIs không được khuyến khích hoặc việc sử dụng một số hình ảnh không rõ nguồn gốc xuất xứ, có tranh cãi trong quyền sở hữu trí tuệ…đây có thể coi như là một rủi ro lớn mà nhà phát triển native app luôn phải cân nhắc và lựa chọn các tài nguyên sử dụng cho ứng dụng của mình.

Mặc dù phát triển ứng dụng native app có một số thách thức như trên, nhưng một ứng dụng native có khả năng tương thích hoàn toàn với thiết bị chạy nền tảng tương ứng đem đến người dùng trải nghiệm tốt nhất, cả về hiệu năng lẫn giao diện thiết kế và khả năng tương tác. Các ứng dụng đòi hỏi đồ họa cao hoặc cần sử dụng các APIs trực tiếp theo tác với các component của thiết bị di động, ví dụ GPS, các sensor…thì ứng dụng native app là lựa chọn số một mà nhà phát triển hướng tới.

Một hướng phát triển khác của ứng dụng di động chính là ứng dụng web di dộng (mobile web app) cũng được nhiều nhà phát triển lựa chọn cho các giải pháp của mình. Các ứng dụng dạng này dựa trên sức mạnh của nền tảng web, các nhà phát triển chỉ cần am hiểu kỹ năng phát triển web là có thể triển khai các ứng dụng web app. Ứng dụng web di động chạy trên trình duyệt web và có thể được tìm thấy, chia sẻ thông qua môi trường web. Nhờ vậy, chi phí phát triển ứng dụng web app thường thấp hơn so với việc phát triển một native app, mà việc bảo trì lại dễ dàng, đây là là một lợi thế đáng kể của ứng dụng dạng này. Tuy nhiên, ứng dụng web di động thường không mang lại các giao diện mượt mà như ứng dụng native, đặc biệt là những ứng dụng có yêu cầu đồ họa cao.

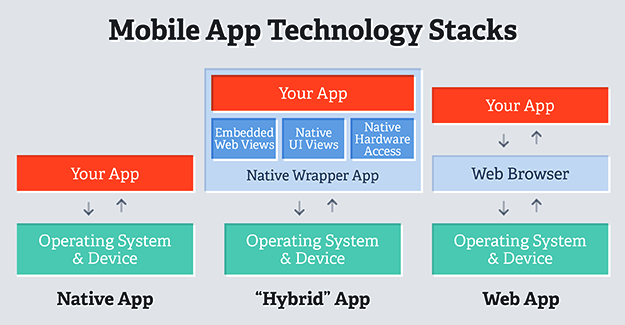
Một hướng phát triển ứng dụng di động đang là xu thế hiện nay, bao gồm ưu diểm của cả native app và web app, đó chính là ứng dụng lại (hybrid app). Hybrid app cũng không yêu cầu nhà phát triển cần có quá nhiều kỹ năng chuyên sâu hay đa nền tảng, mà chỉ cần có các kỹ năng của một người phát triển web. Một ứng dụng dạng hybrid app có thể hỗ trợ đa nền tảng mà không cần thiết phải viết từng ứng dụng cho mỗi nền tảng như native app, do đó rút ngắn thời gian thương mại hóa ứng dụng, đặc biệt là thương mại hóa ứng dụng đồng thời trên các chợ ứng dụng của các nền tảng phổ biến như Google Play cho hệ điều hành Android hay App Store cho các thiết bị của Apple iOS. Các ứng dụng hybrid có thể hỗ trợ các tính năng giống như một native app, đem đến trải nghiệm giống như thao tác với một native app cho các ứng dụng yêu cầu đồ họa tương đối cao. Tuy nhiên, các ứng dụng đòi hỏi đồ họa rất cao như các game, thì hybrid app vẫn chưa thực sự tốt như một native app.

* + 1. Cách thức hoạt động chung của ứng dụng di động

Một ứng dụng di động phụ thuộc vào việc ứng dụng đó được phát triển theo hướng nào mà có cách thức vận hành, tương tác với các thành phần hệ thống khác nhau. Hình 1.2 minh họa cơ chế tương tác của các ứng dụng di động với thiết bị người dùng.

Ứng dụng native app được cài trực tiếp trên thiết bị người dùng và tương tác trực tiếp với hệ điều hành cũng như các thành phần cấu thành của thiết bị, như là các sensor hay các driver khác.

Trong khi đó, ứng dụng hybrid app không trực tiếp tương tác với hệ điều hành và thiết bị di động, mà nó được bao gói trong một native wrapper app. Native wrapper app này cung cấp các cầu nối để ứng dụng hybrid tương tác gián tiếp với hệ điều hành. Các thành phần cầu nối bao gồm có web views, các native UI và các module hỗ trợ truy nhập các phần cứng native. Nhờ có native wrapper app này mà ứng dụng hybrid app đem lại trải nghiệm gần như một native app cho người dùng.



Hình 1. Cách thức hoạt động của ứng dụng di động

Nếu ứng dụng di động được phát triển như là một web app, thì ứng dụng này tương tác trực tiếp với trình duyệt web của thiết bị, điều này đem tới khả năng tương tác đa nền tảng, nhưng lại hạn chế về đáp ứng các giao diện đồ họa phức tạp.

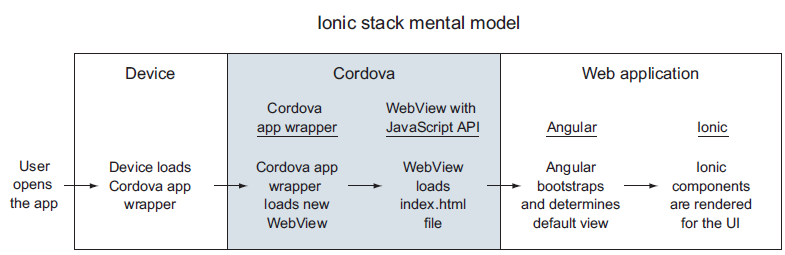
* 1. Ionic Framework
     1. Tổng quan về nền tảng Ionic

Ionic là tổ hợp các công nghệ và tiện ích được thiết kế để phát triển các hybrid app cho thiết bị di động nhanh chóng, giao diện thân thiện và không dòi hỏi quá nhiều kỹ năng phức tạp.

Ionic là một bộ công cụ mã nguồn mở, cho phép phát triển các ứng dụng hybrid cho di động bằng ngôn ngữ HTML5. Nó cũng cung cấp các thành phần tối ưu HTML, CSS và JavaScript cho thiết bị di động, cũng như là các thao tác và công cụ để xây dụng các ứng dụng có độ tương tác cao.

Ionic được xây dựng trên một hệ sinh thái mà bao gồm Angular như là một nền tảng ứng dụng web và Cordova cho việc xây dựng và đóng gói native app.

Hình 1.3. minh họa quá trình làm việc của Ionic



Hình 1. Mô hình cảm quan Ionic

Thiết bị: nạp ứng dụng. Thiết bị gồm hệ điều hành quản lý các ứng dụng được cài đặt từ các chợ ứng dụng. Hệ điều hành cũng cung cấp một tập các API cho các ứng dụng để có thể sử dụng các dịch vụ khác do thiết bị cung cấp như là dịch vụ định vị GPS, danh bạ điện thoại hay các thiết bị cảm biến khác.

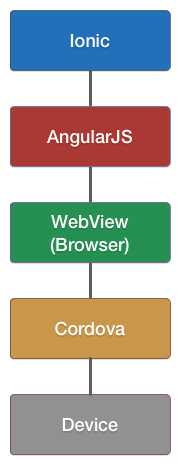
Cordova app wrapper: đây là một ứng dụng native nạp các mã nguồn ứng dụng web. Cordova là một platform cho việc xây dụng các ưng dụng di động sử dụng HTML, CSS và JavaScript bên trong một ứng dụng native, hay chính là các ứng dụng hybrid app. Cordova chính là một tiện ích cầu nối giữa ứng dụng và platform. Nó có thể cho phép nhà phát triển tạo ra các ứng dụng di động cài đặt trực tiếp trên thiết bị của người dùng, và bao gồm Webview với các JavaScript API giúp thực thi các ứng dụng web.

Angular: đây là một ứng dụng web điều khiển các chức năng và điều hướng ứng dụng. Angular chạy bên trong một webview. Nó là một framework phổ dụng cho việc xây dựng các ứng dụng web mạnh mẽ.

Ionic cung cấp các thành phần giao diện người dùng được sinh ra trong ứng dụng. Ionic được phát triển trên nền của Angular, và được sử dụng chính để thiết kế UI và UX. Nó bao gồm các thành phần cơ bản như các tab, các button, và các thanh điều hướng (navigation header). Các giao diện điều khiển này là các thành phần quan trọng nhất của Ionic, cung cấp một giao diện gần như giao diện native bên trong một ứng dụng hybrid. Ionic cũng bao gồm một số các tiện ích và tính năng bổ sung khác cho phép quản lý ứng dụng từ việc tạo và xem trước ứng dụng cho đến phát triển ứng dụng hoàn chỉnh.

* + 1. Kiến trúc của Ionic

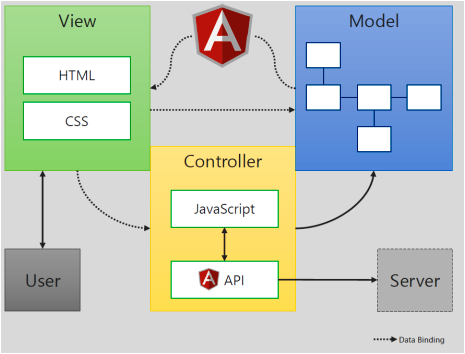
Ionic được xây dựng trên nền của Angular và Cordova framwork. Sự kết hợp của các công nghệ này khiến cho Ionic là một nền tảng đa tính năng mà mạnh mẽ cho việc phát triển các ứng dụng di động. Kiến trúc chung của Ionic được minh họa trong hình 1.4.



Hình 1. Kiến trúc Ionic

* + - 1. Angular

Angular hay còn gọi là AngularJS là một dự án mã nguồn mở của Google đã trở nên khá phổ biến với các nhà phát triển ứng dụng web. Nó cung cấp cho các nhà phát triển một cấu trúc rõ ràng và có khả năng phát triển các ứng dụng web nhanh chóng. Angular được phát triển theo MV\* model, cách thức hoạt động của Angular được minh họa trong hình 1.5. Controller là các function mà được đi kèm một DOM node và được sử dụng để điều khiển cho xử lý logic của ứng dụng. Một controller là một function object quan trọng trong JavaScript để giao tiếp với Angular scope và phản hồi các event. Angular scope giống như context được dùng chung bởi controller và view. Nói cách khác, scope chính là kết nối giữa các sự kiện xảy ra ở controller và giao diện, khi scope được cập nhật bên trong controller thì nó cũng cập nhật ở view.



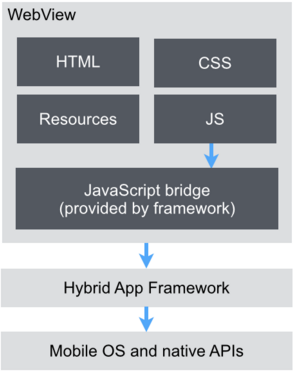
Hình 1. Angular’s operation

Angular được sử dụng như một JavaScript framework cho Ionic.

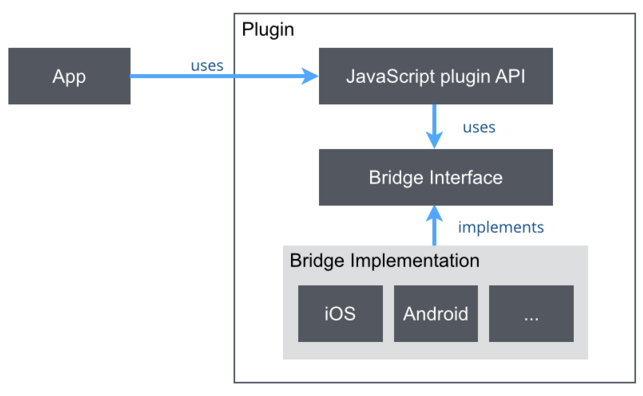
* + - 1. Cordova

Cordova là một hybrid app framework, là một layer có nhiệm vụ quản lý các giao tiếp giữa trình duyệt web và các native API. Cordova là một dự án mã nguồn mở Apache với một cộng đồng phát triển rộng lớn. Phần lõi của Cordova cung cấp rất nhiều feature, nó cũng cung cấp một hệ thống plugin cho các nhà phát triển để có thể tạo ra nhiều tính năng mới, ví dụ như tích hợp các native API với phone camera. Nó được bảo trì một cách tích cực với các bản cập nhật mới được định kỳ đưa tới cộng đồng, bổ sung thêm các tính năng mới hoặc cài thiện để năng cao hiệu năng các tính năng sẵn có. Hình 1.6 minh họa kiến trúc của Cordova. Cordova là một platform cho ứng dụng hybrid dựa trên web. Nó thực hiện hỗ trợ và tương tác với các API của các thiết bị khác nhau và kết hợp những ưu diểm của tất cả các native API trong một JavaScript API mà có thể được truy nhập bởi ứng dụng hybrid. Ứng dụng hybrid app chạy bên trọn một webview mà được kiểm soát bởi Cordova framework.

Bên cạnh đó, Cordova có khái niệm về các plugin mà gói gọn sự tương tác với một (đôi khi là nhiều) native API của thiết bị. Các plugin này được sử dụng cho việc truy nhập các thông tin cơ bản của thiết bị, truy nhập các file cục bộ và truy nhập định vị địa lý của thiết bị là các ví dụ về plugin của hệ sinh thái Cordova. Các plugin luôn luôn bao gồm hai phần. Phần thứ nhất là một bộ phận JavaScript chạy bên trong một webview mà có một API dễ dàng cho ứng dụng hybrid. Phần thứ hai là đặc trưng cho từng nền tảng và được viết bằng ngôn ngữ native tương ứng, ví dụ Java cho Android và Objective-C cho iOS. Các native API được kiểm soát bởi phần thứ hai. Hình 1.7 minh họa cho kiến trúc của plugin.



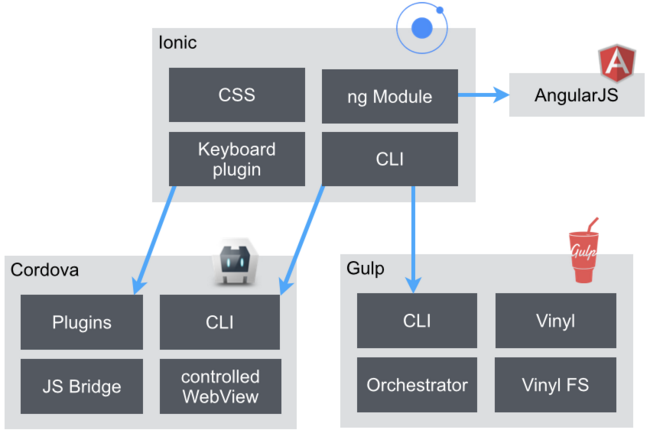
Hình 1. Kiến trúc Cordova



Hình 1. Kiến trúc Plugin Cordova

* + - 1. Ionic

Ionic được xây dựng trên Angular JS. Ionic về cơ bản là một Angular JS module và một stylesheet mà có thể được sử dụng để xây dựng các ứng dụng giống với ứng dụng native. Do đó nó rất dễ để thao tác với Ionic, đặc biệt là với những nhà phát triển đã quen thuộc với Angular JS. Hầu hết các thành phần cấu thành của Ionic, ví dụ như list, toggle hay slide box, đều là các Angular JS directive. Mọi nhà phát triển Angular JS đều có thể đọc và debug mã nguồn Ionic vì nó sử dụng chung concept và cấu trúc như chính ứng dụng của Angular JS. Điều này khiến cho mọi nhà phát triển Angular JS có thể trace bug và có thể trở thành các contributor tiềm năng cho Ionic framework. Hình 1.8 thể hiện cấu trúc của Ionic.



Hình 1. Cấu trúc Ionic

Ionic framework gồm bốn thành phần chính:

* Một stylesheet định nghĩa một layout tối ưu hóa cho thiết bị di động. Layout này được dùng như là thiết lập đầu cho ứng dụng.
* Một AngularJS module có thể dùng chính AngularJS. Nó định nghĩa các directive (có thể hiểu là các thành phần tùy chỉnh), các navigation pattern và các template sẵn có do cộng đồng phát triển và đóng góp.
* Command line tool (CLI) giúp nhà phát triển có thể thao tác nhanh chóng và có thể tạo ra hàng loạt proxy cho Cordova và Gulp CLI.
* Thành phần quan trọng cuối cùng là một keyboard plugin.
  + 1. Các đặc điểm của Ionic
       1. Ưu điểm của Ionic

Tuy là một dự án mới, nhưng Ionic là một dự án mã nguồn mở có một cộng đồng nhà phát triển rộng lớn và rất tích cực, đây cũng chính là một trong những ưu điểm quan trọng của Ionic.

Ionic cho phép xây dựng các ứng dụng với nền tảng web sử dụng HTML, CSS và JavaScript, các nhà phát triển web có thể nhanh chóng xây dựng các ứng dụng hybrid với trải nghiệm người dùng gần như ứng dụng native.

Với công cụ CLI mạnh mẽ, Ionic cho phép nhà phát triển nhanh chóng quản lý các task trong quá trình phát triển ứng dụng như xem trước ứng dụng trên trình duyệt web, mô phỏng ứng dụng hay cài đặt ứng dụng lên một thiết bị được kết nối. Nó cũng cho phép khởi tạo và bắt đầu một project.

Các dịch vụ Ionic thân thiện. Ionic cung cấp nhiều dịch vụ giúp cho việc phát triển trở nên dễ dàng hơn:

Ionic Push: đây là dịch vụ push thông báo được cung cấp bởi Ionic. Ionic push giúp nhà phát triển tạo ra các push notification thông qua một dashboard đơn giản mà có thể gửi tự động khi người dùng kết hợp với các tiêu chí cụ thể, và cung cấp một API đơn giản để gửi push notification từ chính server của nhà phát triển.

Ionic Deploy: Ionic Deploy giúp nhà phát triển cập nhật ứng dụng theo yêu cầu, cho mọi thay đổi mà không yêu cầu binary modification.

Ionic Analytic: telemetry service được cung cấp bởi Ionic, khá giống với Google Analytics.

Ionic Package: tương tự với package trong Cordova, nhà phát triển có thể đóng gói nó để sử dụng trong bất kỳ platform nào được hỗ trợ.

Ionic View: Ionic View làm cho việc test ứng dụng Ionic trên nhiều thiết bị một cách dễ dàng mà không cần phải xây dựng và phân phối ứng dụng.

Ionic Creator: cho phép việc khởi tạo giao diện người dùng thuận tiện và dễ dàng, nhà phát triển ứng dụng có thể tạo ra các giao diện người dùng online rồi sau đó tải vào thư mục www và điều chỉnh offline.

Thiết kế đẹp và linh hoạt với hiệu năng cao: với Ionic, nhà phát triển có thể tạo ra ứng dụng với giao diện và trải nghiệm người dùng gần giống với ứng dụng native.

Một ưu điểm quan trọng khác của Ionic đó chính là khả năng viết một mã nguồn mà ứng dụng có thể chạy trên nhiều nền tảng.

* + - 1. Nhược điểm của Ionic

Ionic nói riêng, hay phát triển ứng dụng hybrid, không phải lúc nào cũng là lựa chọn tối ưu để phát triển ứng dụng. Ionic có một số hạn chế như sau:

Hạn chế nền tảng: Ionic mới chỉ hỗ trợ hoàn toàn hai nền tảng là Android và iOS. Các nền tảng khác như Windows Phone hay Firefox OS có thể được hỗ trợ trong tương lai, nhưng điều này chưa chắc chắn. Các ứng dụng có thể vẫn chạy được trên các nền tảng khác, tuy nhiên Ionic không chính thức hỗ trợ chúng.

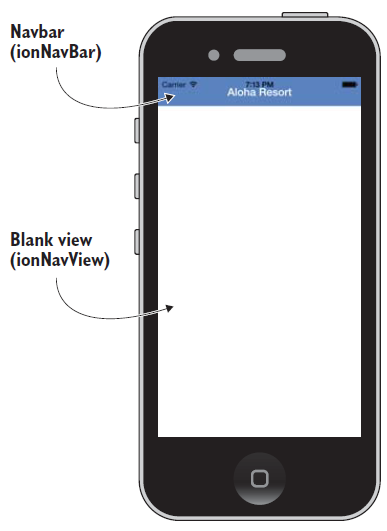
Các nền tảng cũ không được hỗ trợ. Ionic hỗ trợ iOS 7+ và Android 4+. Các phiên bản cũ hơn có thể làm việc nhưng điều này chưa được kiểm chứng. Điều này chính là thách thức cho các nhà phát triển hướng tới thị trường thiết bị tầm trung và tầm thấp.

Không bằng native: các native API chỉ được cung cấp nếu Cordova hỗ trợ chúng. Nếu nhà phát triển cần tích hợp sâu vào thiết bị thì đây có thể là một mục tiêu khó thực hiện được.

Không hiệu quả với các ứng dụng nặng đồ họa. Điều này là hạn chế chung của các ứng dụng hybrid vì chúng chạy trong trình duyệt web. Nếu mục tiêu phát triển là một game hay ứng dụng yêu cầu cao về đồ họa thì ứng dụng hybrid không phải là lựa chọn tốt như là ứng dụng native.

* + 1. Các thành phần chính của Ionic
       1. Ionic Navigation

ionNavView và ionNavBar là các thành phần cơ bản của Ionic cho việc điều hướng ứng dụng. ionNavView hoạt động giống như một nơi lưu trữ cho các thay đổi từ view content để tải lên ứng dụng. Còn ionNavBar hiển thị thanh tiêu đề mà sẽ cập nhật khi người dùng di chuyển giữa các view. Hình 1.9 minh họa hai thành phần này.



Hình 1. Ionic Navigation

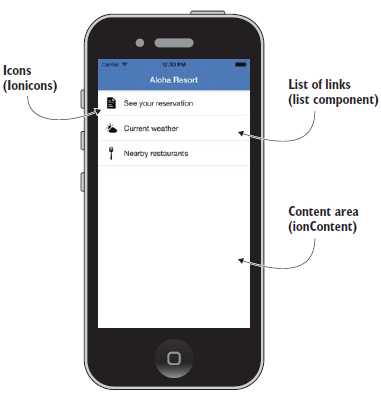
ionNavView cho phép nội dung được tải bên trong nó nhưng nó không là một frame thực sự, mà đó là do Angular có cung cấp một số template với khả năng inject thêm một số markup mớ vào bên trong view (trong trường hợp này là ionNavView). Nếu không có ionNavView, ứng dụng sẽ không thể biết vị trí nào sẽ dùng để nạp nội dung, chính vì vậy, khi ứng dụng có sử dụng navigation, thì cần thiết phải có ít nhất một ionNavView.

ionNavBar sẽ thay thế thanh top bar của ứng dụng. Nó giống như một thanh tiêu đề, hỗ trợ việc di chuyển qua lại giữa các view. Điều này khá giống với một ứng dụng native với nút back hoặc forward.

* + - 1. Content container

ionContent là thành phần được sử dụng nhiều nhất cho content container. Nó cung cấp các chức năng như sau:

* Thay đổi kích thước vùng hiển thị nội dung tùy theo thiết bị.
* Biết thanh header và thanh footer
* Quản lý thao tác cuộn lên xuống



Hình 1. Ionic content container

* + - 1. IonTab

Tab được sử dụng rất nhiều trong ứng dụng di động, và Ionic cung cấp một thành phần giàu tính năng cho phép nhà phát triển dễ dàng và nhanh chóng tạo ra các tab. Tab thường được sử dụng để hiện thị một cách trực quan sự liên kết giữa các view. Mặc dù không có giới hạn nào về số lượng tab được sử dụng trong khi phát triển ứng dụng, nhưng chỉ nên sử dụng từ 2 đến 5 tab vì màn hình các thiết bị di động thường giới hạn. Các tab có thể được sử dụng gần như mọi nơi trong ứng dụng, ngoại trừ bên trong ionContent directive bởi vì sẽ xảy ra CSS collision.

Ionic cung cấp hai thành phần hỗ trợ xây dựng các tab là ionTabs và ionTab. ionTabs sẽ được khai báo trước và bên trong nó có thể thêm vào nhiều ionTab. Các tab có thể có icon, tiêu đề hoặc cả hai.

* + - 1. Side menu

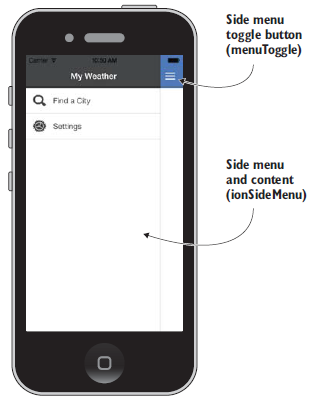
Side menu thường được sử dụng bởi vì nó có thể trượt vào hoặc ra khỏi view theo yêu cầu, cho phép người dùng truy xuất nhanh vào các liên kết chức năng khác mà không gián đoạn nội dung chính. Một side menu có thể được mở từ phải hoặc trái tại một thời diểm.

Một side menu có thể được mở bằng ba cách, phụ thuộc vào việc thực thi. Theo mặc định thì Ionic hỗ trợ việc mở side menu bằng cách vuốt vào cạnh để mở slide menu. Nhà phát triển có thể vô hiệu hóa chức năng này để sử dụng thao tác vuốt cho mục đích khác và có thể thêm một button để mở side menu, thường thì button này được đặt ở trên cùng bên trái của màn hình. Và cuối cùng, nhà phát triển cũng có thể tạo một toggle menu sử dụng sidebar delegate service.

Hình 1.11 minh họa việc thiết kế Side Menu sử dụng toggle. Side menu dễ khai báo, bởi vì nó chỉ yêu cầu sử dụn ionSideMenus, ionSideMenuContent và ionSideMenu directives mà không cần sử dụng JavaScript. Để sử dụng được, trước tiên phải khai báo ionSideMenus. Bên trong ionSideMenus gồm hai thành phần con là ionSideMenuContent và ionSideMenu. Mỗi side menu có thể có một ionSideMenuContent nhưng sẽ chỉ có thể có tối đa hai ionSideMenu cho bên trái hoặc/và bên phải.

Bên trong side menu có thể sử dụng một header bar và một content area để bọc danh sách điều hướng nếu không, vùng hiển thị nội dung sẽ không thể tính toán đúng kích thước và định vụ các phần tử.

Side menu cso thể chứa bất kỳ nội dung nào mà người phát triển muốn đưa vào, nhưng một danh sách liên kết điều hướng thường được sử dụng nhiều nhất cho một side menu.



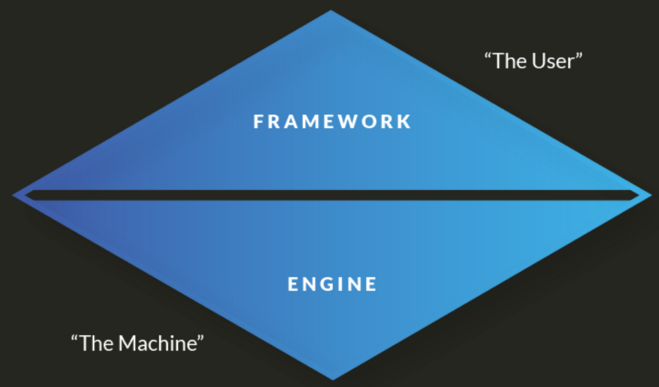
Hình 1. Side menu

* 1. Một số nền tảng tương tự
     1. Famous
        1. Famous Overview

Famous là một framework cho phát triển các ứng dụng web app với cách thức mới trong việc kiểm soát các phần tử HTML, đơn giản hóa việc render DOM trong trình duyệt web. Famous kết hợp DOM tree với WebGL, hiển thị mọi thứ trong một canvas tương tự như những gì mà các game engine HTML thực hiện. Kỹ thuật mới này giúp cho framework có thể chạy ứng dụng với tốc độ 60 khung hình trên giây, khiến người dùng có trải nghiệm ứng dụng mượt mà giống như đang thao tác với ứng dụng native.

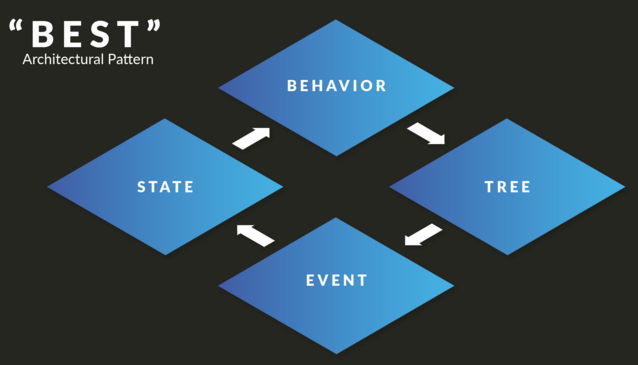
Hình 1.12 minh họa kiến trúc cơ bản của Famous framework.

Lớp Engine (hay còn gọi là “The Machine”) được tập trung vào khả năng và hiệu năng của Famous. Trong khi đó, lớp Framework của Famous (hay còn gọi là “The User”) tận dụng khả năng của Engine nhưng tập trung vào tính tái sử dụng, tính bền vững và khả năng tích hợp với các công nghệ khác.



Hình 1. Kiến trúc tổng quan Famous

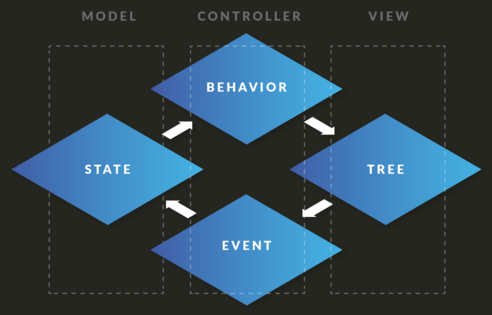
Famous được xây dựng theo kiến trúc Behavior Event State Tree (BEST). BEST là một mô hình kiến trúc mới cho việc tái sử dụng, đan xen các thành phần.



Hình 1. Kiến trúc BEST

Hình 1.13 minh họa mô hình kiến trúc BEST. BEST giống như là một biến thể của mô hình MVC dựa trên event. MVC cung cấp một Model để quản lý trạng thái và tính bền vững, một View để thể hiện trạng thái với người dùng và thu thập đầu vào, và một Controller để kết nối các giao tiếp và luồng logic giữa Model và View.

Trong Famous, Controller chính là điểm chính yếu khiến BEST trở nên khác biệt so với MVC. Trên thực tế, BEST là một biến thể của MVC dựa trên sự kiện mà được chia tách các đầu vào, đầu ra của Controller thành các phần riêng biệt là Behaviours và Events. Hình 1.14 thể hiện sự khác biệt của BEST và MVC.



Hình 1. So sánh BEST và MVC

Tree: là một phân cấp của các thành phần được viết bằng HTML. Có thể hiểu đơn giản, Tree là một cấu trúc xương sống cho các thành phần giao diện người dùng của ứng dụng.

Events: events là nơi tạo ra điều kiện thay đổi cho các thành phần logic của ứng dụng. Ví dụ như, khi nhấn chuột vào một nút, thì biến x tăng giá trị lên 10. Tất cả các thao tác của trạng thái ứng dụng xảy ra bên trọng một xử lý sự kiện, và xử lý sự kiện có thể bắt đầu một event khác. Điều này khác giống với các phương thức trong hệ thống hướng đối tượng, ngoại trừ việc xử lý sự kiện được kích hoạt bởi event chứ không bởi một lời gọi trực tiếp. Các event được gắn với các thành phần của Tree nhờ CSS và JavaScript thông qua các Selector. Selector cho phép các event và tree được bảo trì như các thành phần riêng biệt, tương tự như việc cho phép CSS và HTML cùng được duy trì rời nhau trong phát triển web truyền thống.

State: state là trạng thái thể hiện thông qua các dữ liệu và cài dặt của các thành phần ứng dụng. Khác với mô hình MVC, một trạng thái của BEST được khai báo và logic linh hoạt, các trạng thái của một thành phần có thể được định nghĩa bằng JSON.

Beheviors: behaviors hoàn tất chu trình, kết nối các State và Tree. Behaviors có thể là các function và cũng có thể là hàm của một biến trạng thái nào đó. Khác với các Event, behaviors được gắn vào các thành phần của Tree sử dụng các Selector, mặc dù các behaviors giống CSS hơn các Event, và thực sự behaviors được thể hiện bằng các function JavaScript.

* + - 1. So sánh Famous và Ionic

Famous framework khá tương đồng Ionic trong việc hướng tới phát triển các ứng dụng hybrid dựa trên HTML, CSS và JavaScript. Tuy nhiên, mô hình mà Famous sử dụng là BEST, một biến thể của mô hình MVC, và nó có khả năng cho phép tạo ứng dụng hỗ trợ giao diện mượt mà (tới 60 fps) gần như một ứng dụng native; đây là điểm đặc biệt ở một nền tảng phát triển ứng dụng hybrid.

Mặc dù vậy, Famous là một nền tảng mới với cộng đồng chưa lớn mạnh như cộng đồng Ionic, và các tài liệu giới thiệu và mô tả của Famous cũng là một điểm yếu so với Ionic.

* + 1. OnsenUI
       1. OnsenUI overview

Là mã nguồn mở, miễn phí. Được thiết kế và thực thi để cung cấp giao diện người dùng cho các ứng dụng mobile và hybrid apps, nó sử dụng thẻ HTML5 để phát triển web. Onsen UI là giải pháp hoàn hảo cho phát triển hybrid apps bởi vì chúng có hiệu năng tốt và dễ dùng. OnsenUI có thể được sử dụng ở bất kỳ framework cuối nào mà bạn muốn, tuy nhiên OnsenUI cũng cung cấp một thư viện cho Angular JS cái giúp nó dễ dàng tích hợp với Angular JS apps. Onsen UI cũng cung cấp thành phần CSS, một tài nguyên miễn phí của giao diện mẫu có sẵn. Nhà phát triển có thể chọn cái họ cần rồi ghép và chạy, họ cũng có thể tạo mẫu của họ và đưa lên thành templates có sẵn.

Onsen UI cho phép điều chỉnh view dựa trên kích cỡ màn hình và nó cũng dễ để sử dụng với thẻ <ons-split-view> để có hai cột hiển thị trên máy tính bảng và một cột cho smartphone.

* + - 1. OnsenUI và Ionic

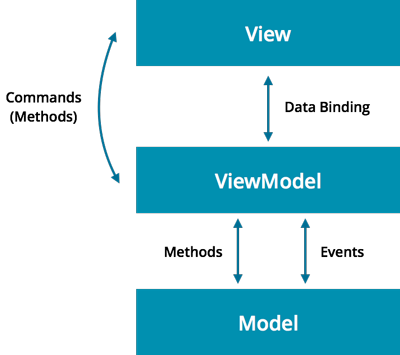
Ionic hỗ trợ iPhone(6+) và android devices (4.1 up), nó chủ yếu được làm cho native/hybrid mobile apps hơn là mobile websites

Onsen UI hỗ trợ Android 2.3 trở lên, ios 6 trở lên, Firefox OS, Chrome and Safari. Nó hỗ trợ hiển thị 2 cột ở tablet và 1 cột ở smartphone. Onsen UI cũng hỗ trợ Windows Mobile 8.1

OnsenUI không có host để lưu các template bởi vậy các lập trình viên không thể viết report về các vấn đề họ gặp phải để trao đổi với các đồng nghiệp khác. Việc cập nhật cũng khó khăn. Ionic đã làm rất tốt vấn đề này.

* + 1. Sencha Touch
       1. Sencha Touch overview

Sencha touch là một framework được viết bằng ngôn ngữ Javascript dùng để viết các ứng dụng cho điện thoại di động. Đây là 1 frame work mạnh được viết theo mô hình MVC, two-way data binding hoặc MVVM. Nó có sẵn các control, event dùng để điều khiển và xây dưng giao diện, chúng ta chỉ việc gọi ra và xử lý các sự kiện trên đó.



Hình 1. Mô hình của Sencha Touch

MVVM cho phép thay đổi view một các tự động, nó đồng bộ giữa view và model (dùng two-way data binding).

Một ứng dụng xây dựng sử dụng Sencha Touch có thể được truy cập từ một trình duyệt hoặc được đóng gói cho các thiết bị touch và chạy trên chúng.

Khi viết các ứng dụng với Sencha Touch thì các bạn phải tạo các Service cung cấp dữ liệu cho nó bởi vì bản thân nó là JS nên không thể trực tiếp lên server để lấy thông tin được, các service này có thể được viết bằng ngôn ngữ PHP hoặc một ngôn ngữ khác.

Sencha Touch 2.2 hỗ trợ WebKid, trình duyệt IE 10 trên Windows 8 với các flatforms Anroid, iOS, BlackBerry.

* + - 1. Sencha Touch và Ionic

Sencha Touch là một framework mạnh và khó. Với một lập trình viên chưa có nhiều kinh nghiệm thì Ionic là một lựa chọn tốt hơn.

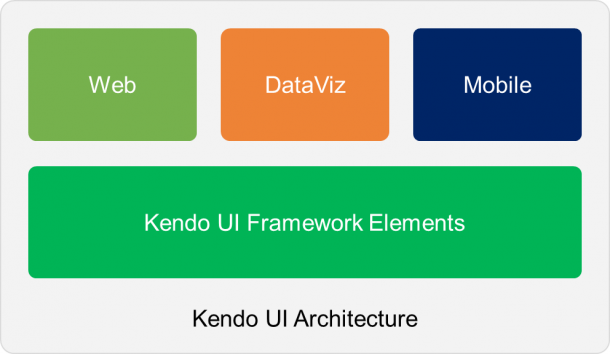
* + 1. Kendo UI
       1. Kendo UI overview

Là một framework dựa trên nền tảng HTML5 và jQuery, hỗ trợ toàn diện trong việc xây dựng các ứng dụng Web hiện đại một cách dễ dàng, linh hoạt. Kendo UI hỗ trợ mạnh mẽ cho cả ứng dụng trên Web và trên Mobile. Ngoài ra nó còn rất mạnh trong việc trực quan hóa dữ liệu (Data Visualization) giúp ứng dụng Web trông như ứng dụng thuần Mobile hay thuần Desktop.

Kendo UI widgets sử dụng HTML5, CSS và jQuery để xây dựng các ứng dụng web một các mạnh mẽ mà không cần sử dụng framework thứ ba hay plugins.

Kendo UI gồm 3 phần chính:

* Kendo UI Web: hỗ trợ xây dựng một ứng dụng Web trên desktop
* Kendo UI Mobile: hỗ trợ xây dựng ứng dụng Web trên Mobile
* Kendo UI DataViz: hỗ trợ thiết lập và tương tác dữ liệu trực quan



Hình 1. Kiến trúc của Kendo UI

Ngoài ra, Kendo UI còn có cả tính năng Kendo UI-server Sapper độc đáo giúp chúng ta xây dựng 1 ứng dụng HTML5 mà không cần phải viết code JavaScript. Để làm được điều đó Kendo UI phát triển một số thư viện dưới dạng các helpers trên Server, Kendo UI-Server Swapper sẽ tiếp nhận các câu lệnh đơn giản của lập trình viên trong mã nguồn các ngôn ngữ lập trình ASP, PHP, JSP và sau đó sẽ tự phát sinh (render) các mã HTML và JavaScript cần thiết.

* + - 1. Kendo UI và Ionic

Kendo UI tạo nhiều sự điều chỉnh đặc dựa trên platform mà bạn đang sử dụng (fonts, CSS,…) vậy nên nó mềm dẻo hơn Ionic.

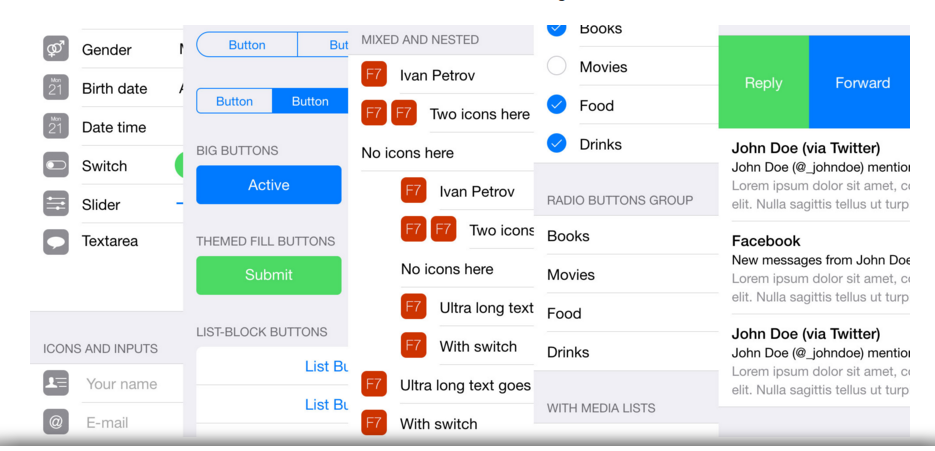
* + 1. F7 Framework
       1. F7 framework overview

Framework này cho phép phát triển các web app, hybrid app bằng cách sử dụng HTML, CSS, JavaScript.

Framework7 không tương thích hết với tất cả các flatform, chủ yếu dùng cho iOS và Android. Bạn bắt buộc phải sử dụng F7-MVC framework để phát triển được các ứng dụng di động.

Framework 7 đặc biệt phù hợp với các app iOS. Để tạo một iOS apps sử dụng framework7 bạn cần tạo một layout đơn giản với HTML và gắn thêm CSS và JS file. Framework7 không bắt buộc phải mô tả các nội dung của bạn trong JavaScript mà chỉ cần dùng HTML.

Framework7 có sẵn rất nhiều các thành phần UI và widget như modals, popup, action sheet, popover, list views, media lists, tabs, side panels, layout grid, preloader, …



Hình 1. Các UI elements của Framework7

Framework7 có nhiều đặc tính “killer” như Swipe back, Swipe Actions, Dynamic Navbar, Full to refresh, Messages giúp các ứng dụng được xây dựng trông như một native app. Swipe back là swipe từ bên trái (hoặc kéo chuột xuống) của màn hình để gọi lại màn hình trước đó. Swipe actions giúp lướt từ trái qua phải hoặc ngược lại để chọn thêm các thao tác khác. …

Còn rất nhiều đặc tính tiện ích khác của Framework7 cho thấy đây là một framework vô cùng tiện dùng.

* + - 1. F7 Framework và Ionic

F7 framework phù hợp nhất cho phát triển app trên iOS, ionic có thể dùng cho cả iOS, android và các OS khác một cách mượt mà hơn. Mặc dù F7 tiện dùng nhưng lập trình viên cần hiểu về F7 MVC framework để có thể phát triển được phần mềm.

Chương 2. Phân tích thiết kế phần mềm

1. Phân tích và đặc tả yêu cầu
   1. Giới thiệu

### 2.1.1. MỤC ĐÍCH

* Tên ứng dụng: UET Todo List
* Mục đích của ứng dụng: Nhắc người dùng các công việc cần làm hằng ngày
* Ứng dụng Todo List là ứng dụng cho phép người dùng thêm tạo danh sách các công việc phải làm. Dựa trên mức độ hoàn thành các công việc trong thực tế, người dùng có thể chỉnh sửa, cập nhật danh sách công việc đã lưu trên ứng dụng.

Ứng dụng này có khả năng nhắc nhở, gợi nhớ người dùng về các công việc cần làm và phải làm, từ đó tăng hiệu quả làm việc của người dùng.

### *2.1.2. PHẠM VI*

Trong giới hạn của một bài tập nhóm môn Các vấn đề hiện đại của Công nghệ phần mềm dành cho chương trình đào tạo Thạc sỹ Công nghệ thông tin trường Đại học Công nghệ - Đại học Quốc Gia Hà Nội, ứng dụng cần đáp ứng các yêu cầu sau:

* Được xây dựng bằng Ionic Framework
* Có sử dụng các mẫu có sẵn của Ionic
* Có sử dụng các plugin có sẵn của Ionic
* Đáp ứng được nhu cầu sử dụng đối với một ứng dụng Todo List
* Có không quá 03 màn hình

### 2.1.3. NGOÀI PHẠM VI

Các công việc không nằm trong phạm vi của việc xây dựng ứng dụng này

* Đưa ứng dụng lên các kho ứng dụng
* Thêm các tính năng phát sinh so với thống nhất ban đầu
* Cung cấp các tài liệu hướng dẫn sử dụng
* Đồng bộ danh sách các công việc với tài khoản Google Calendar
  1. Mô tả chung

### 2.2.1. TỔNG QUAN VỀ SẢN PHẨM

Ứng dụng Todo List là ứng dụng cho phép người dùng thêm tạo danh sách các công việc phải làm. Dựa trên mức độ hoàn thành các công việc trong thực tế, người dùng có thể chỉnh sửa, cập nhật danh sách công việc đã lưu trên ứng dụng.

Ứng dụng này có khả năng nhắc nhở, gợi nhớ người dùng về các công việc cần làm và phải làm, từ đó tăng hiệu quả làm việc của người dùng.

### 2.2.2. CHỨC NĂNG SẢN PHẨM

Để đạt được mục đích của ứng dụng, sản phẩm cần có các chức năng như Bảng 1.

Bảng 2. Chức năng sản phẩm

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên chức năng** | **Mô tả** |
| 1 | Thêm công việc | Người dùng có thể thêm công việc vào danh sách |
| 2 | Loại bỏ công việc | Khi đã hoàn thành công việc, người dùng có thể loại bỏ công việc khỏi danh sách |
| 3 | Cài đặt | Chức năng này cho phép người dùng tùy biến một số thiết đặt của chương trình như: Đồng bộ, Thông báo |
| 4 | Giới thiệu | Cung cấp cho người dùng thông tin về sản phẩm |

### 2.2.3. ĐỐI TƯỢNG NGƯỜI DÙNG

Sản phẩm hướng tới đối tượng người dùng:

* Độ tuổi: 16 – 50
* Sử dụng smartphone nền tảng: Android, iOS, Window Phone và Black Berry
* Sử dụng tiếng Anh
* Khu vực: Toàn thế giới

### 2.2.4. CÁC RÀNG BUỘC PHÁT TRIỂN

* Về nguồn lực: 6 manday
* Về thời gian: 6 tuần
* Về chất lượng: Đáp ứng được yêu cầu của một ứng dụng trình diễn công nghệ

### 2.2.5. GIẢ THIẾT

* Các thành viên trong nhóm có thể dành 2 tiếng một tuần để xây dựng ứng dụng
* Các yêu cầu ứng dụng không thay đổi
* Các plugin của ionic hoạt động đúng
* Các bản cập nhật của hệ điều hành không làm ảnh hưởng đến sự vận hành của ứng dụng
* Nhóm trang bị đủ các cơ sở vật chất cần thiết cho các tính năng
  1. Yêu cầu chi tiết
     1. Yêu cầu chức năng

Các chức năng ở đây được sắp xếp theo thứ tự **M**o**SC**o**W**:

M – Must Have (Bắt buộc)

S – Should Have (Nên có)

C – Could Have (Có thể có)

W – Would like to have (Mong muốn)

THÊM CÔNG VIỆC

Sử dụng để thêm một công việc vào danh sách

C:\Users\Wolf\Desktop\Untitled Diagram.png

Danh sách các chức năng nhỏ

| STT | Tên các chức năng nhỏ | Mức ưu tiên  (MoSCoW) | Ghi chú |
| --- | --- | --- | --- |
| 1.1 | Cho phép người dùng nhập từ bàn phím nội dung của một công việc. | M |  |
| 1.2 | Người dùng có thể nhập (import) danh sách các công việc từ file đã được xuất (export) từ trước | W |  |

**Trường hợp sử dụng: Thêm công việc từ bàn phím**

* Tác nhân: Người sử dụng
* Mục đích: Thêm một công việc vào danh sách công việc hiện có từ bàn phím
* Mô tả: Sau khi ấn vào nút thêm công việc, người dùng nhập nội dung của công việc cần thêm và ghi lại.
* Luồng sự kiện chính:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn nút thêm công việc | Hiển thị khung nhập liệu |
| Người dùng nhập nội dung công việc và lưu | Hệ thống lưu công việc mới vào cơ sở dữ liệu và hiển thị danh sách mới cho người dùng |

* Luồng sự kiện phụ:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn nút thêm công việc | Hiển thị khung nhập liệu |
| Người dùng ấn nút hủy, không nhập công việc nữa | Ứng dụng trở về cửa sổ trước đó |

* Ngoại lệ: Nếu người dùng chọn thêm công việc nhưng để nội dung trống thì hệ thống không phản hồi.

**Trường hợp sử dụng: Nhập từ tệp dữ liệu**

* Tác nhân: Người sử dụng
* Mục đích: Thêm một công việc vào danh sách công việc hiện có từ tệp dữ liệu
* Mô tả: Sau khi chọn tính năng thêm công việc từ tệp dữ liệu, người dùng chọn tệp cần nhập, hệ thống bổ sung các công việc từ tệp.
* Luồng sự kiện chính:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn tính năng nhập công việc từ tệp dữ liệu | Gọi tính năng chọn tệp của hệ điều hành |
| Người dùng chọn tệp cần nhập liệu | Hệ thống bổ sung các công việc từ tệp |

* Luồng sự kiện phụ:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn tính năng nhập công việc từ tệp dữ liệu | Gọi tính năng chọn tệp của hệ điều hành |
| Người dùng ấn nút hủy, không nhập công việc nữa | Ứng dụng trở về cửa sổ trước đó |

* Ngoại lệ: Nếu người dùng chọn tệp không đúng định dạng, hệ thống trở về cửa sổ trước đó

#### LOẠI BỎ CÔNG VIỆC

Sử dụng để loại bỏ một công việc khỏi danh sách

C:\Users\Wolf\Desktop\Untitled Diagram.png

Danh sách các chức năng nhỏ

| STT | Tên các chức năng nhỏ | Mức ưu tiên  (MoSCoW) | Ghi chú |
| --- | --- | --- | --- |
| 2.1 | Cho phép người dùng loại bỏ từng công việc | M |  |
| 2.2 | Người dùng có thể xóa toàn bộ công việc trong danh sách | W |  |

**Trường hợp sử dụng: Loại bỏ từng công việc**

* Tác nhân: Người sử dụng
* Mục đích: Cho phép người dùng loại bỏ từng công việc
* Mô tả: Khi hoàn thành một công việc, người dùng có thể ấn vào công việc đó trong danh sách các công việc cần thực hiện, hệ thống sẽ xóa công việc đó khỏi danh sách các công việc.
* Luồng sự kiện chính:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng ấn vào công việc đã hoàn thành | Hệ thống thống hiện hội thoại để xác nhận việc xóa công việc của người dùng |
| Người dùng chọn đồng ý xóa công việc | Hệ thống loại bỏ công việc đó khỏi cơ sở dữ liệu và làm mới lại danh sách |

* Luồng sự kiện phụ:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng ấn vào công việc đã hoàn thành | Hệ thống thống hiện hội thoại để xác nhận việc xóa công việc của người dùng |
| Người dùng hủy yêu cầu xóa | Ứng dụng trở về cửa sổ trước đó |

* Ngoại lệ: Không có.

**Trường hợp sử dụng: Xóa toàn bộ danh sách công việc**

* Tác nhân: Người sử dụng
* Mục đích: Người dùng toàn bộ công việc trong danh sách
* Mô tả: Người dùng chọn tính năng xóa toàn bộ công việc trong danh, hệ thống sẽ xóa toàn bộ danh sách công việc
* Luồng sự kiện chính:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn tính năng xóa danh sách | Hệ thống thống hiện hội thoại để xác nhận việc xóa danh sách của người dùng |
| Người dùng chọn đồng ý xóa danh sách | Hệ thống loại bỏ toàn bộ công việc đó khỏi cơ sở dữ liệu và làm mới lại danh sách |

* Luồng sự kiện phụ:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn tính năng xóa danh sách | Hệ thống thống hiện hội thoại để xác nhận việc xóa danh sách của người dùng |
| Người dùng hủy thao tác xóa danh sách | Ứng dụng trở về cửa sổ trước đó |

* Ngoại lệ: Không có.

#### PHẢN HỒI THÔNG BÁO

* Tác nhân: Người sử dụng
* Mục đích: Người sử dụng đọc được thông báo và đưa ra phản hồi
* Mô tả: Theo định kỳ hoặc khi có thông tin quảng bá từ nhà phát triển, ứng dụng đẩy thông báo cho người dùng thông qua thanh thông báo. Người dùng sẽ nhấp vào thông báo để tương tác với ứng dụng
* Tiền điều kiện: Đến thời điểm đẩy thông báo hoặc có thông tin quảng bá từ nhà phát triển
* Luồng sự kiện chính:

* Ngoại lệ: Người dùng bỏ qua thông báo.

#### CÀI ĐẶT

* Tác nhân: Người sử dụng
* Mục đích: Hiện thị trang cài đặt để người dùng thiết đặt tùy chọn
* Mô tả: Đây là tính năng cho phép người dùng bật tắc các lựa chọn của ứng dụng
* Luồng sự kiện chính:

|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn trang cài đặt trên ứng dụng | Hệ thống hiển thị trang cài đặt |
| Người dùng nhấp vào các lựa chọn cụ thể để cài đặt | Hệ thống ghi nhận lựa chọn của người dùng và thể hiện trên giao diện |

* Danh sách các lựa chọn cần có

|  |  |
| --- | --- |
| **Tính năng** | **Lựa chọn** |
| Đồng bộ hóa | - Đồng bộ qua kết nối Wifi  - Đồng bộ qua kết nối Dữ liệu di động |
| Hiện thông báo | - Lựa chọn cho phép ứng dụng đẩy thông báo  - Lựa chọn nhận các thông tin quảng bá từ nhà phát triển |

#### GIỚI THIỆU

* Tác nhân: Người sử dụng
* Mục đích: Hiện thị trang giới thiệu về ứng dụng
* Mô tả: Đây là tính năng cho phép người dùng đọc các thông tin giới thiệu về ứng dụng
* Luồng sự kiện chính:

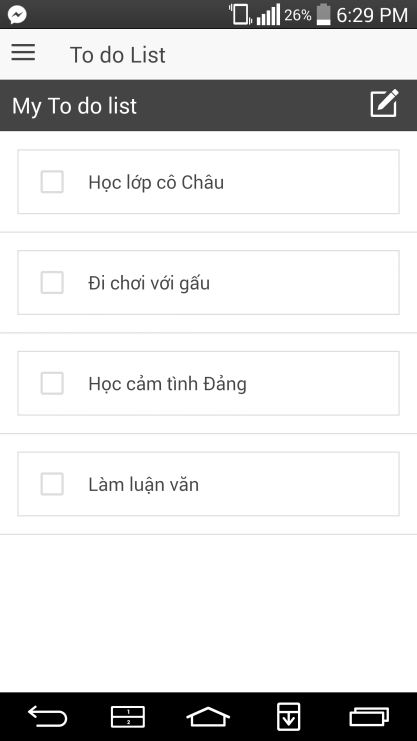
|  |  |
| --- | --- |
| **Hành động của tác nhân** | **Phản ứng của hệ thống** |
| Người dùng chọn trang giới thiệu trên ứng dụng | Hệ thống hiển thị trang giới thiệu |

* + 1. Yêu cầu giao diện ngoài

#### GIAO DIỆN NGƯỜI DÙNG

Sử dụng màn hình của thiết bị thông minh

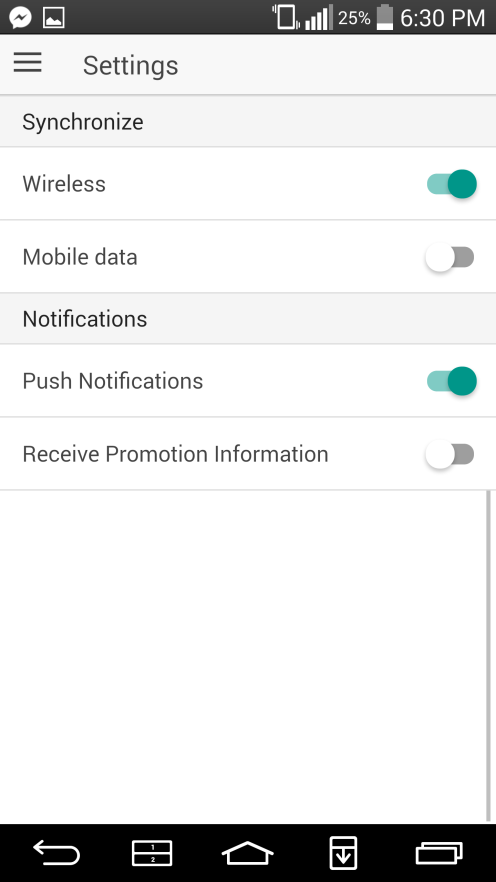
- Màn hình xem danh sách công việc



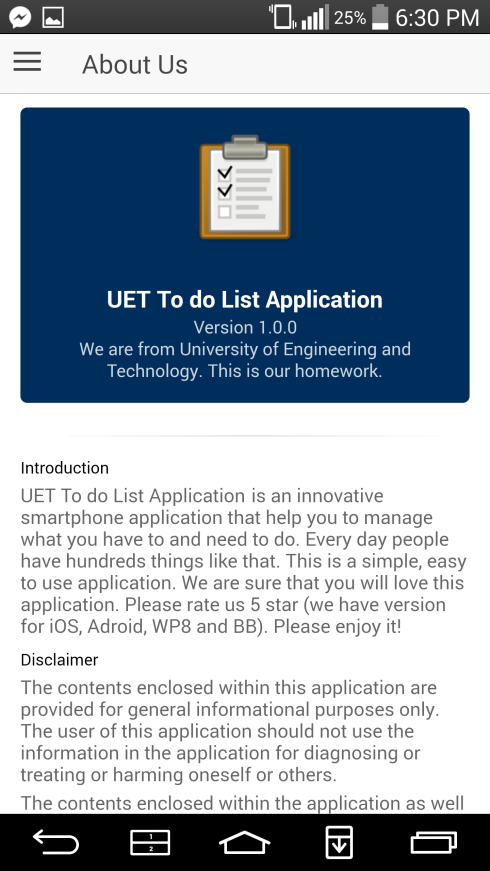
- Màn hình thêm công việc

|  |  |
| --- | --- |
| C:\xampp\htdocs\share\Screenshot_2016-03-15-18-29-33.png | C:\xampp\htdocs\share\Screenshot_2016-03-15-18-30-29.png |

- Màn hình cài đặt



- Màn hình giới thiệu



#### GIAO DIỆN PHẦN CỨNG

* Loa của thiết bị

#### GIAO DIỆN PHẦN MỀM

* Ứng dụng quản lý file của hệ điều hành
* Ứng dụng quản lý trạng thái kết nối mạng
* Ứng dụng hệ thống quản lý thông báo của hệ điều hành

### 2.3.3. YÊU CẦU HIỆU SUẤT

|  |  |
| --- | --- |
| **Tên công việc** | **Thời gian phản hồi** |
| Thêm từng công việc | Tức thời |
| Thêm công việc từ tệp | 30s/MB tệp dữ liệu |
| Xóa một công việc | Tức thời |
| Xóa toàn bộ danh sách | 1s/1000 công việc |
| Hiển thị thông báo | Tức thời |
| Phản hồi thông báo | Tức thời |
| Hiện trang cài đặt | Tức thời |
| Hiện trang giới thiệu | Tức thời |

### 2.3.4. RÀNG BUỘC THIẾT KẾ

#### RÀNG BUỘC CÔNG NGHỆ

* Thiết kế ứng với lập trình web tĩnh (html, js, css)
* Sử dụng Ionic Framework
* Sử dụng các template và plugin của Ionic để tương tác phần cứng

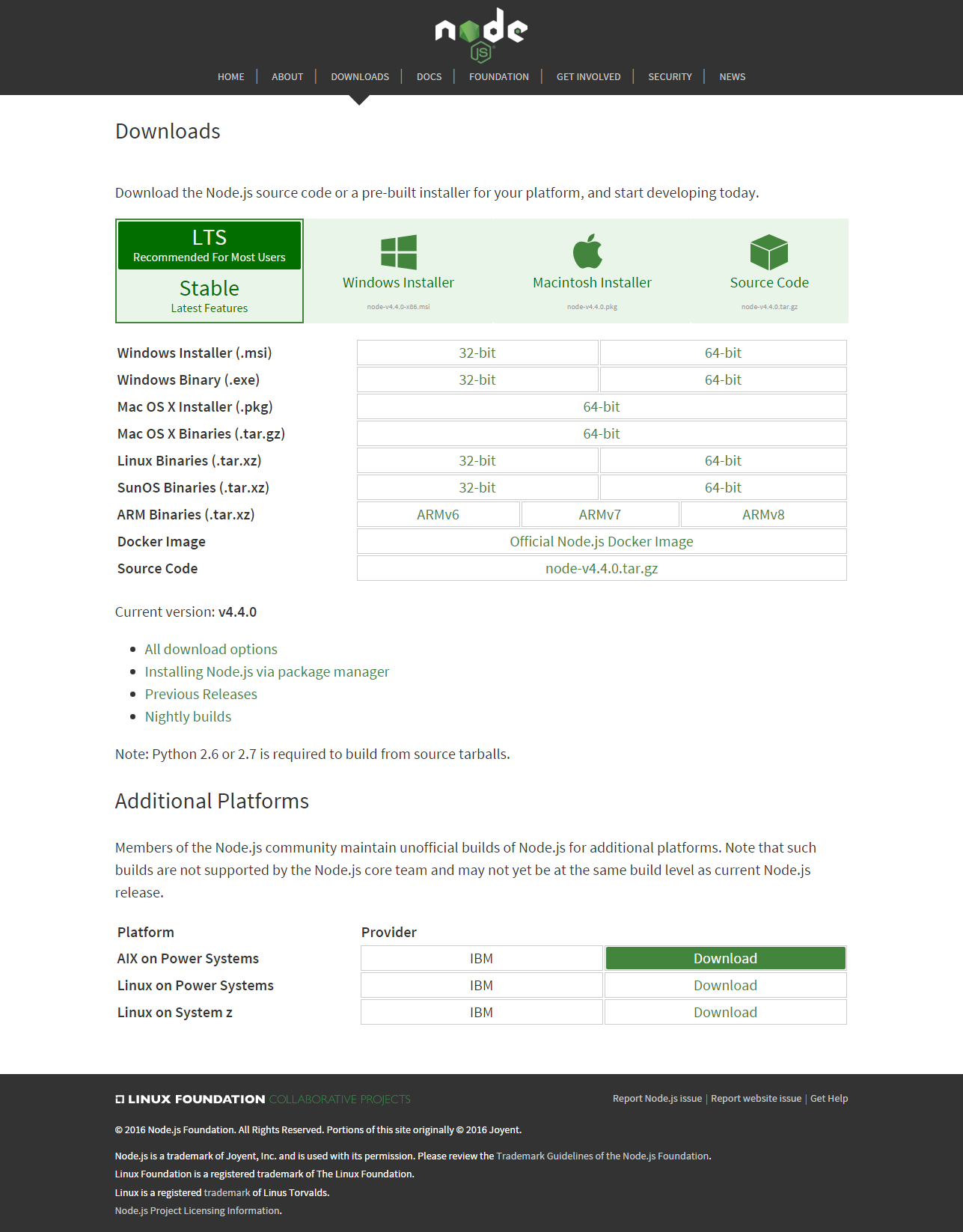
#### RÀNG BUỘC CÔNG VIỆC

* Cần hoàn thành trước ngày 24/3/2016
* Ngân sách nhỏ hơn 1 triệu đồng

# Chương 3. CÀI ĐẶT VÀ TRIỂN KHAI

## 3.1. CÀI ĐẶT NODEJS

Node.js là một nền tảng chạy trên môi trường V8 JavaScript runtime - một trình thông dịch JavaScript cực nhanh chạy trên trình duyệt Chrome. Bình thường thì bạn cũng có thể tải bộ V8 và nhúng nó vào bất cứ thứ gì; Node.js làm điều đó đối với các web server.



Hình 3.. Trang Tải chương trình Nodejs

Ionic được hỗ trợ bởi Nodejs, vì vậy cài sử dụng Nodejs sẽ giúp đơn giản hóa các công việc cài đặt và cấu hình Ionic Framework.

Chúng ta có thể tải Nodejs tại trang chủ https://nodejs.org/en/download/

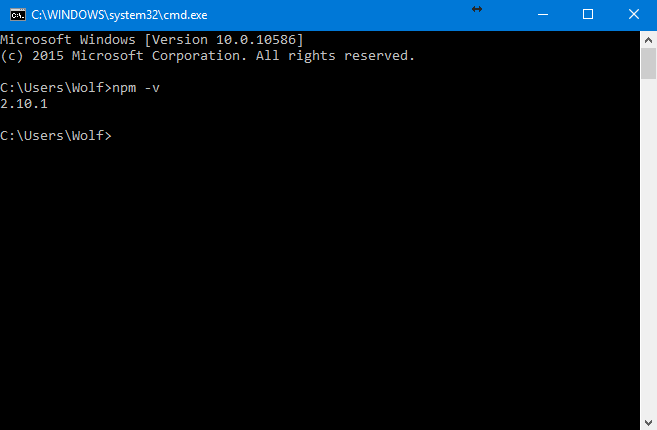
Các nên tảng hỗ trợ Nodejs

* Windows
* MacOS
* Linux
* SunOS
* ARM

Chúng ta cần kiểm tra kiến trúc của máy tính đang sử dụng là 32 bit hay 64 bit trước khi tải và cài đặt Nodejs. Việc kiểm tra này sẽ giúp chúng ta loại bỏ các lỗi phát sinh không mong muốn khi dùng Nodejs cài đặt các phần mềm khác, cụ thể ở đây là Ionic framework.

Để chắc chắn rằng Nodejs đã được cài thành công, chúng ta sử dụng lệnh kiểm tra phiên bản

|  |
| --- |
| npm -v |



Hình 3.. Kiểm tra phiên bản Nodejs

## 3.2. CÀI ĐẶT CORDOVA

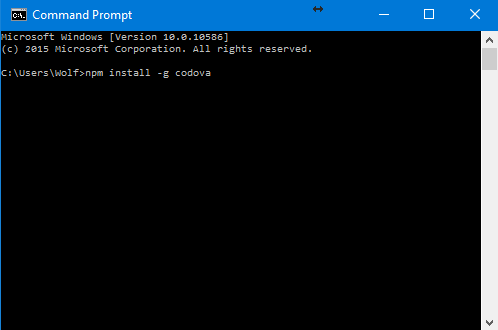
Sau khi đã có Nodejs trên máy, chúng ta tiến hành cài đặt [Apache Cordova](http://cordova.apache.org/) phiên bản mới nhất

Đối với Window, trên cmd ta sử dụng lệnh

|  |
| --- |
| npm install -g cordova |

Đối với Ubuntu (hoặc các hệ điều hành Unix khác), trên terminal của hệ điều hành, ta sử dụng lệnh:

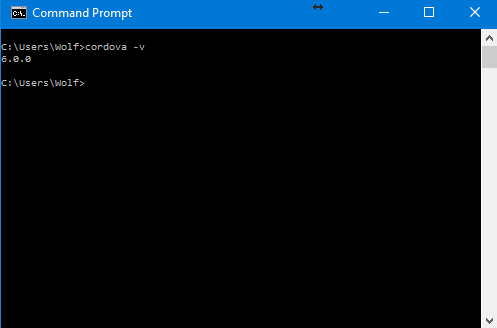
|  |
| --- |
| sudo npm install -g cordova |



Hình 3.. Cài đặt cordova trên Windows 10

Chúng ta cần phải kiểm tra xem cordova đã thực sự được cài đặt lên máy hay chưa. Ionic là một framework của cordova, do đó hãy chắc chắn chạy thành công lệnh kiểm tra phiên bản cordova trước khi tiếp tục các bước tiếp theo. Ta có thể kiểm tra phiên bản Cordova vừa được cài đặt với lệnh:

|  |
| --- |
| cordova -v |



Hình 3.. Kiểm tra phiên bản Cordova

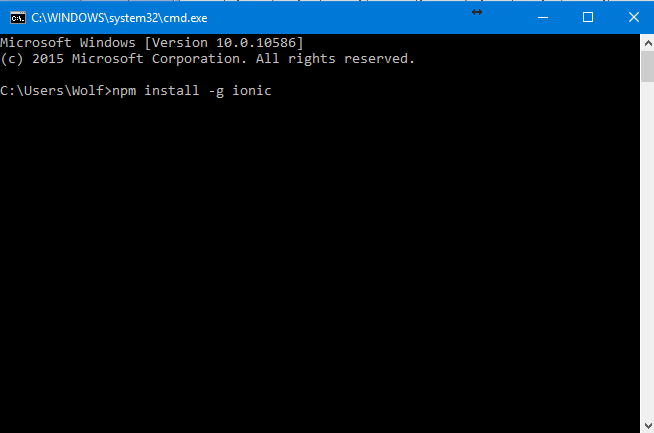
## 3.3. CÀI ĐẶT IONIC FRAMEWORK

Đối với Window, trên cmd ta sử dụng lệnh

|  |
| --- |
| npm install -g ionic |

Đối với Ubuntu (hoặc các hệ điều hành Unix khác), trên terminal của hệ điều hành, ta sử dụng lệnh:

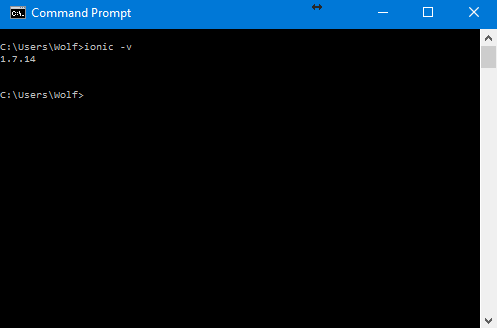
|  |
| --- |
| sudo npm install -g ionic |



Hình 3.. Cài đặt Ionic Framework trên Windows 10

Để chắc chắn rằng ionic đã được cài đặt thành công, tương tự như Nodejs và Cordova chúng ta có thể kiểm tra phiên bản của Ionic Framework với lệnh:

|  |
| --- |
| cordova -v |



Hình 3.. Kiểm tra phiên bản Ionic Framework

## 3.4. KHỞI TẠO ỨNG DỤNG

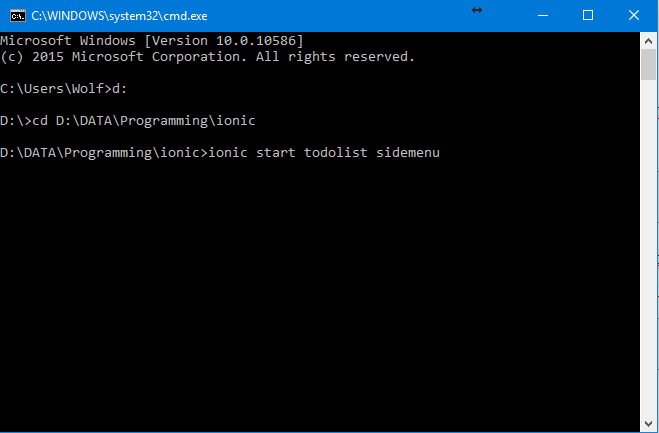
Sau khi đã cài đặt thành công Ionic Framework, chúng ta có thể tạo mới một ứng dụng Cordova ở bất kỳ đâu trên máy tính. Giả sử, chúng ta tạo các ứng dụng ionic của mình trong thư mục **D:\DATA\Programming\ionic**

Chúng ta di chuyển tới thư mục **D:\DATA\Programming\ionic** và tạo ứng dụng với câu lệnh mặc định sau:

|  |
| --- |
| ionic start <tên ứng dụng> |

Bên cạnh đó, chúng ta có thể sử dụng template có sẵn (*sidemenu* hoặc *tabs*) của Ionic. Ví dụ, chúng ta tạo ứng dụng tên **TodoList** với template *sidemenu* như sau:

|  |
| --- |
| ionic start todolist sidemenu |



Hình 3.. Tạo ứng dụng mới với Ionic Framework

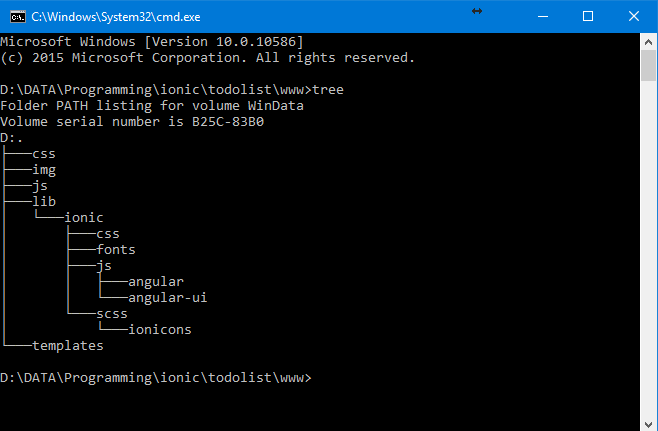
## 3.5. TRIỂN KHAI CODE TRÊN KHUNG CỦA ỨNG DỤNG

### 3.5.1. CẤU TRÚC THƯ MỤC

Trong thư mục mà Ionic Framework tạo ra, chúng ta chỉ cần quan tâm đến thư mục www, là nơi ta sẽ triển khai mã nguồn. Các thư mục khác là các thư mục lưu thông tin phát triển và chứa các dữ liệu khi biên dịch ứng dụng của chúng ta ra các nền tảng khác nhau.

Trong thư mục www, Ionic đã thiết kế cấu trúc thư mục hợp lý và tiện dụng đối với nhà phát triển ứng dụng.

* Thư muc css: Chứa các file định dạng giao diện cho ứng dụng
* Thư mục img: Chứa các file ảnh
* Thư mục js: Chứa các mã nguồn javascript
* Thư mục templates: Chứa các file html (phần view trong mô hình MVC)
* Thư mục lib: Chứa các thư viện có sẵn của Ionic, nhà phát triển sử dụng được những công cụ này một cách thuận tiện



Hình 3.. Cấu trúc trong thư mục www

### 3.5.2. CẤU TRÚC MÃ NGUỒN

***Hệ thống mã nguồn Javascript – Phần Controller***

Nền tảng của các ứng dụng được xây dựng bằng Ionic Framework là lập trình web tĩnh, do đó xương sống của mã nguồn chính là các file js.

Các file trong thư mục js

|  |  |
| --- | --- |
| **Tên file** | **Chức năng** |
| app.js | Chứa các thông tin cấu hình ban đầu của ứng dụng |
| controllers.js | Chứa các Controller điều phối các hoạt động của các Scope được định nghĩa trong các file html |

***Hệ thống template – Phần View***

Có thể thấy rằng cách tổ chức mã nguồn của Ionic phù hợp với nguyên lý phát triển hướng thành phần của công nghệ phần mềm hiện đại. Thư mục *templates* chứa các file html định dạng trước các trang màn hình cần có của ứng dụng, các nguyên mẫu này được điều khiển bởi các *controller*, dễ dàng lắp ghép vào ứng dụng tùy vào mục đích sử dụng.

Các file trong thư mục templates

|  |  |
| --- | --- |
| **Tên file** | **Chức năng** |
| menu.html | Chứa nguyên mẫu của menu trượt bên trái, nơi điều hướng chính của ứng dụng. |
| todo.html | Chứa nguyên mẫu của trang danh sách công việc cần làm (xem, thêm, xóa) |
| about.html | Chữa nguyên mẫu về giới thiệu ứng dụng |
| setting.html | Chứa nguyên mẫu phần cài đặt |

***Lưu trữ dữ liệu – Phần Model***

Được phát triển trên nền của Cordova, Ionic được trang bị ít nhất 4 phương pháp lưu trữ dữ liệu là LocalStorage, WebSQL, IndexedDB và các cơ chế lưu trữ do các plugin cung cấp.

Chi tiết về các phương pháp lưu trữ dữ liệu

|  |  |
| --- | --- |
| **Phương pháp lưu trữ** | **Mô tả ngắn** |
| LocalStorage | Còn được biết với các tên gọi như *web storage, simple storage*. API này cung phương pháp lưu trữ theo cặp (key, value)  Nó được phát triển tại nền WebView, theo đặc tả của W3C. |
| WebSQL | Giống như LocalStorage API này được cung cấp tại nền WebView, theo đặc tả của W3C.  Cơ sở dữ liệu được truy cập bởi các truy vấn SQL.  Các platform hỗ trợ:  - Android  - BlackBerry 10  - iOS |
| IndexedDB | API này được cung cấp tại nền WebView, có nhiều tính năng hơn LocalStorage nhưng ít tính năng hơn WebSQL  Các platform hỗ trợ:  - Android  - Windows Phone 8  - Windows 8 |
| Plugin-base Options | Được cung cấp bởi các nhà phát triển plugin của Ionic Framework hay Cordova. Ví dụ điền hình cho phương pháp lưu trữ này là plugin cordova-plugin-file |

Trong ứng dụng quản lý các công việc, chúng tôi sử dụng phương pháp Local-Storage vì nó phù hợp với nhu cầu sử dụng và tương đối dễ cài đặt và sử dụng.

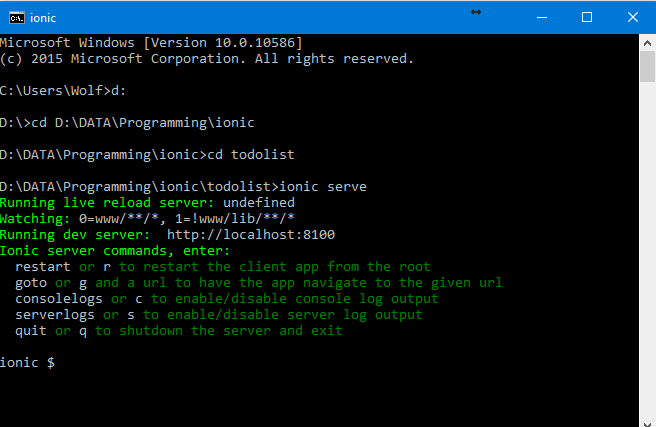
## 3.6. CHẠY ỨNG DỤNG

Sau khi tạo xong ứng dụng, chúng ta di chuyển vào thư mục *todolist*:

|  |
| --- |
| cd todolist |

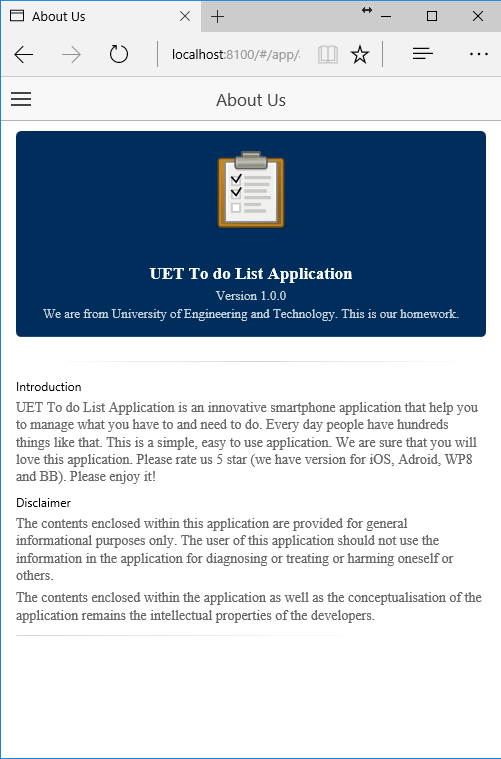
Chúng ta có thể xem ứng dụng vừa tạo có hình thù, hoạt động ra sao ngay trên trình duyệt web bằng cách chạy lệnh:

|  |
| --- |
| ionic serve |



Hình 3.. Ionic Framework hỗ trợ kiểm thử bằng chính trình duyệt

Đến đây, chúng ta đã có thể phát triển tiếp ứng dụng của mình với giả lập giao diện trên web view tại địa chỉ mặc định là [**http://localhost:8100**](http://localhost:8100/)**.**

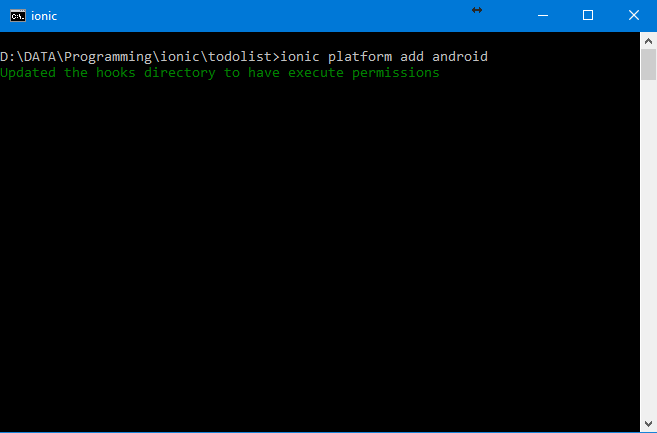


Hình 3.. Hình ảnh ứng dụng trên trình duyệt

Để xây dựng (build) và kiểm thử (test) trên các nền tảng khác nhau (ios, android), chúng ta cần add platform và build trên các nền tảng tương ứng.

|  |
| --- |
| ionic platform add android |

|  |
| --- |
| ionic platform add ios |

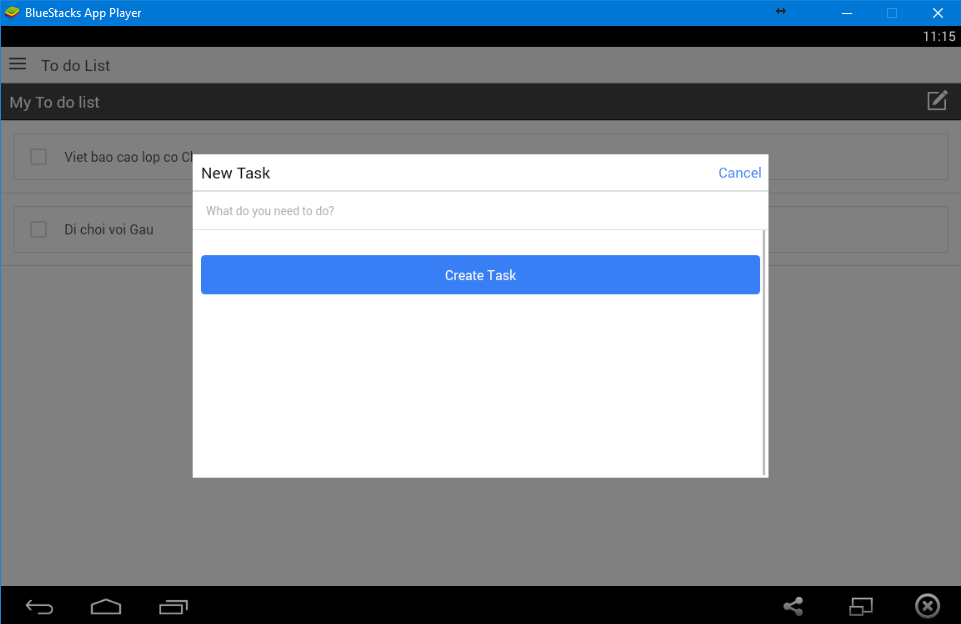


Hình 3.. Thêm nền tảng Android

Cuối cùng chạy thử ứng dụng (tương ứng với từng platform) với lệnh:

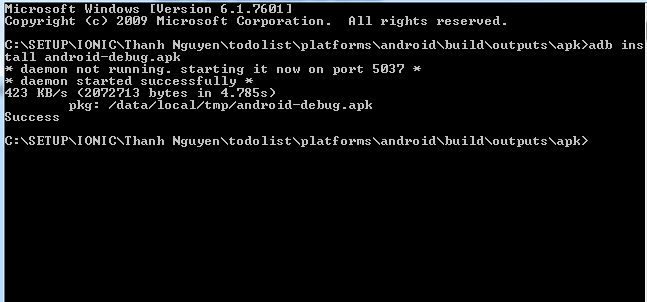
|  |
| --- |
| ionic build android |
| ionic emulate android |

Ứng dụng sẽ được tải lên thiết bị đã kết nối adb với máy tính của chúng ta. Tại đây chúng tôi sử dụng bluestack làm thiết bị Android phục vụ kiểm thử



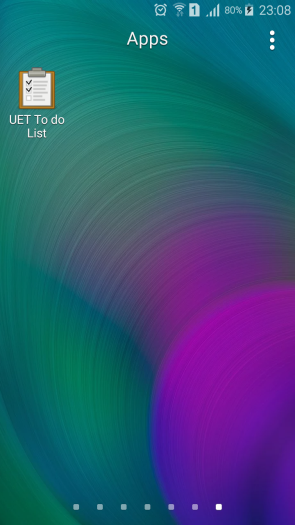
Hình 3.. Hình ảnh ứng dụng trên Bluestack

Chạy ứng dụng trên thiết bị android bằng cách chuyển đến thư mục chứa file apk vừa build rồi thực hiện lệnh **adb install**



Hình 3. adb install

Ứng dụng sẽ được cài vào điện thoại



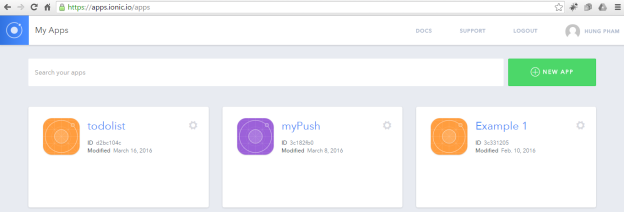
Hình 3. Cài ứng dụng vào điện thoại

Sau đó có thể upload ứng dụng lên kho của ionic với cú pháp **ionic upload**



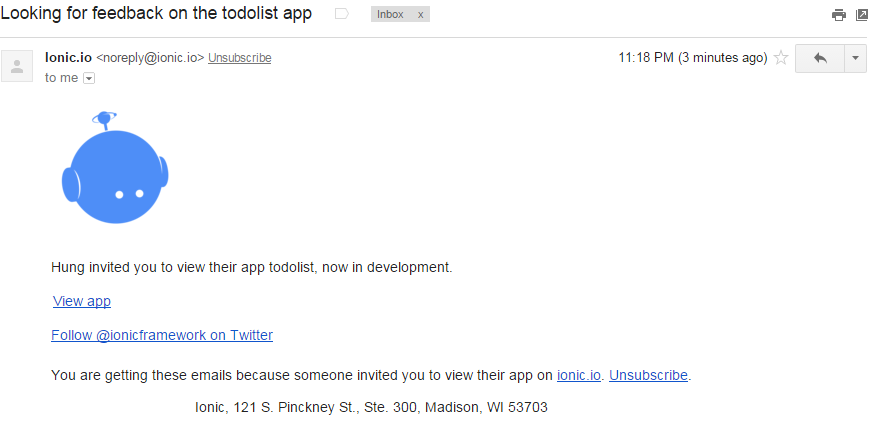
Hình 3. Upload ứng dụng lên kho

Ứng dụng sẽ được lưu trên server **apps.ionic.io**



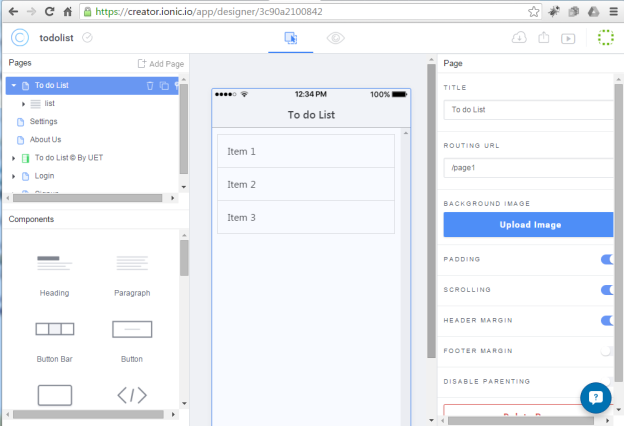
Hình 3. Lưu app trên server

Sau đó có thể chia sẻ ứng dụng với người khác qua email của họ với câu lệnh **ionic share email**



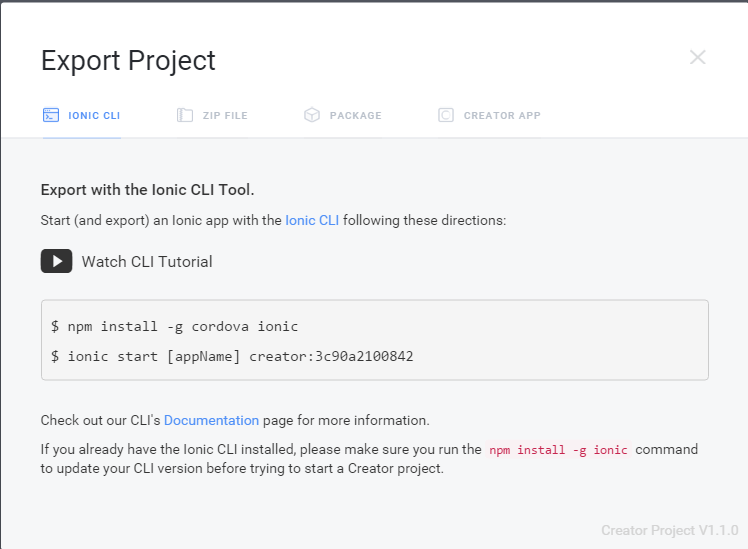
Hình 3. Share app qua email

Ngoài ra có thể sửa hoặc tạo ứng dụng trực tiếp trên trang web Creator của Ionic tại địa chỉ: https://**creator.ionic.io**

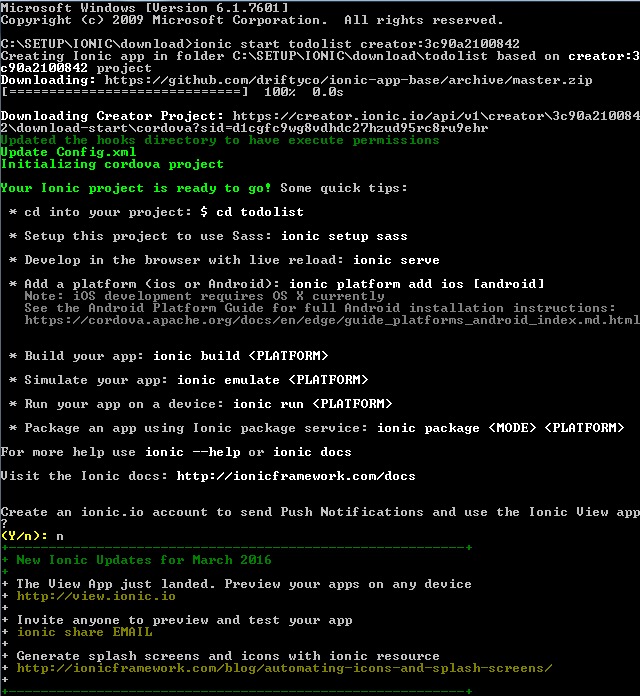


Hình 3. Tạo app trực tiếp trên web

Sau đó thực hiện tải source code của ứng dụng về máy tính để build



Hình 3. Tải source code từ web



Hình 3. Chạy app

## 3.7. So sánh cài đặt ứng dụng bằng Ionic và Android

Ở đây chúng ta tập trung so sánh việc xây dựng một ứng dụng bằng Ionic rồi built sang chạy ở Android và ứng dụng bằng Java chạy trên Android.

**Cài đặt và phát triển:**

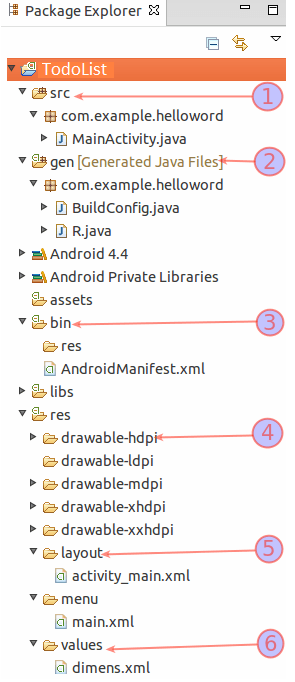
Để phát triển một ứng dụn native Android, trước tiên chúng ta cần chuẩn bị môi trường phát triển bằng cách cài đặt Java SDK hoặc Java JRE, rồi cài đặt Android SDK, và sử dụng một IDE nào đó để phát triển ứng dụng. Hiện tại có thể lựa chọn Eclipse hoặc Android Studio để khởi tạo và xây dựng một ứng dụng native cho Android. Với những ứng dụng phức tạp hơn, có sự giao tiếp với hệ điều hành hoặc các tiến trình hệ thống hay sử dụng các driver hoặc các native APIs, thì nhà phát triển cũng cần cài đặt thêm Android NDK để có thể tạo ra các project với sự kết hợp của Java và C++.

Thay vì phải tải các IDE với bộ SDK nặng nề để cài vào máy, đối với Ionic ta chỉ cần tải NodeJS và dùng một số command prompt không quá phức tạp để cài ionic và cordova. Việc xây dựng các app bằng Ionic không yêu cầu IDE phức tạp, người phát triển có thể dùng bất kỳ editor nào để code, ví dụ notepad++ là một sựa lựa trọn cho việc xây dựng các lớp giao diện hay các JavaScript function cho ứng dụng Ionic.

**Cấu trúc chương trình**:

Với Ionic như phần 3.5.1 đã mô tả.

Các lớp được dùng trong ứng dụng viết bằng Java bao gồm



Hình 3. Cấu trúc Android project

* (1) Mặc định chứa file MainActivity.java, chứa toàn bộ source code
* (5) Chứa các tập tin giao diện. Activity\_main.xml là phần giao diện chính của ứng dụng
* (3) bin: chứa file cài đặt. AndroidManifest.xml nơi khai báo activity được gọi
* (2) gen: nơi chứa các file R.java file này được sinh tự động, là nơi truy xuất các đối tượng trong code
* (4) Chứa các đối tượng đồ họa

**Chạy và test chương trình:**

Như phần 3.5 đã chỉ ra cách chạy một ứng dụng Ionic, bạn chỉ cần cd vào thư mục chứa project và chạy “Ionic serve”. Có thể dễ dàng test ứng dụng đang viết qua desktop browser, chỉnh sửa HTML, CSS, JavaScript file và tải lại trình duyệt để xem kết quả. Cũng có thể mô phỏng test trên các hệ điều hành bằng cách build ứng dụng theo OS bạn muốn cài ứng dụng. Bạn cũng có thể test ứng dụng của bạn trực tiếp trên mobile browser hoặc chạy chúng như một native app để test bằng câu lệnh “ionic run android”.

Với một ứng dụng Android bạn có thể F5/F6 để chạy/ debug chương trình, bộ phát triển hỗ trợ mô phỏng một thiết bị mobile có thể thao tác như trên điện thoại thật.

## 3.8 Đánh giá

Việc lập trình mobile theo cách truyền thống bạn cần phải học Java, Objective-C, C# để viết ứng dụng cho Android, iOS, Windows-phone hay viết ứng dụng theo hướng hybrid app đều có ưu và nhược điểm riêng. Một bên bạn có thể viết những native app có thể can thiệp sâu vào các chức năng của điện thoại hoặc một ứng dụng có thể sử dụng trên tất cả các hang điện thoại.

Nếu hiện tại đang có một ứng dụng viết cho Android bạn chỉ có thể cài đặt nó trên các điện thoại sử dụng Android OS, nếu muốn dùng nó ở các OS khác chỉ có cách là viết lại toàn bộ code và thiết kế của ứng dụng đó. Như vậy lập trình viên cần phải biết về tất cả các nền tảng và các ngôn ngữ lập trình tương ứng.

Có thể dễ dàng nhận ra lợi thế đầu tiên của cài đặt ứng dụng bằng Ionic là lập trình viên có thể viết code một lần dùng nhiều lần (built nó theo từng OS). Các lập trình viên chỉ cần biết kiến thức về HTML, CSS, JavaScript, các nhà phát triển Web dễ dàng chuyển sang mảng Mobile mà không cần tốn quá nhiều công sức để học ngôn ngữ mới.

# Kết Luận

Trong phạm vi báo cáo này, nhóm đã tìm hiểu và trình bày về các hướng phát triển ứng dụng di động, các đặc điểm và kiến trúc chung của từng loại ứng dụng di động. Từ những điểm mạnh yếu của từng loại ứng dụng di động, có một xu hướng cho việc phát triển ứng dụng di dộng một cách nhanh chóng và giảm chi phí phát triển cũng như bảo trì ứng dụng, đó chính là ứng dụng hybrid. Các tính chất đặc diểm của ứng dụng hybrid cũng được chỉ rõ trong báo cáo này, và quan trọng nhất, báo cáo đã trình bày cụ thể về nền tảng Ionic cho phát triển ứng dụng hybrid.

Ionic framework cho phép các nhà phát triển có thể nhanh chóng xây dựng các ứng dụng hybrid dựa trên công nghệ web như HTML, CSS và JavaScript. Ionic được xây dựng dựa trên kết hợp của nhiều công nghệ khác, tiêu biểu là Angular JS và Cordova. Kiến trúc và các thành phần quan trọng của Ionic cũng được trình bày chi tiết trong báo cáo.

Trong quá trình nghiên cứu nền tảng Ionic và phát triển ứng dụng Todo List, nhóm nhận thấy rằng đây là một nền tảng mạnh mẽ, với khả năng hỗ trợ phát triển ứng dụng nhanh chóng, không dòi hỏi người phát triển cần quá nhiều kỹ năng chuyên sâu và riêng biệt như đối với phát triển ứng dụng native. Tuy nhiên, để phát triển ứng dụng dựa trên Ionic, người phát triển cần hiểu biết JavaScript, HTML, CSS nói chung và AngularJS, Cordova ở mức độ nhất định.

Ứng dụng Todolist được phát triển trong quá trình các thành viên nhóm bắt đầu tìm hiểu Ionic framework nên còn tương đối cơ bản, chưa thực sự có nhiều chức năng ấn tượng. Trong tương lai, nhóm dự định sẽ phát triển thêm và cải thiện hiệu năng cho ứng dụng này với chức năng bổ sung như chia sẻ qua mạng xã hội (ví dụ Facebook, Google+) hay chức năng thông báo một item của TodoList bằng giọng nói sử dụng công nghệ Text-to-Speech với chất lượng tiếng nói cao (MOS ~ 3.5).

Tài liệu tham khảo

1. Goran K. (2014), *Building mobile app using Cordova and AngularJS, common practices.*
2. Jeremy W. (2016), *Ionic in action*, Manning Publications Co., New York.
3. Michaels, Ross & Cole, ltd. (mrc), *Native mobile apps: The wrong choice for business?*
4. <https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/>
5. http://blog.nawaratne.com/2015/11/rapid-hybrid-mobile-development-with.html
6. https://duythanhcse.wordpress.com/2013/03/25/bai-tap-3-cach-tao-android-project-va-tim-hieu-cau-truc-ben-trong-cua-mot-android-project/
7. http://www.slideshare.net/fch415/mobile-app-with-sencha-touch
8. [http://www.rishabhsoft.com/blog/kendo-ui-for-mobile-application#sthash.6qMv5f2e.dpuf](http://www.rishabhsoft.com/blog/kendo-ui-for-mobile-application)
9. <http://mobipro.vn/thu-vien/kien-thuc-chung/tim-hieu-bo-cong-cu-ho-tro-phat-trien-da-nen-tang-titanium.html>
10. <http://framework7.io/>