**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**---------------------------------------------**



## BÁO CÁO THỰC TẬP

## TỐT NGHIỆP ĐẠI HỌC

**Đề tài :**

**XÂY DỰNG ỨNG DỤNG QUẢN LÝ TIỀN TỆ TRÊN NỀN TẢNG ANDROID**

**Người hướng dẫn :** **TS. TRƯƠNG BÁ THÁI**

**Sinh viên thực hiện : Nguyễn Nhật Trường**

**Mã số sinh viên : N14DCCN287**

**Lớp : D14CQCPN01**

**TP.HCM tháng 08/ 2018**

**LỜI CÁM ƠN**

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Trong suốt thời gian từ khi bắt đầu học tập tại trường đến nay, em đã nhận được rất nhiều sự quan tâm, giúp đỡ của quý thầy cô, gia đình và bạn bè. Với lòng biết ơn sâu sắc nhất, em xin gửi đến quý thầy cô ở khoa Công Nghệ Thông Tin 2 – Học viện Công Nghệ Bưu Chính Viễn Thông cơ sở tại TP.HCM đã tận tâm chỉ bảo, truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường. Và đặc biệt trong học kỳ này, nếu không có những lời hướng dẫn, dạy bảo của các thầy cô thì em nghĩ bài báo cáo này của em rất khó có thể hoàn thiện được. Một lần nữa, em xin chân thành cảm ơn thầy cô. Bài báo cáo thực tập thực hiện trong khoảng thời gian 6 tuần. Bước đầu đi vào thực tế của em còn hạn chế và còn nhiều bỡ ngỡ. Do vậy, không tránh khỏi những thiếu sót là điều chắc chắn, em rất mong nhận được những ý kiến đóng góp quý báu của quý Thầy Cô và các bạn học cùng lớp để kiến thức của em trong lĩnh vực này được hoàn thiện hơn.

Em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với các thầy cô ở khoa Công Nghệ Thông Tin 2 – Học viện Công Nghệ Bưu Chính Viễn Thông cơ sở tại TP.HCM đã tạo điều kiện cho em để em có thể hoàn thành tốt bài báo cáo thực tập này. Đặc biệt em xin gửi lời cảm ơn đến thầy Trương Bá Thái với những buổi vừa học kết hợp với thực hành trên lớp cùng những buổi nói chuyện rất hữu ích đã giúp em định hướng và hoàn thành tốt đề tài này.

Trong quá trình làm bài báo cáo, cũng như là trong quá trình làm đề tài khó tránh khỏi sai sót, rất mong các Thầy, Cô bỏ qua. Đồng thời do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp Thầy, Cô để em học thêm được nhiều kinh nghiệm làm hành trang vững chắc để em tự tin theo đuổi sự nghiệp của mình.

*Em xin chân thành cảm ơn!*

***Thành phố Hồ Chí Minh, 08/2018***

*Nguyễn Nhật Trường*

# MỤC LỤC

[**MỞ ĐẦU 1**](#_Toc521577919)

[**CHƯƠNG I: GIỚI THIỆU ĐỀ TÀI 2**](#_Toc521577920)

[1.1. Mục đích 2](#_Toc521577921)

[1.2. Công nghệ sử dụng 2](#_Toc521577922)

[**CHƯƠNG II: GIỚI THIỆU CÔNG NGHỆ SỬ DỤNG 3**](#_Toc521577923)

[2.1. Giới thiệu Java. 3](#_Toc521577924)

[2.2. Giới thiệu XML 3](#_Toc521577929)

[2.3. Giới thiệu Android 5](#_Toc521577932)

[2.3.1. Hệ điều hành Android là gì 5](#_Toc521577933)

[2.3.2. Giao diện 6](#_Toc521577934)

2.3.3 Mã nguồn mở 8

[2.3.4 Giá cả hợp lý từ bình dân đến cao cấp 9](#_Toc521577935)

2.3.5 **Kho ứng dụng Google Play Store đồ sộ 10**

2.3.6 **Dễ sử dụng 10**

[2.3.7. Ưu và nhược điểm 11](#_Toc521577936)

[2.3.7.1. Ưu điểm: 11](#_Toc521577937)

[2.3.7.2. Nhược điểm: 11](#_Toc521577938)

[2.4. Giới thiệu RESTful API 11](#_Toc521577939)

[2.4.1. API là gì? 11](#_Toc521577940)

[2.4.2. Ví dụ về API 12](#_Toc521577942)

[2.4.3. RESTful API là gì? 12](#_Toc521577945)

[*2.5. Các thư viện hỗ trợ trong quá trình xây dựng ứng dụng* 13](#_Toc521577947)

[2.5.1. Butter knife 13](#_Toc521577948)

[2.5.2. Glide 13](#_Toc521577949)

[2.5.3. Retrofit 13](#_Toc521577950)

[2.5.4. Dagger 14](#_Toc521577951)

[2.5.5. MVP? 14](#_Toc521577952)

[2.5.6. Kiến trúc Android 14](#_Toc521577953)

[**CHƯƠNG III: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 16**](#_Toc521577954)

[3.1 Màn hình Splash: 16](#_Toc521577955)

[3.2 Màn hình Login: 16](#_Toc521577958)

[3.3 Màn hình đăng kí: 17](#_Toc521577961)

[3.4 Màn hình Home: 17](#_Toc521577964)

[3.5 Màn hình Profile: 18](#_Toc521577967)

[3.6 Màn hình Detail: 18](#_Toc521577970)

[3.7 Màn hình tạo các khoản thu, chi, nợ: 19](#_Toc521577973)

[3.8 Màn hình Todo List: 19](#_Toc521577976)

[3.9 Màn hình tạo todo: 20](#_Toc521577979)

[3.10 Màn hình Chart: 20](#_Toc521577982)

[**CHƯƠNG IV: CÀI ĐẶT VÀ SỬ DỤNG 21**](#_Toc521577985)

[4.1 Mục tiêu của ứng dụng: 21](#_Toc521577986)

[4.2 Phân tích yêu cầu: 21](#_Toc521577987)

[4.2.1 Yêu cầu chức năng: 21](#_Toc521577988)

[4.2.2 Yêu cầu phi chức năng: 21](#_Toc521577989)

[4.3 Mô tả chức năng: 22](#_Toc521577990)

[4.3.1 Đăng nhập: 22](#_Toc521577991)

[4.3.2 Đăng kí: 22](#_Toc521577992)

[4.3.3 Chỉnh sửa thông tin người dùng: 22](#_Toc521577993)

[4.3.4 Xem tổng quan 22](#_Toc521577994)

[4.3.5 Tạo dữ liệu các khoản thu, chi , nợ 23](#_Toc521577995)

[4.3.6 Navigation 23](#_Toc521577996)

[4.3.7 Xem chi tiết lọ 23](#_Toc521577997)

[4.3.9 Biểu đồ 24](#_Toc521577998)

[4.3.10 Dự định, công việc phải làm 24](#_Toc521577999)

[**KẾT LUẬN 25**](#_Toc521578000)

[**DANH MỤC TÀI LIỆU THAM KHẢO 26**](#_Toc521578007)

# MỞ ĐẦU

Tiền là thứ quan trọng trong cuộc sống mỗi chúng ta, là phương tiện để trao đổi hàng hóa, mua bán, giao dịch…Mỗi người có một số tiền khác nhau nhưng làm sao để quản lý tiền một cách khoa học, hợp lí nhất thật sự là vấn đề. **Sự khác biệt lớn nhất giữa người giàu và người nghèo là ở cách quản lý tài chính. Người giàu biết cách**[quản lý tài chính cá nhân hiệu quả](http://phamngocanh.com/thu-vien/phat-trien-ca-nhan-thu-vien/phuong-phap-quan-ly-tai-chinh-ca-nhan-hieu-qua/)**còn người nghèo thì không hoặc trốn tránh mọi vấn đề về tiền bạc. Trong quá trình phát triển sự nghiệp nếu bạn bỏ qua khâu quản lý tài chính, sẽ là sai lầm.**

Nếu bạn không biết quản lý tốt những gì bạn đang có, cuộc sống của bạn sẽ chẳng bao giờ có thêm giá trị nào cả. Việc quản lý tài chính mỗi ngày sẽ giúp bạn tạo dựng thói quen tuyệt vời và có lợi cho cuộc sống của bạn.

Quản lý tài chính không đơn giản chỉ là việc biết cách chi tiêu và sử dụng nguồn tài chính do mình tạo ra một cách hợp lý. Việc quản lý tài chính còn hàm ý rất nhiều yếu tố. Trong đó bao gồm: kế hoạch phân chia chi tiêu khoa học, xây dựng quỹ tiết kiệm, lựa chọn các kênh đầu tư hiệu quả….

Theo một báo cáo của các nhà lý luận kinh doanh, các triệu phú trung bình dành khoảng 8.4 giờ mỗi tháng để quản lý và hoạch định tài chính. Trong thực tế, phần lớn chúng ta chưa có phương pháp quản lý tài chính tốt.

Ứng dụng “Quản lý tiền tệ” sẽ hỗ trợ thuận tiện nhất cho người dùng quản lý tiền một cách tốt nhất.

* + 1. GIỚI THIỆU ĐỀ TÀI
  1. Mục đích

Hiện nay, smart phone đã phổ biến trên toàn thế giới. Hầu như mỗi người đều có cho mình 1 chiếc smart phone. Smart phone không những có những chức năng nghe, gọi, nhắn tin, chụp ảnh, quay video mà còn có thể kết nối Internet. Từ đó, ứng dụng trên điện thoại di động (mobile app) ra đời. Đến nay, có rất nhiều ứng dụng cho việc giải trí, kinh doanh, giáo dục, sức khỏe…Quản lí tiền đúng cách luôn là việc quan trọng. Và phương pháp 6 chiếc lọ giúp chúng ta quản lí tiền một cách khoa học nhất. Vì thế áp dụng phương pháp này để xây dựng một ứng dụng để quán lí tiền tệ giúp cho người dùng thuận tiện quản lý tiền tốt nhất mà không cần phải ghi chép, tính toán. Ứng dụng sẽ tự động xử lí dữ liệu và đảm bảo dữ liệu được an toàn tránh mất mát so với những cách thủ công truyền thống.

1.2. Công nghệ sử dụng

* Java
* Xml
* Android
* RESTful API
* Các thư viện hỗ trợ trong quá trình xây dựng ứng dụng
* Mô hình MVP
  + 1. GIỚI THIỆU CÔNG NGHỆ SỬ DỤNG

2.1. Giới thiệu Java.

Java (phiên âm Tiếng Việt: "*Gia-va*") là một [ngôn ngữ lập trình](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh) [hướng đối tượng](https://vi.wikipedia.org/wiki/L%E1%BA%ADp_tr%C3%ACnh_h%C6%B0%E1%BB%9Bng_%C4%91%E1%BB%91i_t%C6%B0%E1%BB%A3ng) (OOP) và dựa trên các lớp (class). Khác với phần lớn ngôn ngữ lập trình thông thường, thay vì [biên dịch](https://vi.wikipedia.org/wiki/Tr%C3%ACnh_bi%C3%AAn_d%E1%BB%8Bch) [mã nguồn](https://vi.wikipedia.org/wiki/M%C3%A3_ngu%E1%BB%93n) thành [mã máy](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_m%C3%A1y) hoặc [thông dịch](https://vi.wikipedia.org/wiki/Tr%C3%ACnh_th%C3%B4ng_d%E1%BB%8Bch) mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành [bytecode](https://vi.wikipedia.org/wiki/Bytecode), bytecode sau đó sẽ được môi trường thực thi (runtime environment) chạy.

Trước đây, Java chạy chậm hơn những ngôn ngữ dịch thẳng ra mã máy như C và [C++](https://vi.wikipedia.org/wiki/C%2B%2B), nhưng sau này nhờ công nghệ "biên dịch tại chỗ" - Just in time compilation, khoảng cách này đã được thu hẹp, và trong một số trường hợp đặc biệt Java có thể chạy nhanh hơn.

[Cú pháp](https://vi.wikipedia.org/wiki/C%C3%BA_ph%C3%A1p_h%E1%BB%8Dc) Java được vay mượn nhiều từ [C](https://vi.wikipedia.org/wiki/C_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)) & [C++](https://vi.wikipedia.org/wiki/C%2B%2B) nhưng có cú pháp hướng đối tượng đơn giản hơn và ít tính năng xử lý cấp thấp hơn. Do đó việc viết một chương trình bằng Java dễ hơn, đơn giản hơn, đỡ tốn công sửa lỗi hơn.

Trong Java, hiện tượng rò rỉ bộ nhớ hầu như không xảy ra do bộ nhớ được quản lý bởi Java Virtual Machine (JVM) bằng cách tự động "dọn dẹp rác". Người lập trình không phải quan tâm đến việc cấp phát và xóa bộ nhớ như C, C++. Tuy nhiên khi sử dụng những tài nguyên mạng, file IO, database (nằm ngoài kiểm soát của JVM) mà người lập trình không đóng (close) các streams thì rò rỉ dữ liệu vẫn có thể xảy ra

2.2. Giới thiệu XML

XML (viết tắt từ [tiếng Anh](https://vi.wikipedia.org/wiki/Ti%E1%BA%BFng_Anh): *eXtensible Markup Language*, tức "Ngôn ngữ đánh dấu mở rộng") là [ngôn ngữ đánh dấu](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_%C4%91%C3%A1nh_d%E1%BA%A5u) với mục đích chung do [W3C](https://vi.wikipedia.org/wiki/W3C) đề nghị, để tạo ra các ngôn ngữ đánh dấu khác. Đây là một tập con đơn giản của [SGML](https://vi.wikipedia.org/wiki/SGML), có khả năng mô tả nhiều loại [dữ liệu](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u) khác nhau. Mục đích chính của XML là đơn giản hóa việc chia sẻ dữ liệu giữa các hệ thống khác nhau, đặc biệt là các hệ thống được kết nối với [Internet](https://vi.wikipedia.org/wiki/Internet). Các ngôn ngữ dựa trên XML (Ví dụ: [RDF](https://vi.wikipedia.org/wiki/RDF), [RSS](https://vi.wikipedia.org/wiki/RSS_(%C4%91%E1%BB%8Bnh_d%E1%BA%A1ng_t%E1%BA%ADp_tin)), [MathML](https://vi.wikipedia.org/wiki/MathML), [XHTML](https://vi.wikipedia.org/wiki/XHTML), [SVG](https://vi.wikipedia.org/wiki/SVG), [GML](https://vi.wikipedia.org/wiki/GML)

và [cXML](https://vi.wikipedia.org/wiki/CXML)) được định nghĩa theo cách thông thường, cho phép các chương trình sửa đổi và kiểm tra hợp lệ bằng các ngôn ngữ này mà không cần có hiểu biết trước về hình thức của chúng.

XML cung cấp một phương tiện dùng văn bản (*text*) để mô tả thông tin và áp dụng một cấu trúc kiểu cây cho thông tin đó. Tại mức căn bản, mọi thông tin đều thể hiện dưới dạng text, chen giữa là các [thẻ đánh dấu](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_%C4%91%C3%A1nh_d%E1%BA%A5u) (*markup*) với nhiệm vụ ký hiệu sự phân chia thông tin thành một cấu trúc có thứ bậc của các *dữ liệu ký tự*, các *phần tử* dùng để chứa dữ liệu, và các *thuộc tính* của các phần tử đó.

Bằng cách cho phép các tên dữ liệu, cấu trúc thứ bậc được phép, và ý nghĩa của các phần tử và thuộc tính có tính chất mở và có thể được định nghĩa bởi một [giản đồ](https://vi.wikipedia.org/w/index.php?title=Gi%E1%BA%A3n_%C4%91%E1%BB%93_XML&action=edit&redlink=1) tùy biến được, XML cung cấp một cơ sở cú pháp cho việc tạo lập các ngôn ngữ đánh dấu dựa XML theo yêu cầu. Cú pháp chung của các ngôn ngữ đó là cố định — các tài liệu phải tuân theo các quy tắc chung của XML, bảo đảm rằng tất cả các phần mềm hiểu XML ít ra cũng phải có khả năng đọc (*phân tích cú pháp* - *parse*) và hiểu bố cục tương đối của thông tin trong các tài liệu đó. Giản đồ chỉ bổ sung một tập các ràng buộc cho các quy tắc cú pháp. Các giản đồ thường hạn chế tên của phần tử và thuộc tính và các cấu trúc thứ bậc được phép, ví dụ, chỉ cho phép một phần tử tên 'ngày sinh' chứa một phần tử tên 'ngày' và một phần tử có tên 'tháng', mỗi phần tử phải chứa đúng một ký tự. Đây là điểm khác biệt giữa XML và [HTML](https://vi.wikipedia.org/wiki/HTML). HTML có một bộ các phần tử và thuộc tính không mềm dẻo, chỉ có một tác dụng và nói chung là không thể dùng cho mục đích khác.

2.3. Giới thiệu Android

### 2.3.1. Hệ điều hành Android là gì



[Android](https://www.dienmayxanh.com/dien-thoai?g=android) là một hệ điều hành có mã nguồn mở dựa trên nền tảng Linux được thiết kế dành cho các [thiết bị di động](https://www.dienmayxanh.com/dien-thoai) có [màn hình cảm ứng](http://vi.wikipedia.org/wiki/M%C3%A0n_h%C3%ACnh_c%E1%BA%A3m_%E1%BB%A9ng) như điện thoại thông minh và máy tính bảng.

Ban đầu, Android được phát triển bởi Tổng công ty Android, với sự hỗ trợ tài chính từ Google, sau này được chính Google mua lại vào năm 2005 và hệ điều hành Android đã ra mắt vào năm 2007. Chiếc điện thoại đầu tiên chạy Android là HTC Dream được bán vào ngày 22 tháng 10 năm 2008.

Chính mã nguồn mở cùng với giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên được điều chỉnh và phân phối Android một cách tự do. Những yếu tố này đã giúp Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới. Android chiếm 75% thị phần [điện thoại thông minh](https://www.dienmayxanh.com/tag/dien-thoai-smartphone) trên toàn thế giới vào thời điểm quý 3 năm 2012, với tổng cộng 500 triệu thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Tháng 10 năm 2012, đã có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play (cửa hàng ứng dụng chính của Android) ước tính khoảng 25 tỷ lượt. Hiện nay con số này đã giảm xuống do sự ảnh hưởng lớn của [iOS từ Apple](https://www.dienmayxanh.com/dien-thoai-apple-iphone) và một phần nhỏ của [Windows Phone](https://www.dienmayxanh.com/dien-thoai?g=khoang-4#_), tuy nhiên Android vẫn dẫn đầu thị phần.

### 2.3.2. Giao diện



Giao diện người dùng của [Android](https://www.dienmayxanh.com/dien-thoai?g=android) dựa trên nguyên tắc tác động trực tiếp, sử dụng cảm ứng chạm tương tự như những động tác ngoài đời thực như vuốt, chạm, kéo dãn và thu lại để xử lý các đối tượng trên màn hình.

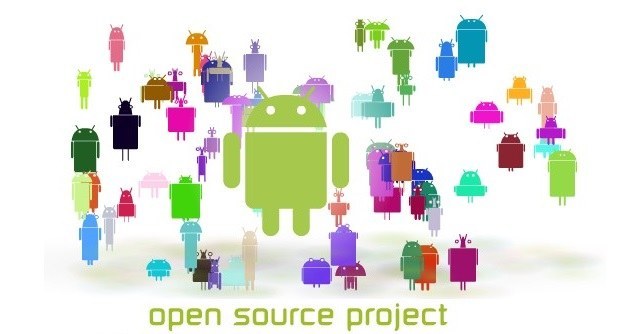
Các thiết bị Android sau khi khởi động sẽ hiển thị màn hình chính, điểm khởi đầu với các thông tin chính trên thiết bị, tương tự như khái niệm desktop trên máy tính để bàn. Màn hính chính Android thường gồm nhiều biểu tượng (icon) và tiện ích (widget). Giao diện màn hình chính của Android có thể tùy chỉnh ở mức cao, cho phép người dùng tự do sắp đặt hình dáng cũng như hành vi của thiết bị theo sở thích.

Những ứng dụng do các hãng thứ ba có trên Google Play và các kho ứng dụng khác còn cho phép người dùng thay đổi "chủ đề" của màn hình chính, thậm chí bắt chước hình dáng của hệ điều hành khác như [Windows Phone](https://www.dienmayxanh.com/dien-thoai?g=khoang-4#_) hay iOS. Phần lớn những nhà sản xuất, và một số nhà mạng, thực hiện thay đổi hình dáng và hành vi của các thiết bị Android của họ để phân biệt với các hãng cạnh tranh.

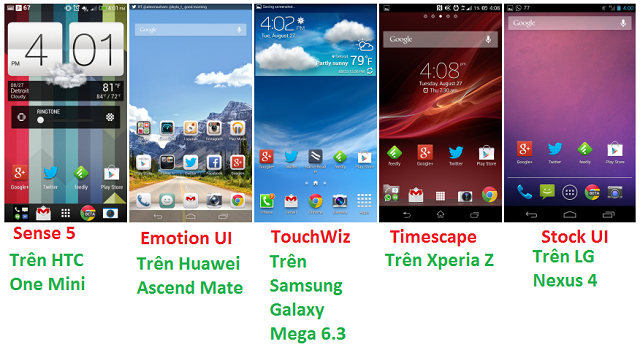


Google đưa ra các bản cập nhật lớn cho [Android](https://www.dienmayxanh.com/dien-thoai?g=android) theo chu kỳ từ 6 đến 9 tháng, mà phần lớn thiết bị đều có thể nhận được qua sóng không dây. Bản cập nhật lớn mới nhất là Android Oreo 8.0 .

**2.3.3 Mã nguồn mở**

****

Trong khi các hệ điều hành như iOS, WindowPhone sử dụng mã nguồn đóng thì Android lại đi ngược lại bằng cách mở mã nguồn và cung cấp cho người dùng tùy ý chỉnh sửa, mặc sức “xào nấu” mà không có sự can thiệp hay cấm cản từ Google. Do đó, trước khi các nhà sản xuất lớn như Samsung, Sony, LG, HTC, Oppo … cài đặt hệ điều hành Android trên thiết bị của mình thì họ đã tinh chỉnh và bổ sung rất nhiều thứ từ giao diện, đến vô số những tính năng thông minh, tiện lợi.



### ****2.3.4 Giá cả hợp lý từ bình dân đến cao cấp****



Mức giá để sở hữu một thiết bị là vấn đề mà không ít người quan tâm đến. Với các thiết bị Android, trong khoảng từ 1 – 20 triệu đồng, bạn sẽ có muôn vàn những lựa chọn, từ những thiết bị bình dân vừa túi tiền đến những thiết bị cao cấp sở hữu vô số những tính năng hữu ích. Trên thị trường hiện nay có rất nhiều hãng điện tử như Samsung, LG, Sony, Lenovo, Oppo, Huawei … tung ra hàng loạt các sản phẩm đủ mọi kích cỡ, đủ mọi mức giá vì vậy người dùng sẽ có nhiều sự để lựa chọn hơn so với các thiết bị sử dụng hệ điều hành khác.

**2.3.5** **Kho ứng dụng Google Play Store đồ sộ**



Trước đây, khi mới ra mắt kho ứng dụng Google Play chỉ vỏn vẹn có một số ứng dụng. Nhưng đến nay, kho ứng dụng này đã đạt đến hơn 650.000 ứng dụng, trong đó có rất nhiều ứng dụng miễn phí mà vô cùng tiện dụng. Đây cũng là lý do để giải thích cho sự phát triển chóng mặt của hệ điều hành Android trong thế giới di động. Nếu bạn sử dụng một thiết bị Android, chắc hẳn bạn sẽ không phải bận tâm vì “thiếu ứng dụng để cài” mà cái bạn sẽ bận tâm đó là “không đủ bộ nhớ để cài”.

**2.3.6** **Dễ sử dụng**

****

Có lẽ hệ điều hành Android chưa chắc đã dễ sử dụng hơn các hệ điều hành khác (điều này tùy thuộc vào ý kiến chủ quan của mỗi người), nhưng vì nó đã quá phổ biến trên các thiết bị di động, nên chắc hẳn bạn không cảm thấy lạ lẫm gì với nó nữa. Bạn sẽ có 2 môi trường để làm việc trên các thiết bị Android, đó là màn hình chính (HomeScreen) và menu chính. Màn hình chủ giống như một ngôi nhà của bạn, gồm có các bức tường xung quanh mà bạn có thể thõa sức trang trí, treo đồng hồ, dán lịch, đặt máy nghe nhạc … Menu chính thì giống như trong hộc bàn của bạn, có đầy đủ những thứ bạn cần như: máy nhắn tin, máy nghe nhạc, điện thoại, máy chụp hình …

### ****2.3.7. Ưu và nhược điểm****

#### ****2.3.7.1. Ưu điểm:****

+ Là hệ điều hành có mã nguồn mở nên khả năng tuỳ biến cao, có thể tùy ý chỉnh sửa mà không có sự can thiệp hay cấm cản từ Google.

+ Đa dạng sản phẩm, rất nhiều hãng điện thoại, thiết bị công nghệ đã ưu ái chọn Android cho thiết bị của họ, giá cả thì hợp lý từ bình dân đến cao cấp.

+ Kho ứng dụng Google Play Store đồ sộ.

+ Thân thiện và dễ sử dụng.

+ Khả năng đa nhiệm, chạy cùng lúc nhiều ứng dụng cao.

#### ****2.3.7.2. Nhược điểm:****

+ Khả năng dọn rác kém.

+ Dễ nhiễm phần mềm độc hại và virus. Do tính chất mã nguồn mở, nhiều phần mềm không được kiểm soát có chất lượng không tốt hoặc lỗi bảo mật vẫn được sử dụng.

+ Kho ứng dụng quá nhiều dẫn đến khó kiểm soát chất lượng, thiếu các ứng dụng thật sự tốt.

+ Sự phân mảnh lớn. Trong khi một số thiết bị Android xuất sắc đã trình làng như Samsung Galaxy S9, S9 Plus …, vẫn còn rất nhiều sản phẩm giá rẻ bình thường khác.

+ Cập nhật không tự động với tất cả thiết bị. Khi một phiên bản hệ điều hành mới ra mắt, không phải tất cả sản phẩm đều được cập nhật, thậm chí nếu muốn trải nghiệm bạn thường xuyên phải mua mới thiết bị.

2.4. Giới thiệu RESTful API

2.4.1. API là gì?

****API****là viết tắt của ****Application Programming Interface****(giao diện lập trình ứng dụng), là phần mềm trung gian cho phép 2 ứng dụng giao tiếp với nhau. Mỗi khi sử dụng các ứng dụng, chẳng hạn như Facebook, hoặc các ứng dụng gửi tin nhắn tức thì hoặc kiểm tra thời tiết trên điện thoại, đồng nghĩa với việc bạn đang sử dụng API.

### 2.4.2. Ví dụ về API

Khi sử dụng ứng dụng trên thiết bị di động, ứng dụng sẽ kết nối Internet và gửi dữ liệu tới máy chủ. Sau đó máy chủ lấy dữ liệu, diễn giải dữ liệu, thực hiện các hành động cần thiết và gửi dữ liệu trở lại thiết bị của bạn. Ứng dụng giải thích dữ liệu và hiển thị các thông tin đọc được cho bạn. Và nó được gọi là API.

Thử tưởng tượng bạn đang ở trong một nhà hàng, và menu có 1 loạt các món ăn để bạn lựa chọn. Nhà bếp là một phần của hệ thống, thực hiện nhiệm vụ chuẩn bị các món mà bạn đặt. Cái còn thiếu ở đây là người để truyền đạt món ăn mà bạn yêu cầu tới nhà bếp và mang đồ ăn ra cho bạn. Đó chính là người phục vụ hoặc có thể nói là API. Người phục vụ sẽ truyền đạt các yêu cầu của bạn tới nhà bếp và mang đồ ăn ra cho bạn.

### 2.4.3. RESTful API là gì?

RESTful API là một tiêu chuẩn dùng trong việc thết kế các thiết kế API cho các ứng dụng web để quản lý các *resource*. *RESTful* là một trong những kiểu thiết kế API được sử dụng phổ biến nhất ngày nay.

Trọng tâm của REST quy định cách sử dụng các HTTP method (như *GET*, *POST*, *PUT*, *DELETE*...) và cách định dạng các URL cho ứng dụng web để quản các resource. Ví dụ với một trang blog để quản lý các bài viết chúng ta có các URL đi với HTTP method như sau:

* URL tạo bài viết: http://my-blog.xyz/posts. Tương ứng với HTTP method là POST
* URL đọc bài viết với ID là 123: http://my-blog.xyz/posts/123. Tương ứng với HTTP method là GET
* URL cập nhật bài viết với ID là 123: http://my-blog.xyz/posts/123. Tương ứng với HTTP method là PUT
* URL xoá bài viết với ID là 123: http://my-blog.xyz/posts/123. Tương ứng với HTTP method là DELETE

Với các ứng dụng web được thiết kế sử dụng RESTful, lập trình viên có thể dễ dàng biết được URL và HTTP method để quản lý một resource. Bạn cũng cần lưu ý bản thân RESTful không quy định logic code ứng dụng và RESTful cũng không giới hạn bởi ngôn ngữ lập trình ứng dụng. Bất kỳ ngôn ngữ lập trình (hoặc framework) nào cũng có thể áp dụng RESTful trong việc thiết kế API cho ứng dụng web.

## *2.5. Các thư viện hỗ trợ trong quá trình xây dựng ứng dụng*

* + 1. Butter knife

Thư viện Android ButterKnife là một thư viện view injection nó tác động đến các view trong activity, fragment bằng việc sử dụng annotations. Ví dụ: annotation [@BindView](https://viblo.asia/u/BindView) được dùng để tránh sử dụng phương thức findViewById().Butterknife cung cấp rất nhiều các tùy chọn hữu ích khác như binding strings, dimens, drawables, click events...giúp chúng ta viết code ngắn gọn hơn, dễ nhìn hơn, không bị lặp code.

* + 1. Glide

Glide là một thư viện open source hỗ trợ load ảnh trên Android. Dùng Glide sẽ đơn giản hóa các công việc mà bạn cần làm khi sử dụng một bức ảnh trong Android đi rất nhiều. Chúng ta không cần quan tâm đến việc decoding, memory and disk caching mà thay vào đó chỉ cần sử dụng interface rất đơn giản từ Glide.

Glide hỗ trợ fetching, decoding và hiển thị cả ảnh tĩnh hoặc ảnh động đó là điểm nổi bật của nó so với các thư viện load ảnh khác cho android. Glide có thể được dễ dàng đưa vào bất kỳ mô hình mạng nào.

* + 1. Retrofit

[Retrofit](https://square.github.io/retrofit/) là một HTTP client type-safe cho Android và Java. Retrofit giúp dễ dàng kết nối đến một dịch vụ REST trên web bằng cách chyển đổi API thành Java Interface. Trong bài này, tôi sẽ hướng dẫn bạn cách sử dụng một trong những thư viện HTTP phổ biến nhất và được khuyến dùng nhất cho Android.

Thư viện mạnh mẽ này giúp bạn dễ dàng xử lý dữ liệu JSON hoặc XML sau đó phân tích cú pháp thành Plain Old Java Objects (POJOs). Tất cả các yêu cầu GET, POST, PUT, PATCH và DELETE đều có thể được thực thi.

Giống như hầu hết các phần mềm mã nguồn mở khác, Retrofit được xây dựng dựa trên một số thư viện mạnh mẽ và công cụ khác. Đằng sau nó, Retrofit làm cho việc sử dụng [OkHttp](http://square.github.io/okhttp/) (từ cùng một nhà phát triển) để xử lý các yêu cầu trên mạng. Ngoài ra, Retrofit không tích hợp bất kỳ một bộ chuyển đổi JSON nào để phân tích từ JSON thành các đối tượng Java.

Việc phát triển thư viện HTTP type-safe của chính bạn để giao tiếp với REST API có thể thật sự là một điều khó khăn: bạn phải xử lý nhiều chức năng chẳng hạn như tạo các kết nối, lưu trữ cach, thử lại các yêu cầu thất bại, phân tích luồng phản hồi, xử lý lỗi, và nhiều hơn nữa. Retrofit, mặt khác, được hoạch định rất tốt, có tài liệu hướng dẫn và kiểm đã được kiểm thử—một thư viện đã được thử nghiệm rất nhiều trong thực tế sẽ giúp bạn tiết kiệm rất nhiều thời gian quý báu và giảm bớt căng thẳng.

* + 1. Dagger

Dependency Injection giống như việc di chuyển vào một ngôinhà đã có đầy đủ đồ đạc nội thất. Ta không cần chuẩn bị mà chỉ việc dùng luônnhững gì cái sẵn có.

Nó cung cấp một đối tượng mà bạn yêu cầu khi bạn muốn khởi tạonhanh một đối tượng mới, các đối tượng mới không cần xây dựng hay tuỳ chỉnh nhữngthành phần con của nó.

Trong Android, đôi khi ta cần truy cập đến một đối tượng phứctạp từ nhiều điểm trong ứng dụng, ví dụ như network client, image loader, SharedPreferenceshay local storage. Ta có thể inject những đối tượng đó vào Activity hoặc Fragmentvà truy cập chúng ngay lập tức khi cần.

* + 1. MVP?

Khi chúng ta đang phát triển một ứng dụng phức tạp, chúng ta thường bắt gặp những thách thức mà có lẽ đã được giải quyết trước đó và đã có một số giải pháp khá tuyệt vời.

Các giải pháp như vậy thường được gọi là patterns. Chúng ta thường nói về design patterns và architectural patterns. Chúng đơn giản hóa phát triển và nên sử dụng chúng bất cứ khi nào thích hợp.

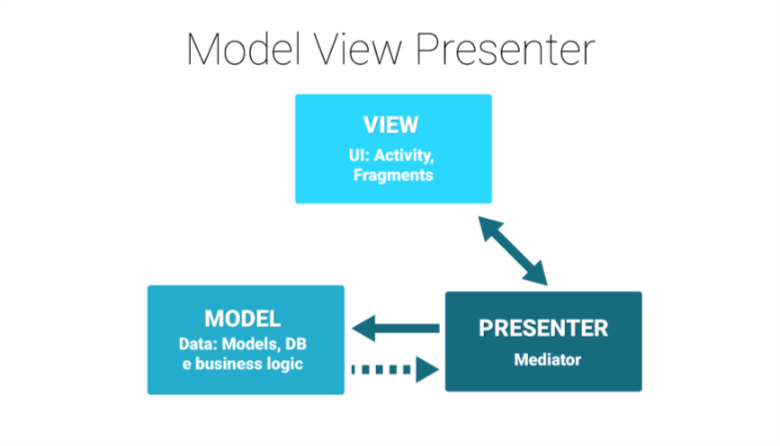
* + 1. Kiến trúc Android

Các thiết kế của một dự án là mối quan tâm ngay từ đầu. Một trong những điều đầu tiên chúng ta nên xem xét là kiến trúc mà chúng ta dự định áp dụng vì nó sẽ xác định cách các yếu tố khác nhau của ứng dụng liên quan đến nhau như thế nào. Nó cũng sẽ thiết lập một số nguyên tắc cơ bản để hướng dẫn chúng ta trong phát triển.

Nói chung, một framework hay SDK hy vọng mọi việc được thực hiện một cách nhất định, nhưng điều đó không phải luôn luôn là một trong những quyền cho một dự án. Đôi khi, không có cách nào xác định trước hoặc đúng làm việc, để lại các quyết định thiết kế lên đến các nhà phát triển. Android SDK hy vọng mọi việc được thực hiện một cách nhất định, nhưng đó không phải là luôn luôn đủ hoặc sự lựa chọn tốt nhất.

Mặc dù Android cung cấp một SDK xuất sắc, mô hình kiến trúc của nó là khá bất thường và có thể dễ dàng có được theo cách của bạn trong thời gian phát triển, đặc biệt là khi xây dựng các ứng dụng phức tạp cần phải được kiểm tra và duy trì trong một thời gian dài. May mắn thay, chúng ta có thể chọn từ nhiều giải pháp kiến trúc để giải quyết vấn đề này. MVP là một trong những giải pháp kiến trúc tốt giúp phát triển ứng dụng Android.

Mô hình MVP gồm 3 lớp: Model, View, Presenter. Mỗi một lớp có nhiệm vụ riêng và thông tin liên lạc giữa các lớp được quản lý bởi Presenter



* Model tổ chức business logic của ứng dụng. Nó kiểm soát như thế nào dữ liệu có thể được tạo ra, lưu trữ, và sửa đổi.
* View là một giao diện thụ động hiển thị dữ liệu và các tuyến đường hành động dùng cho Presenter.
* Presenter đóng vai trò trung gian. Nó lấy dữ liệu từ Model và hiển thị lên các View. Nó cũng xử lý hành động người dùng mong muốn được nó bằng các View.
  + 1. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 3.1 Màn hình Splash:

Kiểm tra xem đã đăng nhập trước đó chưa. Nếu đã đăng nhập sẽ gọi API lấy thông tin User và chạy vào màn hình Home. Nếu chưa đăng nhập sẽ chạy vào màn hình Login.

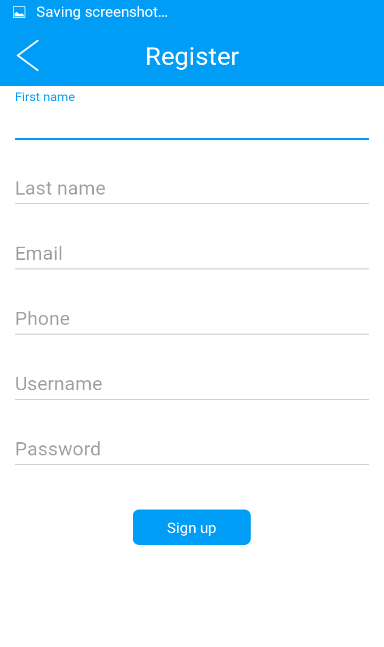
## 3.2 Màn hình Login:

Có 2 trường nhập text là username và password, 2 button Login và Sign up. Nếu chưa có tài khoản thì sẽ chuyển sang màn hình đăng kí để đăng kí tài khoản. Nếu đã có tài khoản thì nhập username và password, thì sau khi nhập username và password nhấn button Login sẽ chuyển tới màn hình Home.



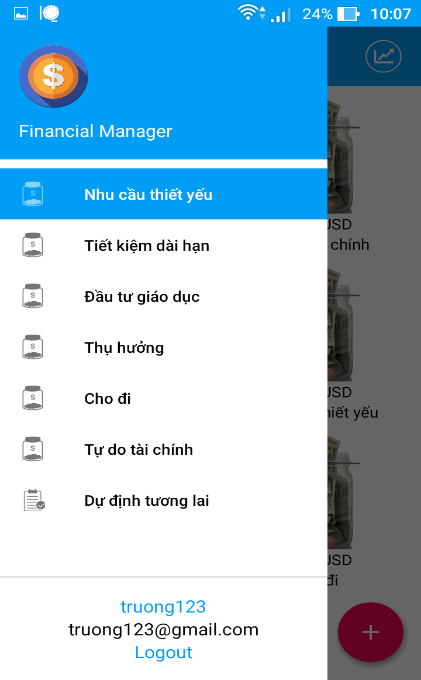
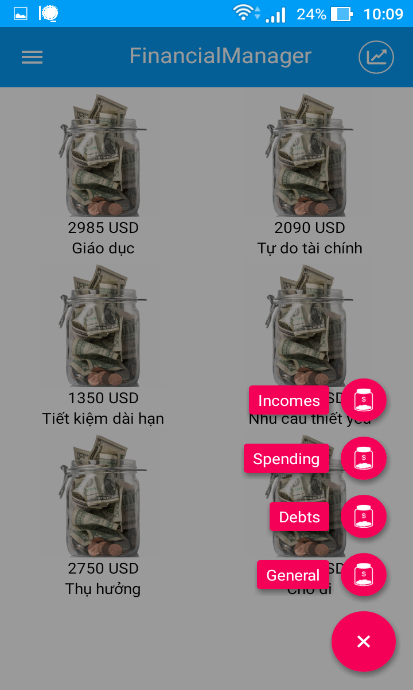
## 3.3 Màn hình đăng kí:

Gồm các trường nhập text: First name, Last name, Email, Phone, Username, Password, icon back và button Sign up. Nếu nhấn icon back thì trở về màn hình Login. Sau khi điền đầy đủ các thông tin nhấn button Sign up để đăng kí tài khoản.



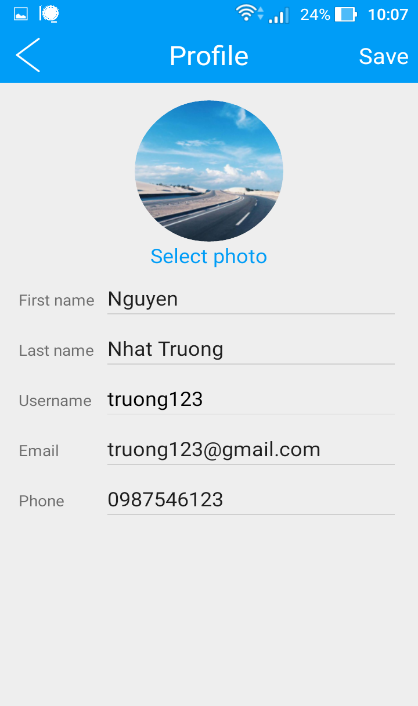
## 3.4 Màn hình Home:

Hiển thị 6 cái lọ, mỗi lọ bao gồm tên và số tiền. Icon nằm trên bên trái nhấn vào sẽ hiển thị Navigation. Icon nằm trên bên phải nhấn vào sẽ chuyển tới màn hình Chart hiển thị khoản thu, chi, nợ của từng tháng trong năm. Button add dùng để thêm chi, tiêu, nợ hoặc nhập tổng tiền.

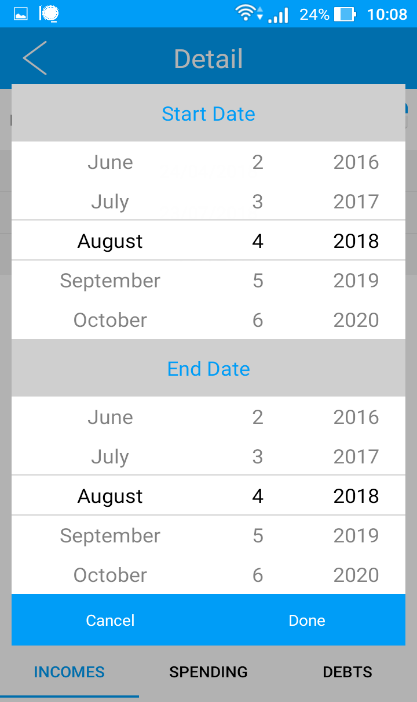
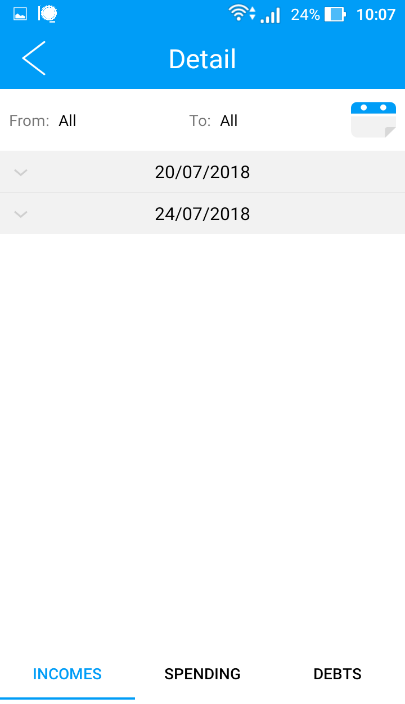
## 3.5 Màn hình Profile:

Gồm ảnh đại diện, First name, Last name, username, email và password. Tại màn hình này có thể cho chỉnh sửa thông tin user như: ảnh đại diện, First name, Last name, email, password.



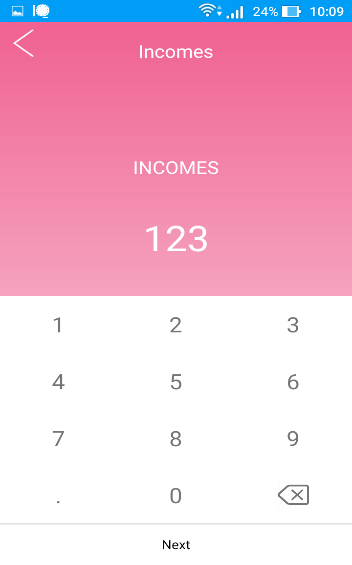
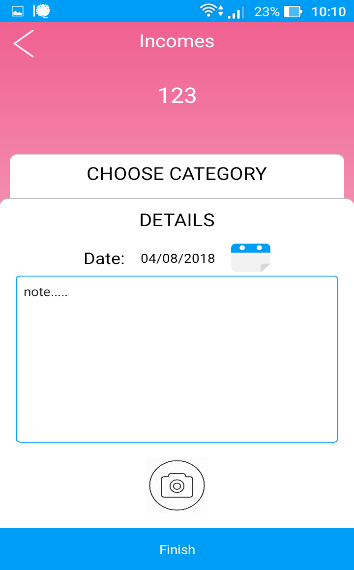
## 3.6 Màn hình Detail:

Gồm 3 tab là: Incomes, Spending, Debts. Icon back để trở về màn hình Home và icon calendar để chọn thời gian để lọc ra danh sách Incomes, Spending, Debts.



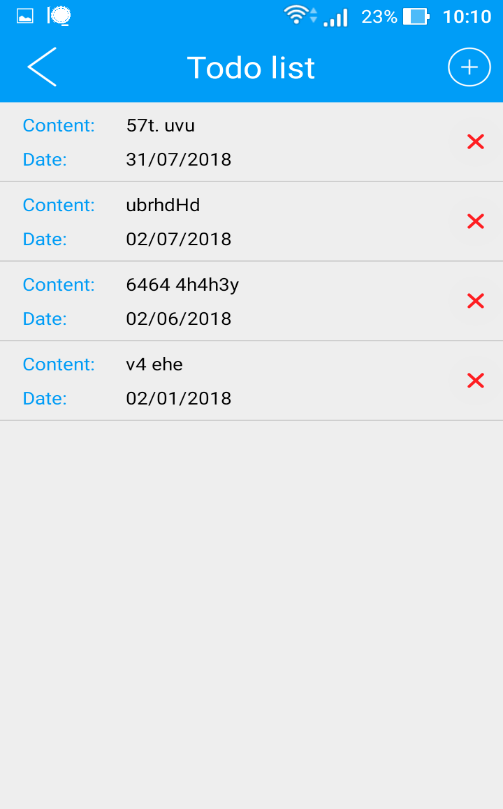
## 3.7 Màn hình tạo các khoản thu, chi, nợ:

Tại màn hình này cho người dùng nhập khoản tiền, ngày tháng năm và nội dung chi tiết.

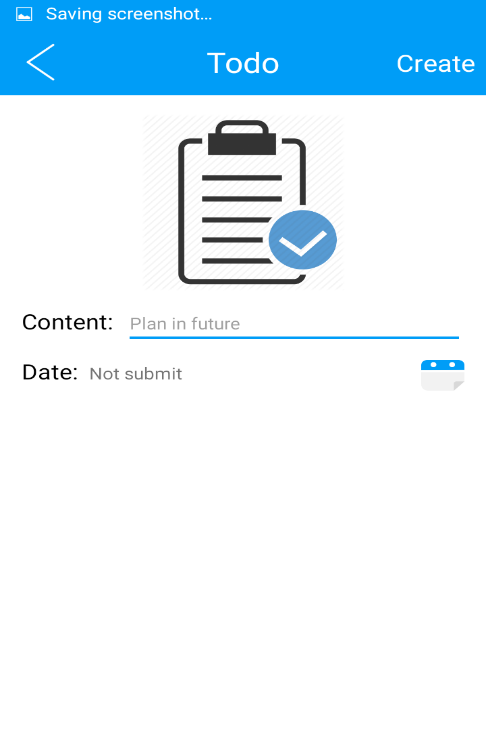
## 3.8 Màn hình Todo List:

Hiện danh sách cách công việc dự định. Có thể cho xóa. Nut back trở về màn hình Home nút Add để thêm mới công việc.



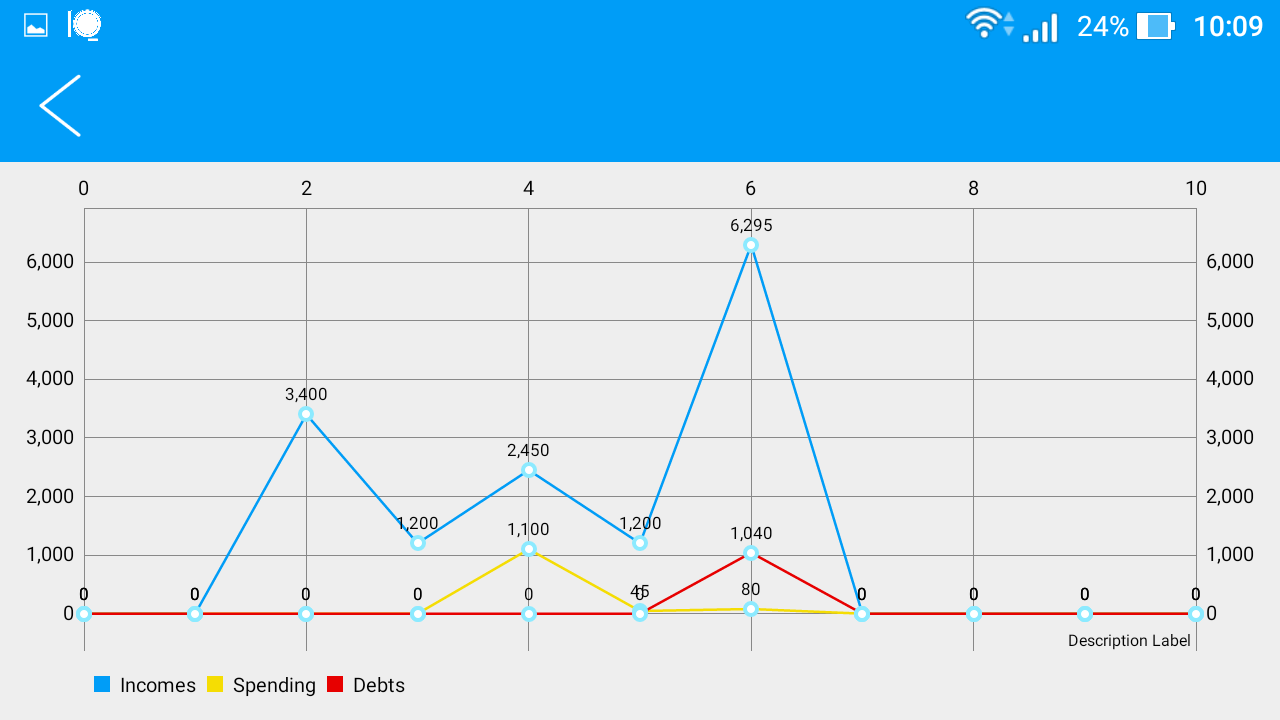
## 3.9 Màn hình tạo todo:

Cho người dùng tạo công việc dự định sẽ làm gồm nội dung và ngày tháng năm.Khi người dùng mở ứng dụng hoặc để chậm ngầm thì sẽ dựa vào thời gian của hệ thống thông báo cho người dùng công việc phải làm tại thời điểm đó.



## 3.10 Màn hình Chart:

Hiển thị biểu đồ khoản tiền thu, chi, nợ theo tháng trong năm để người dùng có thể nhìn lại cái khoản thu, chi, nợ của mình. Từ biểu đồ này người dùng có thể thấy được sự chênh lệch khoản thu, chi, nợ giữa các tháng để có cách chi tiếp theo một cách hợp lí hơn.



# CHƯƠNG IV: CÀI ĐẶT VÀ SỬ DỤNG

## 4.1 Mục tiêu của ứng dụng:

Cung cấp cho người sử dụng một công cụ để quản lý quán tiền tệ và tìm ra cách chi tiêu một cách hợp lí nhất . Thuận tiện cho việc quản lí tiền tệ mà không cần phải quá tốn công ghi nhớ hoặc ghi chép mà chỉ cần một vài thao tác đơn giản. Ứng dụng linh hoạt, thân thiên, dể sử dụng với người dùng.

## 4.2 Phân tích yêu cầu:

### 4.2.1 Yêu cầu chức năng:

**a) Lưu trữ**:

- Lưu các thông tin người dùng: Họ tên, email, số điện thoại, username, ảnh đại diện.

- Lưu dữ liệu từng lọ tiền bao gồm các khoản: Thu, chi và nợ.

- Lưu danh sách các công việc, kế hoạch phải làm.

- Lưu thời gian của các khoản thu, chi, nợ và các công việc, kế hoạch.

**b) Chức năng chính của ứng dụng**:

- Tạo tài khoản bằng cách yêu cầu điền vào những thông tin cần thiết: Họ tên, email, phone, username, password.

- Đăng nhập bằng tài khoản đã đăng kí thành công.

- Đăng xuất tài khoản.

- Chỉnh sửa ảnh thông tin.

- Xem số tiền hiện có cho đến hiện tại của từng chiếc lọ.

- Tạo các khoản thu, chi, nợ.

- Xem chi tiết các khoản thu, chi, nợ của từng chiếc lọ. Có thể lọc danh sách theo ngày, tháng, năm.

- Chỉnh sửa khoản nợ.

- Xem biểu đồ thu, chi, nợ theo từng tháng trong năm.

- Xem danh sách các kế hoạch, công việc đã tạo và khi thời gian các công việc, kế hoạch trùng với thời gian của hệ thống, người dùng được thông báo về công việc, kế hoạch đó.

### 4.2.2 Yêu cầu phi chức năng:

**- Hiệu suất:** giao diện đơn giản, tối ưu hóa code , tổ chức theo mô hình MVP, loại bỏ các dòng code, các import không cần thiết và các cảnh báo trong các file, hạn chế các thể lồng nhau trong file layout xml, áp dụng các kĩ thuật để có thể có hiệu suất tốt nhất.

**- Tính tương thích:** Có thể chạy hầu hết trên các dòng điện thoại Android ở nhiều phiên bản.

**- Tính tin cậy:** Có khả năng tính toán chính xác các dữ liệu đầu vào và dữ liệu đầu ra một cách chính xác. Đảm bảo không sai lệch dữ liệu.

## 4.3 Mô tả chức năng:

### 4.3.1 Đăng nhập:

**-** Người dùng nhập username và password

- Nếu nhấn nút đăng nhập:

+ Kiểm tra username và password có rỗng không. Nếu rỗng thì hiện khung thông báo cho người dùng nhập.

+ Nếu đúng điền đầy đủ username và password, tiến hành gọi API đăng nhập . Nếu failed thì hiện khung thông báo lỗi. Nếu successed thì chuyển tới màn hình Home.

-Nếu nhấn nút đăng kí chuyển tới màn hình Register.

### 4.3.2 Đăng kí:

**-**Người dùng nhập các thông tin cần thiết để đăng kí tài khoản gồm First name, Last name, email, phone, username và password.

- Nhấn nút back trở về màn hình Đăng kí.

- Nhấn nút Sign up để tiến hành đăng kí. Nếu đăng kí thành công hiện ra khung thông báo và chuyển về màn hình Đăng kí, set username và password vừa đăng kí thành công. Nếu đăng kí không thành công hiện khung thông báo lỗi.

### 4.3.3 Chỉnh sửa thông tin người dùng:

**-** Chỉnh sửa First name, Last name, email, phone.

- Nhấn nút back trở vè màn hình Home.

- Nhấn nút chữ Save để lưu lại thông tin vừa chỉnh sửa.

### 4.3.4 Xem tổng quan

- 6 chiếc lọ gồm tên và lượng tiền hiện có sau khi đã tính toán các khoản.

- Nhấn vào icon menu sẽ hiện ra Navigation gôm logo app, tên app, danh mục các chiếc lọ, dự định tương lai, username, email, và log out.

-Nhấn vào icon chart để xem biểu đồ khoản thu, chi , nợ theo từng tháng trong năm.

- Nhấn vào icon add để tạo những khoản thu, chi, nợ cho các chiếc lọ.

- Nhấn vào chiếc lọ để xem chi tiết danh sách thu, chi, nợ.

### 4.3.5 Tạo dữ liệu các khoản thu, chi , nợ

**-** Đối với thu, chi:

+ Nhập vào khoản tiền cần tạo.

+ Nhấn Next để chọn lọ mà ta cần thêm dữ liệu.

+ Nhấn Detail để chọn ngày, tháng, năm tạo và ghi chú nội dụng khoản tiền .

+ Nhấn Finish để hoàn thành tạo khoản tiền thu, chi. Nếu thành công trở về màn hình Home, số tiền của chiếc lọ vừa tạo sẽ được cập nhật lại.

- Đối với nợ:

+ Nhập vào khoản tiền .

+ Nhấn Next để chọn lọ .

+ Nhấn Detail để chọn trạng thái cho khoản nợ.

* Ready: Nợ sẵn sàng trả được tính vào số tiền hiện có.
* Waiting: Nợ đang đợi trả, không được tính vào số tiền hiện có.
* Done: Nợ đã trả, được tính vào số tiền hiện có.
* Positive: Nợ dương, người khác đang nợ mình.
* Negative: Nợ âm, mình nợ người khác.

### 4.3.6 Navigation

- Nhấn vào item lọ sẽ chuyển sang màn hình xem chi tiết.

- Nhấn vào item “Dự định tương lai” sẽ chuyển sang màn hình xem danh sách cách kế hoạch dự định cần làm.

- Nhấn vào username để chuyển sang màn hình cập nhật thông tin người dùng.

- Nhấn log out , một hộp thoại sẽ hiện ra để yêu cầu xác nhận có muốn đăng xuất hay không. Nếu nhấn Cancel thì hộp thoại ẩn đi. Nhấn Ok sẽ trở về màn hình Đăng nhập.

### 4.3.7 Xem chi tiết lọ

- Tab Incomes, hiển thị danh sách khoản thu theo ngày, tháng, năm và nội dung khoản thu.

- Tab Spending hiển thị tương tự bên tab Incomes.

- Tab Debts hiển thị nội dung tương tự như 2 tab Incomes, Spending và có thêm trạng thái khoản nợ và loại nợ là nợ âm hay dương. Có thể cập nhập hoặc xóa khoản nợ.

- Để chỉnh sửa khoản nợ, người dùng nhấn vào biểu tượng cây bút để chuyển sang màn hình Edit Debt.

- Lọc danh sách các khoản tiền theo ngày, tháng, năm bắt đầu và ngày, tháng, nắm kết thúc.

**4.3.8 Chỉnh sửa khoản nợ**

- Người dùng cập nhập ngày, tháng, năm, nội dung, khoản nợ, trạng thái, và nợ âm hay dương.

- Nợ Waiting, khi đã có tiền sẵn sàng trả người dùng sẽ cập nhập sang trạng thái là Ready.

- Nợ Ready, đã trả nợ thì sẽ cập nhật sang trạng thái Done.

- Nhấn back để trở về màn hình xem chi tiết.

- Nhấn Save để cập nhật lại khoản nợ. Nếu success sẽ cập nhật lại khoản nợ vừa chỉnh sửa, nếu failed sẽ hiển thị thông báo lỗi và trở về màn hình xem chi tiết.

### 4.3.9 Biểu đồ

**-** Hiển thị lượng tiền tăng giảm theo các tháng trong năm. Đường màu xanh hiển thị cho khoản thu. Đường mà vàng hiển thị cho khoản chi. Đường màu đỏ hiển thị cho khoản nợ. Từ biểu đồ này người dùng có cái nhìn tổng quan hơn về các khoản tiền của mình trong các tháng.

- Nhấn nút back để trở về màn hình Home.

### 4.3.10 Dự định, công việc phải làm

**-** Hiển thị danh sách công viêc, dự định sẽ làm. Có thể xóa bằng cách nhấn vào dấu X màu đỏ, hộp thoại xác nhận xóa hiện ra, nếu nhấn OK xóa, nhấn Cancel ẩn thông báo.

- Nhấn vào nút back trở về màn hình Home.

- Nhấn vào nút Add chuyển sang màn hình tạo mới.

- Ở màn hình tạo mới, người dùng nhập vào nội dung và chọn ngày, tháng, năm. Nhấn Create để tạo và trở về màn hình xem danh sách công việc, dự định.

- Mục đích tạo các công việc, dự định này để khi người dùng sử dụng app hoặc để app chạy ngầm thì sẽ lấy thời gian của hệ thống để thông báo công việc, dự định cho người dùng biết.

# KẾT LUẬN

* Hạn chế:
* Chỉ dùng cho cá nhân**.**
* Chưa nhiều tính năng.
* Chưa có tính bảo mật.
* Hình ảnh còn đơn giản chưa sinh động, bắt mắt.
* Kết quả đạt được:
* Giao diện đơn giản, dễ dùng.
* Ứng dụng đảm bảo được các yêu cầu cơ bản như quản lý tiền tệ, cho người dùng xem thống kê lượng tiền chi, tiêu, nợ qua các tháng, thông báo các công việc, kế hoạch cần làm trong ngày.
* Hướng phát triển:
* Phát triển thêm nhiều tính năng mới.
* Hỗ trợ giao diện đa ngôn ngữ.
* Nâng cấp tính bảo mật cho cho ứng dụng.

# DANH MỤC TÀI LIỆU THAM KHẢO

Danh mục các website tham khảo:

1. http://jakewharton.github.io/butterknife/

2. https://www.codehub.vn/RESTful-API-Cho-Nguoi-Bat-Dau

3. https://code.tutsplus.com/vi/tutorials/getting-started-with-retrofit-2--cms-27792

4. https://viblo.asia/p/gioi-thieu-model-view-presenter-trong-android-K1ov16JkR5b9