**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

--------------------------------



**BÁO CÁO THỰC TẬP TỐT NGHIỆP**

**(tại Doanh nghiệp**)

**Sinh viên thực hiện:** Thào A Vảng

**Lớp:** Kỹ thuật phần mềm K15A

**Người hướng dẫn tại doanh nghiệp:** Dương Văn Thắng

**Giáo viên quản lý:** ThS. Đỗ Đình Lực

**Hà Nội – Năm 2021**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

--------------------------------



**BÁO CÁO THỰC TẬP TỐT NGHIỆP**

**(tại doanh nghiệp)**

Sinh viên thực hiện: Thào A Vảng

Lớp: Kỹ thuật phần mềm K15A

**Người hướng dẫn tại Doanh nghiệp** **Giáo viên quản lý**

*(Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên)*

Hà Nội – 2021

MỤC LỤC

[MỤC LỤC 3](#_Toc61819474)

[LỜI CẢM ƠN 4](#_Toc61819475)

[LỊCH LÀM VIỆC TẠI NƠI THỰC TẬP 5](#_Toc61819476)

[CHƯƠNG I GIỚI THIỆU VỀ TỔ CHỨC TẠI NƠI THỰC TẬP 6](#_Toc61819477)

[1. Quá trình hình thành, phát triển và lĩnh vực hoạt động 6](#_Toc61819478)

[2. Cơ cấu tổ chức 6](#_Toc61819479)

[CHƯƠNG II NỘI DUNG CÔNG VIỆC TÌM HIỂU VÀ THỰC HIỆN 8](#_Toc61819480)

[I. Nội dung nhiệm vụ chính được giao 8](#_Toc61819481)

[II. Nội dung chi tiết 8](#_Toc61819482)

[1. Tìm hiểu về Công ty 8](#_Toc61819483)

[2. Ôn lại các kiến thức về lập trình Android 8](#_Toc61819484)

[3. Xây dựng demo ứng dụng thời tiết 11](#_Toc61819485)

[4. Traning các kiến thức cơ bản về Android 23](#_Toc61819486)

[5. Training về git, github 42](#_Toc61819487)

[CHƯƠNG III KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 46](#_Toc61819488)

[I. Kết luận 46](#_Toc61819489)

[1. Kỹ năng mềm 46](#_Toc61819490)

[2. Áp dụng thực tiễn 47](#_Toc61819491)

[3. Hạn chế 47](#_Toc61819492)

[II. Hướng phát triển 47](#_Toc61819493)

[TÀI LIỆU THAM KHẢO 48](#_Toc61819494)

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 49](#_Toc61819495)

# LỜI CẢM ƠN

Trong suốt quá trình học tập, rèn luyện kỹ năng trên ghế nhà trường và sau thời gian 2 tháng thực tập thực tế tại Công Ty Cổ Phần Quốc Tế TDA Việt Nam. Mặc dù thời gian không dài nhưng đã giúp em học hỏi được nhiều kinh nghiệm thực tế hài hòa giữa kiến thức lý thuyết vào trong quá trình thực hành. Với sự giúp đỡ, quan tâm tận tụy của các thầy, cô giáo, cùng các anh chị nhân viên trong công ty đã giúp em học hỏi được nhiều kinh nghiệm quý báu khi ra trường.

Để có được những kiến thức quý báu đó, em xin gửi lời cảm ơn chân thành đến các thầy cô giáo Trường Đại học Công nghệ thông tin và truyền thông Thái Nguyên nói chung, các thầy cô giáo trong Bộ môn Công nghệ phần mềm nói riêng, giáo viên quản lý – Thạc sỹ Đỗ Đình Lực đã hết lòng tận tụy, chỉ bảo, hướng dẫn chúng em trong suốt quá tình học tập, gắn liền kiến thức lý thuyết giảng dạy, cũng như hiểu biết về thực tế trong chuyên ngành để khi chúng em ra trường nắm chắc được những kiến thức quan trọng, chuẩn bị cho tương lai sau này.

Bên cạnh đó, với sự giúp đỡ, chỉ bảo tận tình của Công Ty Cổ Phần Quốc Tế TDA Việt Nam nói chung, anh Dương Văn Thắng, anh Vũ Anh Đức và các anh chị trong team Android nói riêng, đã chỉ bảo tận tình, tạo mọi điều kiện thuận lợi để em thực hiện tốt đợt thực tập vừa qua. Với lòng biết ơn chân thành, em xin chúc toàn thể anh chị trong Công ty luôn có một sức khỏe tốt, một tâm thế chiến binh để có thể hoàn thành tốt mọi nhiệm vụ được giao.

Trong quá trình thực tập và làm báo cáo, do còn nhiều thiếu sót về mặt kinh nghiệm thực tế nên em rất mong nhận được sự đóng góp và chỉ bảo từ các thầy cô giáo để em có thể hoàn thành đợt thực tập một cách tốt hơn.

Em xin chân thành cảm ơn!

Hà Nội, ngày 18 tháng 01 năm 2021

Sinh viên

**Thào A Vảng**

# LỊCH LÀM VIỆC TẠI NƠI THỰC TẬP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tuần** | **Công việc** | **Người hướng dẫn** | **Mức độ hoàn thành** | **Nhận xét của người hướng dẫn** |
| 11 | - Tìm hiểu về công ty, cách tổ chức của công ty.  - Làm quen với các công cụ làm việc trong công ty.  - Học cách trao đổi, báo cáo, làm việc qua email. | Dương Văn Thắng |  |  |
| 2 | Ôn lại các kiến thức cơ bản về Android | Dương Văn Thắng |  |  |
| 3 | Xây dựng demo ứng dụng thời tiết theo kiến trúc tự chọn | Dương Văn Thắng |  |  |
| 4 | Xây dựng demo ứng dụng thời tiết theo mô hình MVVM | Dương Văn Thắng |  |  |
| 5 | Tìm hiểu về Activity trong Android | Dương Văn Thắng |  |  |
| 6 | Tìm hiểu về User Interface trong Android | Dương Văn Thắng |  |  |
| 7 | Tìm hiểu về DataView trong Android | Dương Văn Thắng |  |  |
| 8 | - Tìm hiểu và làm việc với git, github  - Báo cáo cuối đợt thực tập | Dương Văn Thắng |  |  |

# CHƯƠNG I GIỚI THIỆU VỀ TỔ CHỨC TẠI NƠI THỰC TẬP

## **1.** **Quá trình hình thành, phát triển và lĩnh vực hoạt động**

**TDA Việt Nam** có tên đầy đủ là Công Ty Cổ Phần Quốc Tế TDA Việt Namlà chi nhánh của **TDA Corporation** có trụ sở tại Tokyo Nhật Bản. Thị trường chính là khách hàng Nhật Bản.

TDA Corporation có địa chỉ tại Tòa nhà 7F RBM Toranomon, Toranomon 3-23-6, Minato-ku,Tokyo.

TDA Việt Nam được thành lập đầu năm 2020 bởi 3 thành viên chính là Lê Minh Khôi (CEO), Nguyễn Khánh Linh (COO), Vũ Anh Đức (Manager).

TDA Việt Nam có địa chỉ tại Tầng 8, Tòa nhà MD Complex (MHDI), Số 68 Nguyễn Cơ Thạch, P. Cầu Diễn, Q. Nam Từ Liêm, Hà Nội.

TDA Corporation hoạt động trong lĩnh vực tư vấn giải pháp công nghệ cho các doanh nghiệp vừa và nhỏ tại Nhật Bản. Chuyên tư vấn phát triển hệ thống; Phát triển kiểu phòng thí nghiệm; Phát triển hợp đồng. Bao gồm các mảng như phát triển ứng dụng web, hệ thống thông tin, ứng dụng di động thông minh,…

TDA Corporation nói chung và TDA Việt Nam nói riêng mong muốn được làm việc với các member có chung suy nghĩ và định hướng theo phương châm hoạt động của công ty, bao gồm: Nhiệt huyết, Chân thành, Minh bạch, Đồng cảm, Challenge, Chủ động.

## **2. Cơ cấu tổ chức**

*Danh sách các chi nhánh:*

|  |  |  |  |
| --- | --- | --- | --- |
| **Đơn vị** | **Tên đầy đủ** | **Văn phòng** | **Số lượng nhân viên** |
| TDA Corporation | Tập đoàn TDA | **Nhật Bản:** Tòa nhà 7F RBM Toranomon, Toranomon 3-23-6, Minato-ku,Tokyo. | 100 |
| TDA Việt Nam | Công Ty Cổ Phần Quốc Tế TDA Việt Nam | Tầng 8, Tòa nhà MD Complex (MHDI), Số 68 Nguyễn Cơ Thạch, P. Cầu Diễn, Q. Nam Từ Liêm, Hà Nội, Việt Nam | 45 |

**CHƯƠNG II  
NỘI DUNG CÔNG VIỆC TÌM HIỂU VÀ THỰC HIỆN**

## **I. Nội dung nhiệm vụ chính được giao**

- Tìm hiểu về công ty và các quy định

- Ôn lại các kiến thức về lập trình Android

- Xây dựng demo ứng dụng thời tiết.

- Training các kiến thức cơ bản về Android.

- Training về git, github

## **II. Nội dung chi tiết**

### 1. Tìm hiểu về Công ty

**-** Tìm hiểu về công ty, cách tổ chức của công ty.

- Làm quen với các công cụ làm việc trong công ty.

- Học cách trao đổi, báo cáo, làm việc qua email.

### 2. Ôn lại các kiến thức về lập trình Android

**a) Lập trình Android là gì?**

**Android** là hệ điều hành trên thiết bị di động được phát triển bởi Google và dựa trên nền tảng Linux.

**Lập trình Android** là lập trình, tạo ra các ứng dụng, phần mềm trên thiết bị Android, đây là một lĩnh vực lập trình phổ biến.

Android được ra mắt vào ngày 05 tháng 11 năm 2007 gắn với sự thành lập của liên minh thiết bị cầm tay mã nguồn mở, bao gồm 78 công ty phần mềm, phần cứng và viễn thông nhằm mục đích tạo ra một chuẩn mở cho điện thoại di động trong tương lai.

**b) Một số ngôn ngữ lập trình dùng để lập trình Android**

**Một số ngôn ngữ lập trình phổ biến để lập trình Android:**

* **Java :** là ngôn ngữ đơn giản dễ sử dụng, độ linh hoạt cao, đặc biệt là trong Android Studio IDE để phát triển các ứng dụng dành cho Android.
* **Kotlin:** là ngôn ngữ lập trình kiểu tĩnh chạy trên máy ảo Java, được phát triển bởi JetBrains.
* **C#:** là ngôn ngữ lập trình được phát triển từ Java và đã được loại bỏ những nhược điểm của Java trước đó, đây là ngôn ngữ hướng đối tượng, đơn giản và hiện đại, hướng nhiều mục đích sử dụng.
* **Flutter:** là mobile UI framework của Google để tạo ra các giao diện native chất lượng cao trên iOS và Android. Flutter sử dụng ngôn ngữ Dart, là ngôn ngữ hướng đối tượng, được xác định theo lớp.
* **C++:** được xem là ngôn ngữ mạnh mẽ nhất khi xây dựng ứng dụng Android và được sử dụng cho lập trình cấp thấp.

**c) Các mô hình kiến trúc trong ứng dụng Android**

Một ứng dụng Android có thể được xây dựng theo một mô hình kiến trúc cụ thể, tuỳ thuộc vào mục đích phát triển, một số vấn đề cần quan tâm như: tính dễ bảo trì, tính bảo mật, tính kiểm thử ,nâng cấp,…

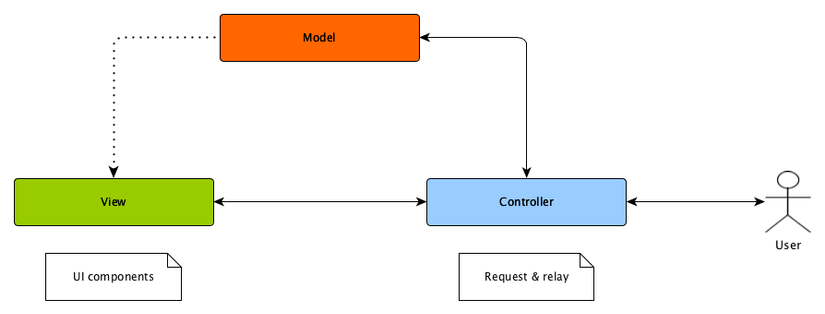
Dưới đây là một số mô hình phổ biến:

+ Model View Controller (MVC)

+ Model View Presenter (MVP)

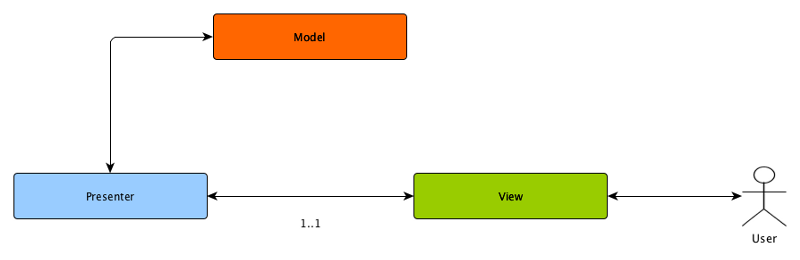
+ Model View ViewModel (MVVM)

**MVC design parttern** chia một đối tượng đồ họa (UI Component) bao gồm 3 thành phần cơ bản: Model, View, và Controller. Model có trách nhiệm đối với toàn bộ dữ liệu cũng như trạng thái của đối tượng đồ họa. View chính là thể hiện trực quan của Model, hay nói cách khác chính là giao diện của đối tượng đồ họa. Và Controller điều khiển việc tương tác giữa đối tượng đồ họa với người sử dụng cũng như những đối tượng khác.



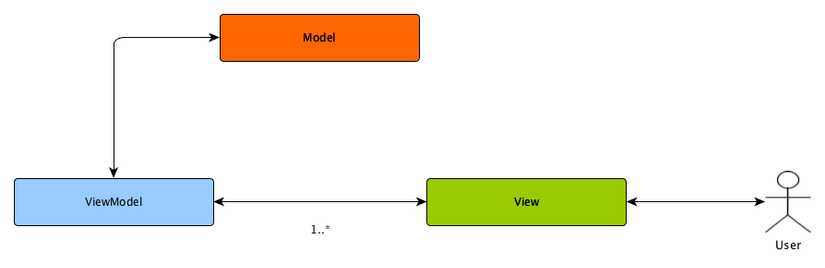
*Hình 1. Mô hình MVC*

**MVP parttern** tương tự như MVC parttern. Tầng Controller của MVC được thay thế bởi tầng Presenter. Mẫu parttern này được chia thành 3 tầng : Model, View, Presenter.



*Hình 2. Mô hình MVP*

**MVVM** hỗ trợ kiểu [Two-way Databinding](http://www.i-programmer.info/news/193-android/8653-android-data-binding-say-goodbye-to-findviewbyid.html) giữa View và Model. Việc này cho phép việc tự động cập nhật các thay đổi hay trạng thái từ ViewModel đến View. Thông thường ViewModel sử dụng [Observer Pattern](http://www.tutorialspoint.com/design_pattern/observer_pattern.htm) để notify các thay đổi trong ViewModel đến Model.



*Hình 3. Mô hình MVVM*

### 3. Xây dựng demo ứng dụng thời tiết

**3.1. Xây dựng ứng dụng thời tiết theo mô hình tự chọn.**

**a) Lựa chọn mô hình, ngôn ngữ, công nghệ.**

- Sử dụng mô hình Model View Controller (MVC).

- Ngôn ngữ : Java

- Cơ dở dữ liệu: Firebase, SharedPreference

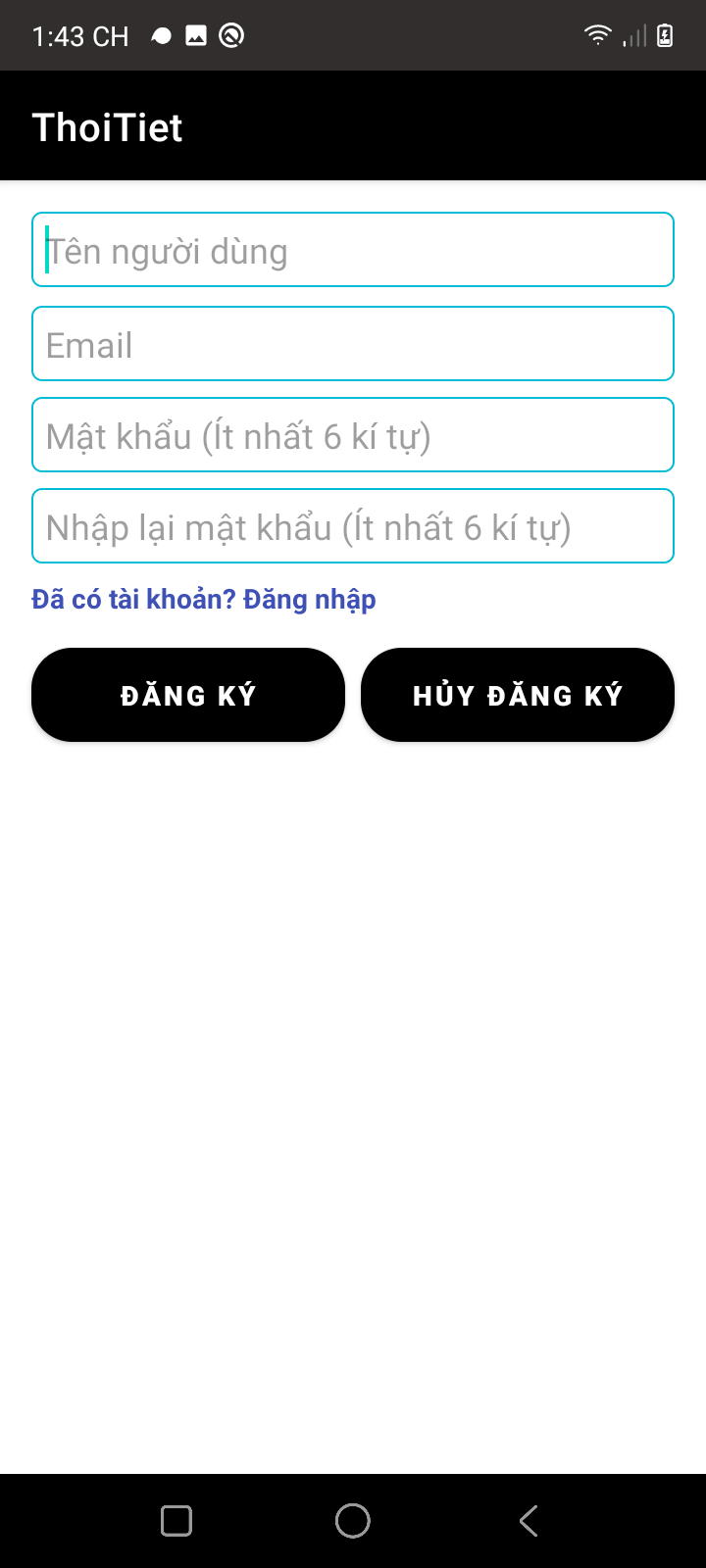
- Network: Retrofit2

- API: Open Weather Map API

- Thư viện: Gson, Retrofit2

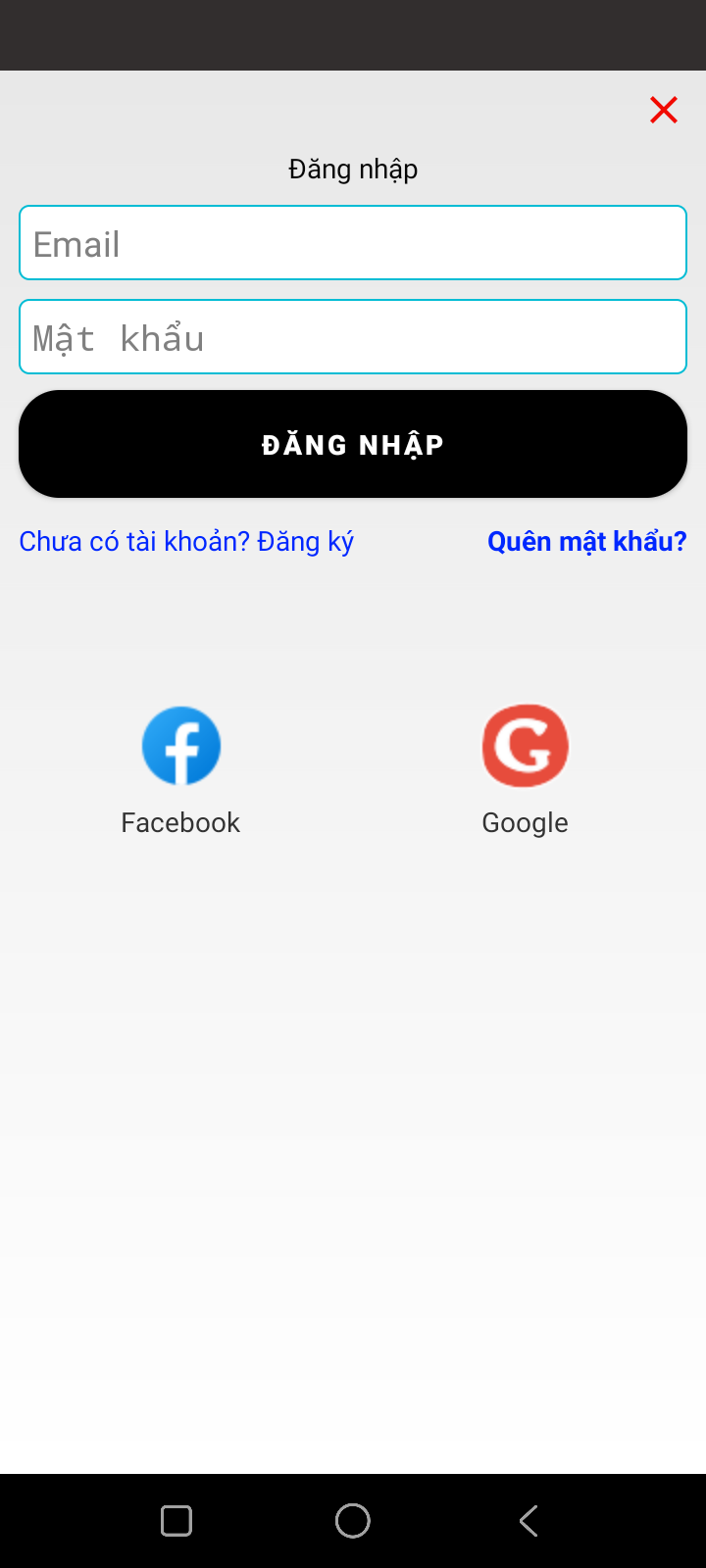
**b) Giao diện ứng dụng**

**- Giao diện đăng ký:**

****

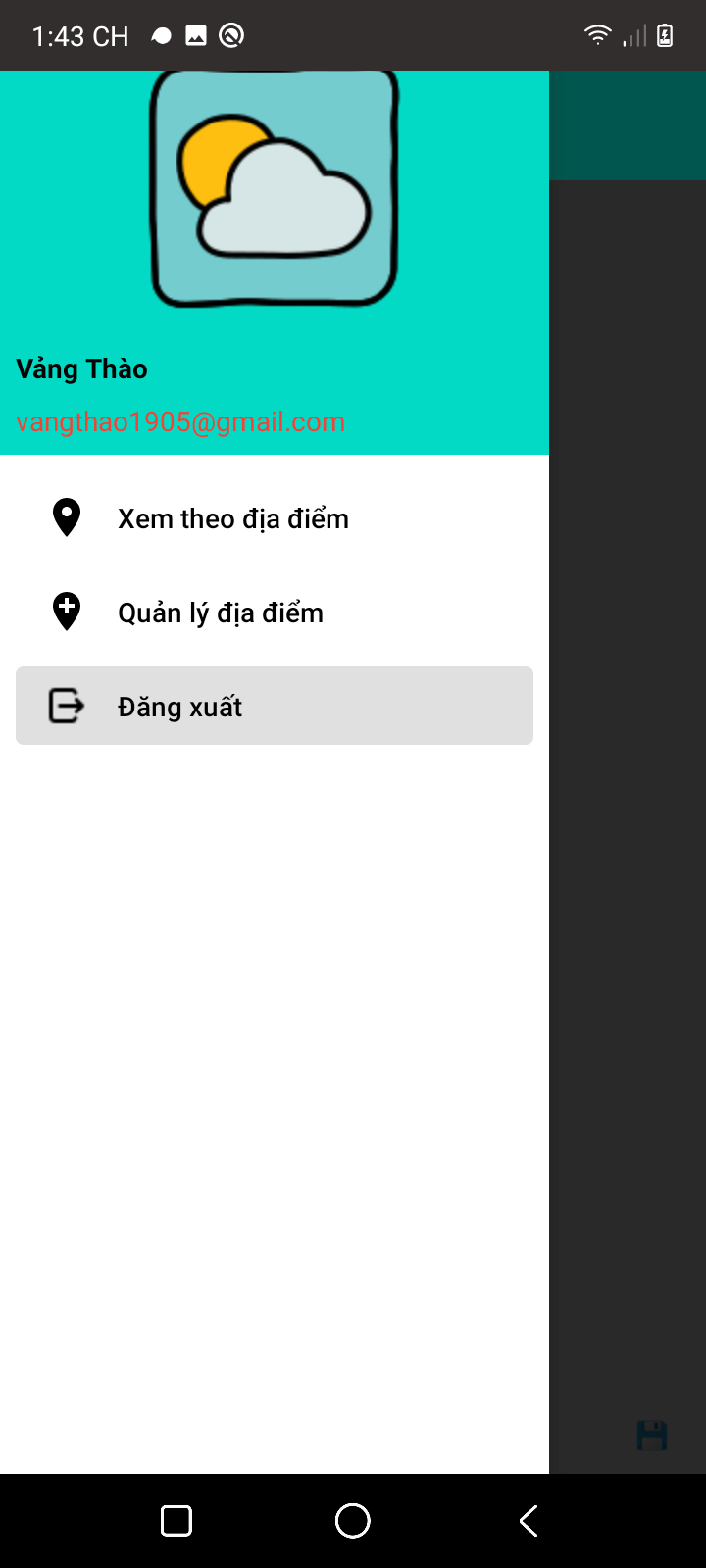
*Hình 4. Giao diện đăng ký theo MVC*

**- Giao diện đăng nhập:**

****

*Hình 5. Giao diện đăng nhập theo MVC*

**- Giao diện màn hình Drawer Natigation:**

****

*Hình 6. Giao diện màn hình Drawer*

**- Giao diện xem thời tiết theo địa điểm:**

****

*Hình 7. Giao diện Xem theo địa điểm*

**- Giao diện trang chủ:**

****

*Hình 8. Giao diện trang chủ*

**3.2 Xây dựng ứng dụng thời tiết theo mô hình MVVM**

**a) Lựa chọn mô hình, ngôn ngữ, công nghệ.**

- Sử dụng mô hình Model View ViewModel (MVVM).

- Ngôn ngữ : Java

- Cơ dở dữ liệu: Firebase, SharedPreference

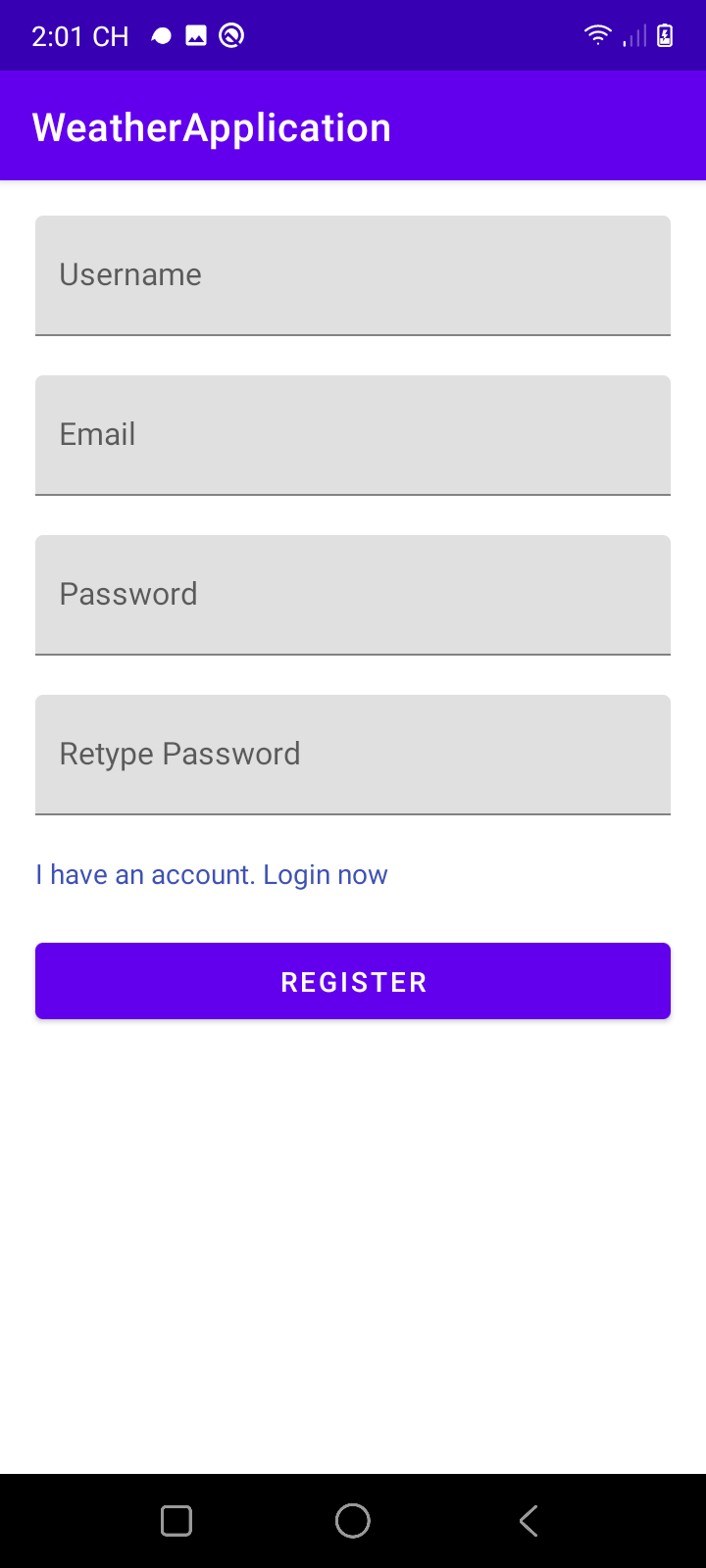
- Network: Retrofit2

- API: Open Weather Map API

- Thư viện: Gson, Retrofit2

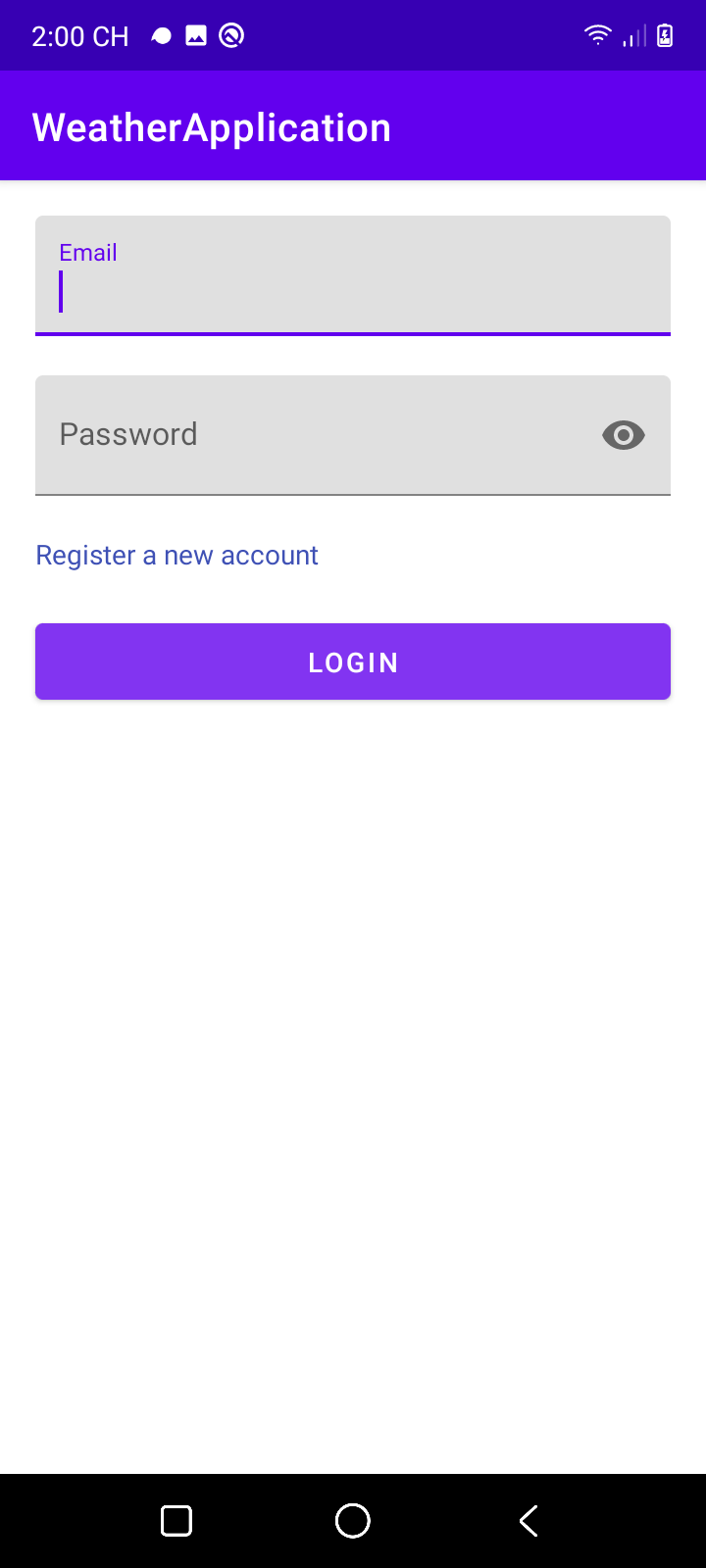
**b) Giao diện ứng dụng**

**- Giao diện đăng ký:**

****

*Hình 9. Giao diện đăng ký theo MVVM*

**- Giao diện đăng nhập:**

****

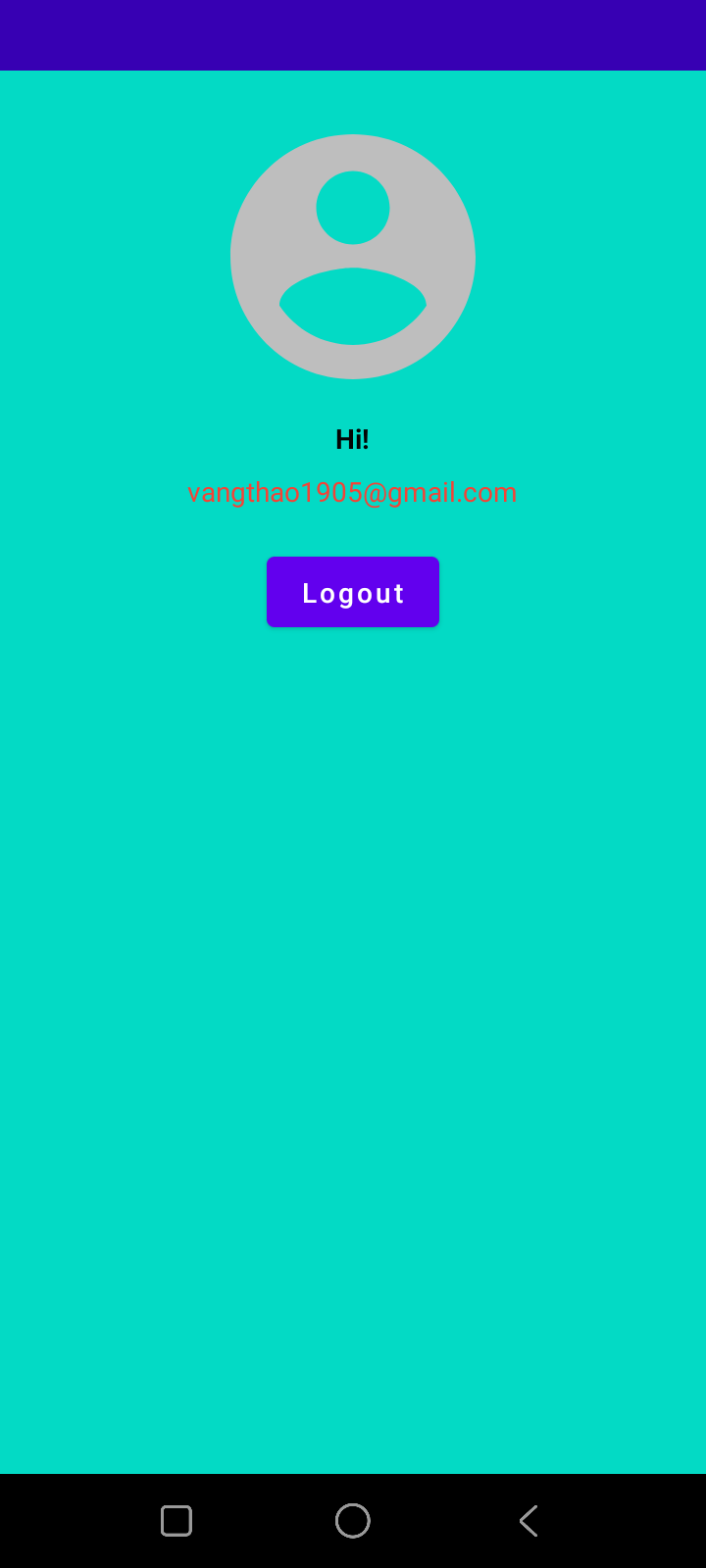
*Hình 10. Giao diện đăng nhập theo MVVM*

**- Giao diện trang chủ:**

****

*Hình 11. Giao diện trang chủ theo MVVM*

**- Giao diện tài khoản của tôi:**

****

*Hình 12. Giao diện tài khoản của tôi*

**- Giao diện quản lý địa điểm:**

****

*Hình 13. Giao diện quản lý địa điểm*

### 4. Traning các kiến thức cơ bản về Android

**4.1. Tìm hiểu về Activity trong Android**

**4.1.1 Activity Lifecycle**

Trong một ứng dụng có nhiều Activity(màn hình). Khi người dùng tương tác thì các Activity sẽ được lưu trữ như một ngăn xếp. Activity đang hiển thị sẽ ở trên cùng của ngăn xếp đó.

Tại một thời điểm bất kỳ, một Activity sẽ ở một trong 4 trạng thái sau:

- **Running**: Activity đang được hiển thị và tương tác với người dùng.

- **Paused**: Activity vẫn đang hiển thị nhưng không còn tương tác với người dùng nữa.

- **Stopped**: Activity không còn hiển thị.

- **Killed**: Activity đã bị giết (killed) bởi hệ thống hoặc là khi hàm finish() được gọi.



*Hình 14. Vòng đời Activity*

**Các hàm của một Activity Lifecycle:**

- onCreate(): Activity đang được tạo nhưng chưa hiển thị với người dùng.

- onStart(): Activity đã được hiển thị nhưng chưa sẵn sàng để người dùng tương tác.

- onResume(): Activity đã được hiển thị, đang ở trên cùng và sẵn sàng để người dùng tương tác.

- onPause(): Activity sắp chuyển bị sáng chế độ background và ngừng tương tác với người dùng.

- onStop(): Activity không còn hiển thị với người dùng.

- onDestroy(): Hàm này có thể được kích hoạt khi finish() được gọi.

- onRestart(): Được gọi khi một Activity đã bị dừng trước khi bắt đầu lại.

**4.1.2 Lưu ý khi sử dụng Activity**

**Khi người dùng Xoay màn hình:**

Android sẽ khởi tạo lại Activity => Tạo nên trải nghiệm không tốt cho người dùng, dữ liệu bị tải lại,...

**Một vài hướng giải quyết:**

- Cấu hình cố định 1 hướng cho Activity(android:screenOrientation)

- Lưu lại trạng thái đối tượng(onSaveInstanceState(Bundle outState))

- Sử dụng fragments(Fragment.setRetainInstance(true))

- Ngăn Activity hủy bằng việc cấu hình android:configChanges

**4.2. Tìm hiểu về User Interface trong Android**

**4.2.1 UI Layout**

****

*Hình 15. UI Layout*

1. **LinearLayout**

**LinearLayout** là một ViewGroup căn chỉnh các View con chứa trong nó theo một hướng nhất định(ngang hoặc dọc).

* Chúng ta có thể chỉ định hướng layout bằng thuộc tính android:orientation.
* Thuộc tính android:gravity để căn chỉnh các View nằm ở vị trí nào trong LinearLayout.
* Chúng ta có thể gán trọng số cho các View con trong LinearLayout bằng thuộc tính android:layout\_weight.
* weigthSum là thuộc tính gắn tổng trọng số của các View

1. **RelativeLayout**

**RelativeLayout** là layout mà nó hiển thị các View con nó chứa ở các vị trí trong mối quan hệ của chúng với nhau. (VD: nằm trên, dưới, trái hoặc phải của View khác)

* View mặc định khi chưa thiết lập mối liên hệ nào sẽ được đặt ở bên trên-trái của RelativeLayout.
* Các View con có thể được dịch chuyển đến một vị trí nhất định trong RelativeLayout thông qua thuộc tính android:gravity.
* Định vị vị trí của các View con thông quan 1 trong các thuộc tính sau kèm với ID của View khác:
* android:layout\_below
* android:layout\_above
* android:layout\_toLeftOf
* android:layout\_toRightOf

1. **ConstraintLayout**

**ConstraintLayout** là một layout được Google giới thiệu lần đầu tại Google I/O 2016.

* ConstraintLayout cho phép các Views kết nối với nhau thông qua các ràng buộc dựa trên mối quan hệ giữa các view khác nhau.
* ConstraintLayout cung cấp các thuộc tính cho phép chúng ta định vị View hiện tại như:
* layout\_constraintTop\_toTopOf
* layout\_constraintTop\_toBottomOf
* layout\_constraintBottom\_toTopOf
* layout\_constraintBottom\_toBottomOf
* layout\_constraintLeft\_toTopOf
* layout\_constraintLeft\_toBottomOf
* Chúng ta có thể kéo thả kết hợp với code xml khi thiết kế một UI bằng ConstraintLayout để tăng hiệu suất thiết kế.

1. **FrameLayout**

**FrameLayout** là một lớp con của ViewGroup.

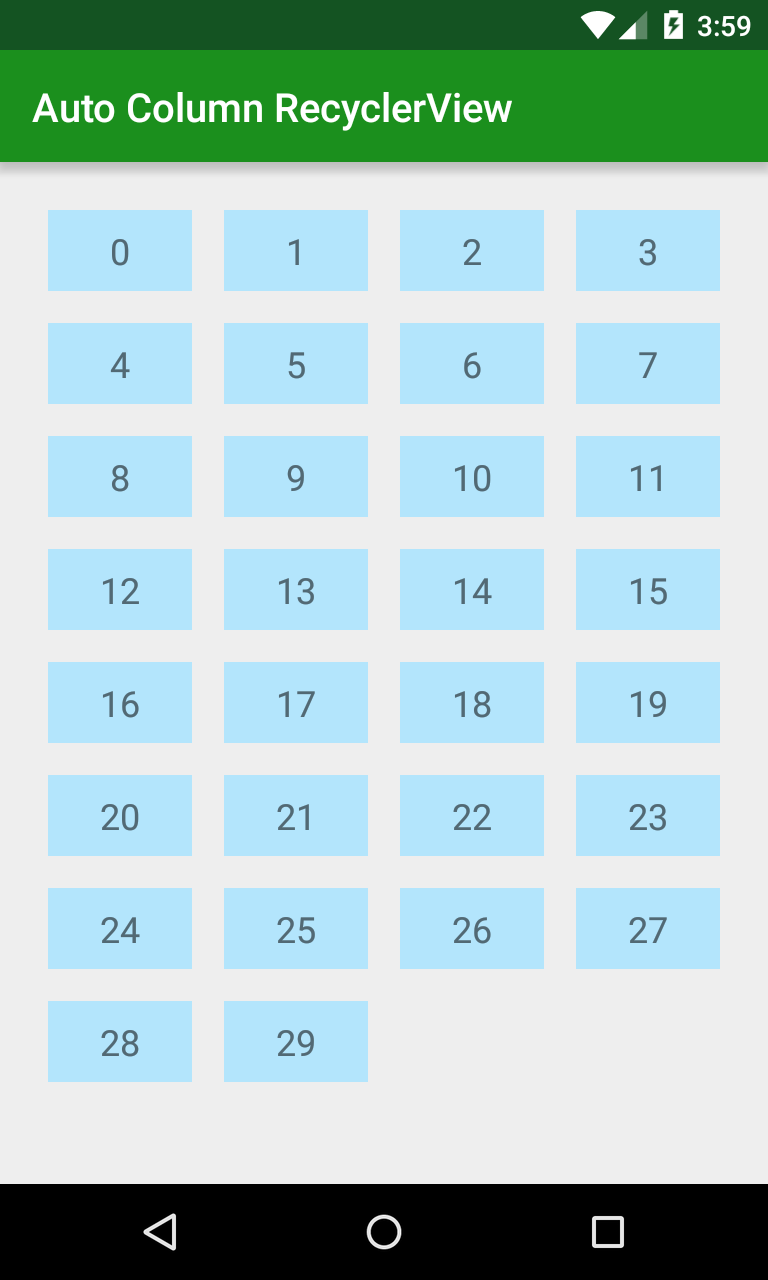
* FrameLayout là loại View cơ sở, loại layout đơn giản nhất.
* Các View chứa bên trong FrameLayout xếp chồng lên nhau.
* Trong FrameLayout các khung nhìn được thêm vào một ngăn xếp(Stack).
* Chúng ta có thể điều chỉnh vị trí của các View con thông qua thuộc tính android:layout\_gravity.

1. **TableLayout**

**TableLayout** là một lớp con của ViewGroup được sử dụng để hiện thị các View trong các hàng và các cột.

* Mỗi hàng là một đối tượng view TableRow, trong TableRow có các view con, mỗi view con này nằm ở vị trí 1 ô bảng(cell).
* TableLayout trong android hoạt động như thẻ table trong HTML.

1. **GridLayout**

****

*Hình 16. GridLayout*

**GridLayout** trong Android là một layout dạng lưới, ta có thể chia các dòng và cột cho cái lưới này.

* Các view sẽ được đặt vào trong các ô trong lưới.
* Chỉ định số dòng tối đa của lưới bởi thuộc tính android:rowCount.
* Chỉ định số cột tối đa của lưới bởi thuộc tính android:columnCount.
* GridLayout là một layout có thể giải quyết được một số vấn đề mà một số layout không giải quyết được như: Khi chúng ta vừa muốn căn chỉnh view hiện tại với view trên(hoặc dưới) nó và view cạnh nó.

**4.2.2 Basic View**

**a)**  **TextView**

**TextView** là một View cho phép hiển thị các chữ(text) trên màn hình.

* TextView có nhiều thuộc tính để chúng ta design theo mong muốn như cỡ chữ, font chữ, màu chữ,...
* Chi tiết một vài thuộc tính hay sử dụng:
* android:text để gán dòng chữ.
* android:textColor để đặt màu chữ.
* android:textSize để thiết lập cỡ chữ.
* android:textStyle để thiết lập định dạng chữ(**bold**, *italic*, normal).
* android:fontFamily dùng để thiết lập font chữ.

- Chúng ta có thể làm việc với TextView thông qua code java bằng việc khai báo biến kiểu TextView và ánh xạ bằng findViewByID(R.id.textViewId) hoặc sử dụng DataBinding.

**b) EditText**

**EditText** là một loại View hiển thị một hộp cho phép người dùng nhập dữ liệu.

* EditText là loại View mở rộng chứ năng từ TextView nên một số thuộc tính như cỡ chữ,font chữ, màu chữ,...sử dụng tương tự nhau.
* Một vài thuộc tính hay sử dụng:
* android:inputType dùng để thiết lập kiểu nhập liệu, hiện thị.
* android:maxLines giới hạn dòng chữ nhập trên một dòng.
* android:digits giới hạn loại ký tự số được nhập vào (VD android:digits="123" chấp nhận số 1,2,3).
* android:drawableLeft thiết lập Drawble vào bên trái của EditText,...
* Nhận thông tin khi đang nhập liệu bằng cách sử dụng method addTextChangedListener với tham số là một đối tương TextWatcher.
* Một số method được override như: beforeTextChanged,onTextChanged,afterTextChanged
* Chúng ta có thể sử dụng TextInputLayout và TextInputEditText được cung cấp bởi thư viện Support Design.

**c) Button**

**Button** là một loại View, nó hiện thị nút bấm để chờ người dùng bấm vào.

* Button được kế thừa từ TextView do vậy các thuộc tính, thiết lập đượp áp dụng tương tự như TextView.
* Chúng ta có thể bắt sự kiện click cho button bằng 2 cách:
* Gọi hàm setOnClickListener trong code.
* Dùng thuộc tính android:onClick và gán cho một method để xử lý sự kiện khi được click.
* Chúng ta có thể custom button riêng theo ý muốn bằng cách design một file .xml trong drawble và gọi file này thông qua thuộc tính android:background.

**d) Checkbox**

**CheckBox** mở rộng từ CompoundButton là một loại Button cho phép hiện thị hộp kiểm, hiện thị thông tin trạng thái checked hay unchecked.

* Trong XML:
* android:checked thiết lập trạng thái checked, unchecked với giá trị "true" hay "false".
* android:button để gán Drawable vào View.
* android:buttonTint để gán màu Tint.
* Trong code Java:
* isChecked() để kiểm tra trạng thái là checked (true) hay unchecked (false).
* setOnCheckedChangeListener bắt sự kiện khi trạng thái (checked/unchecked) chuyển đổi.
* CheckBox có các thuộc tính tương tự như TextView, Button

**e) RadioButton**

**RadioButton** là loại control biểu diễn trạng thái checked/unchecked.

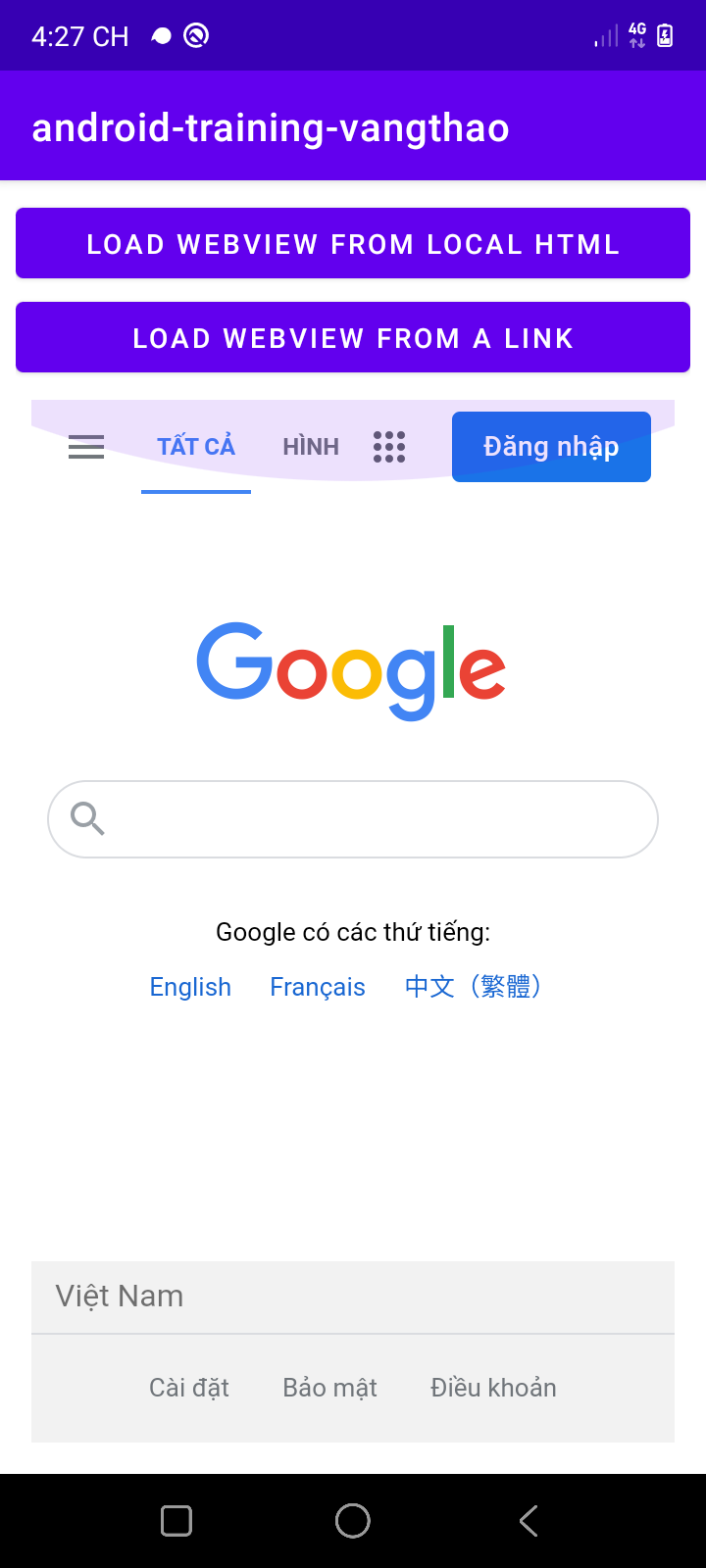
* Thông thường RadioButton được sử dụng với RadioGroup, khi đó có vài RadioButton trong một RadioGroup người dùng sẽ chỉ có thể chọn một RadioButton tại một thời điểm.
* RadioButton cũng có một số thuộc tính tương tự như TextView, CheckBox.
* Một vài thuộc tính:
* android:checked="true" thiết lập cho một RadioButton được mặc định là checked.
* android:orientation thay đổi layout RadioGroup hiện thị các RadioButton trong nó theo hướng ngang hay dọc.

**f) ImageView**

**ImageView** là loại View dùng để hiện thị các tài nguyên hình ảnh như ảnh Bitmap, ảnh Drawbles.

* Tài nguyên ảnh cần hiện thị được gán vào thuộc tính android:src hoặc dùng 1 trong các method phù hợp sau:
* setImageBitmap(Bitmap bm)
* .setImageDrawable(Drawable drawable).
* setImageResource(int resId).
* setImageURI(Uri uri).
* Thuộc tính android:scaleType dùng để thiết lập thu phóng ảnh(fitXY,center,...).
* Thuộc tính android:adjustViewBounds nếu nhận giá trị true thì ImageView tự động co biên vừa với ảnh.

**g) Webview**

****

*Hình 17. WebView*

**WebView** để hiện thị HTML trên ứng dụng, từ một đường dẫn có thể được tải và hiển thị bởi WebView.

Một số lý do sử dụng WebView:

* Khi màn hình thiết kế quá phức tạp, cần nhiều ViewGroup như FrameLayout,RelativeLayout,LinearLayout,...
* Khi cần tận dụng những dữ liệu có sẵn từ 1 trang web nào đó(VD web bán hàng,..).
* Để người dùng có thể mở đường dẫn ngay trong app mà không cần mở trình duyệt.
* Có 2 cách sử dụng WebView (cả 2 cách đều sử dụng method loadUrl(Uri uri)):
* Nạp web từ 1 file HTML vào WebView.
* Nạp web từ một đường dẫn vào WebView.

**4.3. Tìm hiểu về DataView trong Android**

**4.3.1 ListView**

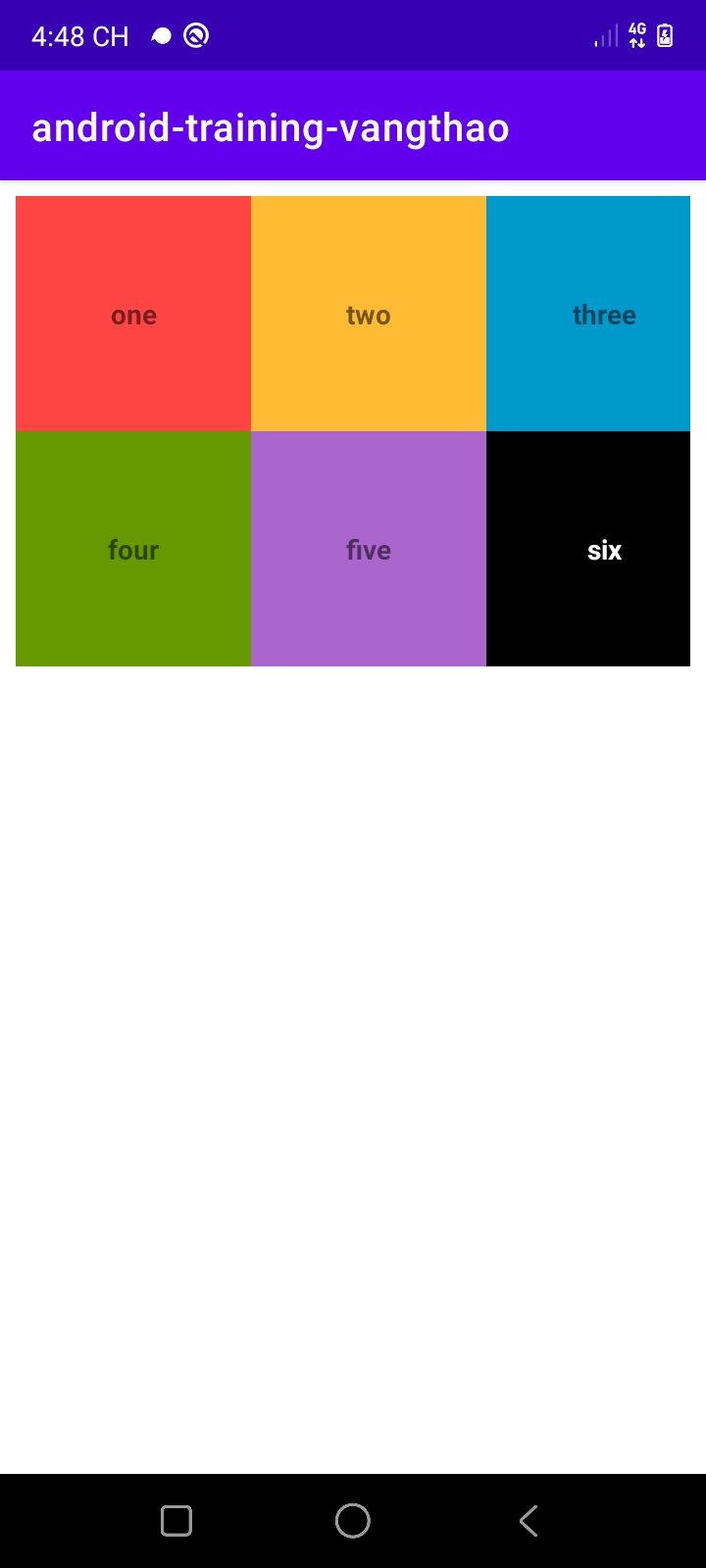


*Hình 18. Ví dụ về ListView*

**ListView** là phần tử View được dùng để hiển thị dữ liệu là một danh sách (list) từ một nguồn cấp gọi là Adapter.

* Các bước tạo và sử dụng ListView:
  + **Bước 1**: Khai báo ListView trong Layout.
  + **Bước 2**: Gán cho ListView một Adapter là nguồn cấp dữ liệu cho nó(listView.setAdapter(myAdapter)).
* Mỗi khi dữ liệu Adapter thay đổi ta cần gọi adapter.notifyDataSetChanged() để thông báo cho ListView biết mà cập nhật dữ liệu.
* Chúng ta có thể bắt sự kiện khi click vào mỗi item trong ListView bằng method listView.setOnItemClickListener(OnItemClickListener).
* Một vài phương thức(thuộc tính) hay dùng của ListView: android:divider để thiết lập Drawble hoặc Color cho đường kẻ giữa các list item.

**4.3.2 GridView**

****

*Hình 19. GridView*

**GridView** hiển thị các item trong mảng lưới hai chiều. Các item này không cần định nghĩa trước, chúng được tự động chèn vào bởi ListAdapter.

* Một số thuộc tính đặc biệt:
* android:verticalSpacing định nghĩa khoảng cách mặc định theo chiều dọc giữa các hàng(px,dp,sp,in hoặc mm).
* android:horizontalSpacing định nghĩa khoảng cách mặc định theo chiều ngang giữa các cột(px,dp,sp,in hoặc mm).
* android:numColumns xác định có bao nhiêu cột để hiển thị.

**4.3.3 ViewPager**

**ViewPager** là một trình quản lý Layout cho phép người dùng vuốt tay qua trái hoặc qua phải để chuyển sang nội dung của trang khác một cách đồng thời.

* ViewPager hỗ trợ từ API 13 trở lên.
* ViewPager không phải là một View chuẩn của Android, mà là một thành phần nằm trong gói android.support:design:xxx

**4.3.4 RecyclerView**

**RecyclerView** là một ViewGroup được dùng để chuẩn bị và hiện thị các View tương tự nhau.

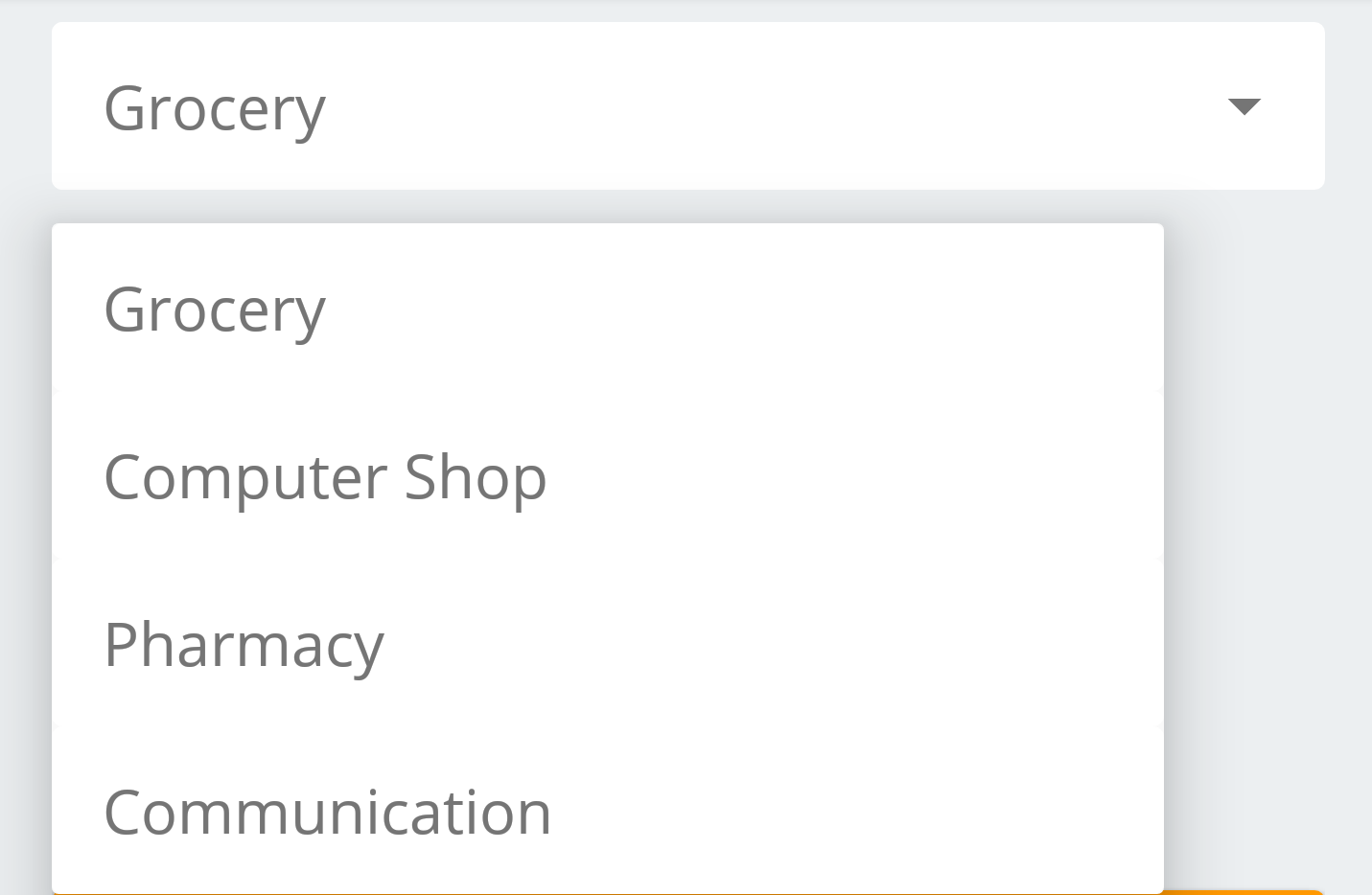
* RecyclerView được cho là sự kế thừa ListView và GridView.
* RecyclerView được giới thiệu trong gói suport-v7.
* Một số thành phần chính của RecyclerView:
* RecyclerView.Adapter : đây là nơi xử lý dữ liệu và gán cho View.
* Layoutmanager: xác định ra vị trí của các item trong RecyclerView.
* ItemAnimator: Tạo hiệu ứng cho các hành động thêm , sửa, xóa các item.
* Các bước để sử dụng RecyclerView:
* Thêm RecyclerView support library vào gradle build file.
* Định nghĩa ra model class để sử dụng data source.
* Thêm RecyclerView vào trong activity mà bạn muốn hiển thị.
* Tạo một tệp XML để xác định một item được biểu diễn như nào.
* Tạo ra RecyclerView.Adapter và ViewHolder để gán dữ liệu cho các item.
* Kết nối adapter với data source để đưa vào RecyclerView.

**4.3.5 AdapterView**

**AdapterView** là một đối tượng View.

* Nội dung của AdapterView luôn được xác định bởi một đối tượng khác, một Adapter.
* Adapter đóng vai trò như một liên kết giữa một tập hợp giữ liệu và một Adapter View.
* AdapterView hiện thị các bộ dữ liệu lớn rất hiệu quả.

**4.3.6 Spinner**

****

*Hình 20. Ví dụ Spinner*

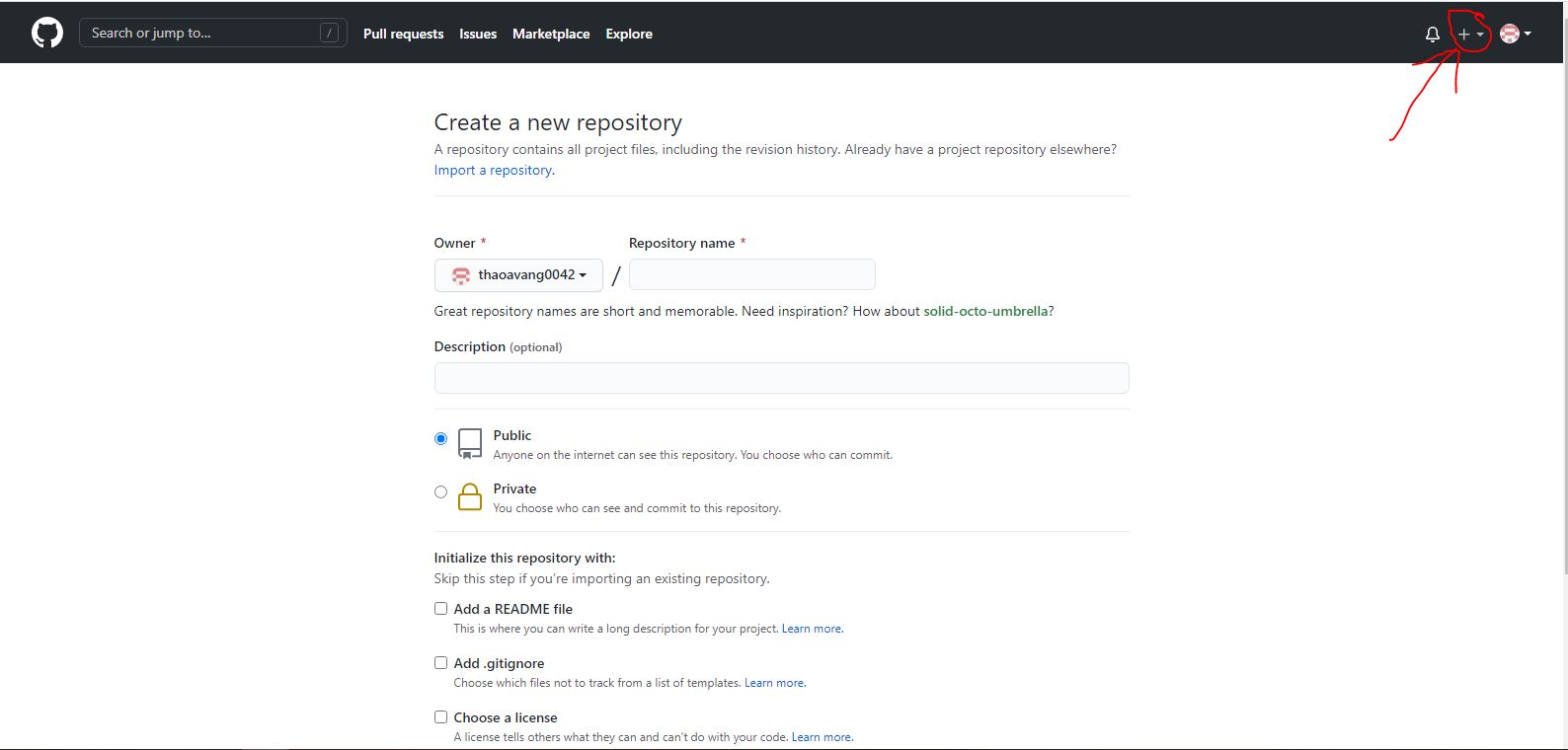
**Spinner** tương tự như ComboBox trong C# hay JComboBox trong Java.

* Spinner là một danh sách xổ xuống cho người dùng lựa chọn hay thực hiện 1 tác vụ nào đó.
* Cách đổ dữ liệu lên Spinner tương tự như ListView.
* Điểm khác biệt giữa Spinner và ListView là: Trong ArrayAdapter của Spinner chúng ta phải gọi setDropDownViewResource.

### 5. Training về git, github

**5.1 Tạo mới một repository**

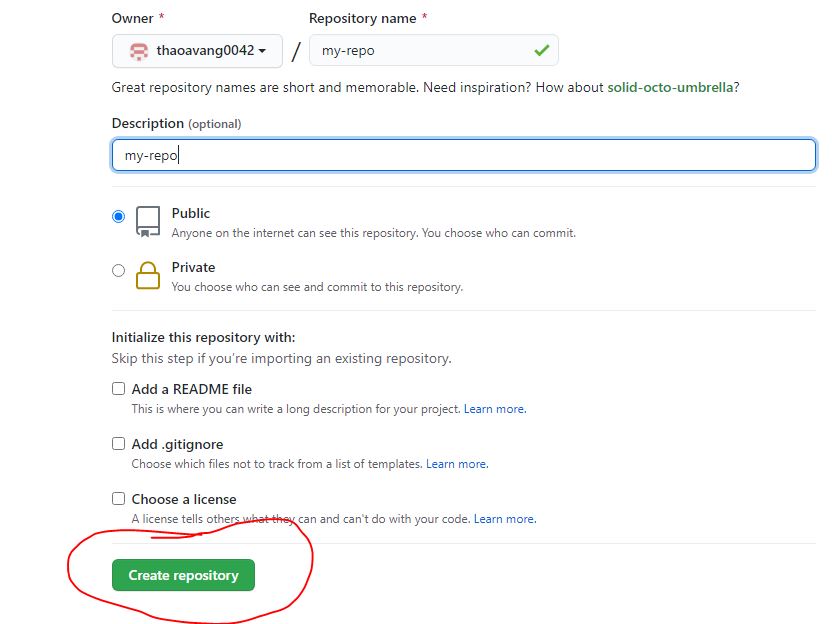
**Bước 1:** Nhấn vào icon dấu cộng bên phải tại trang github trên web.

****

*Hình 21. Tạo một repository mới bước 1*

**Bước 2:** Điền thông tin cho repository của bạn.

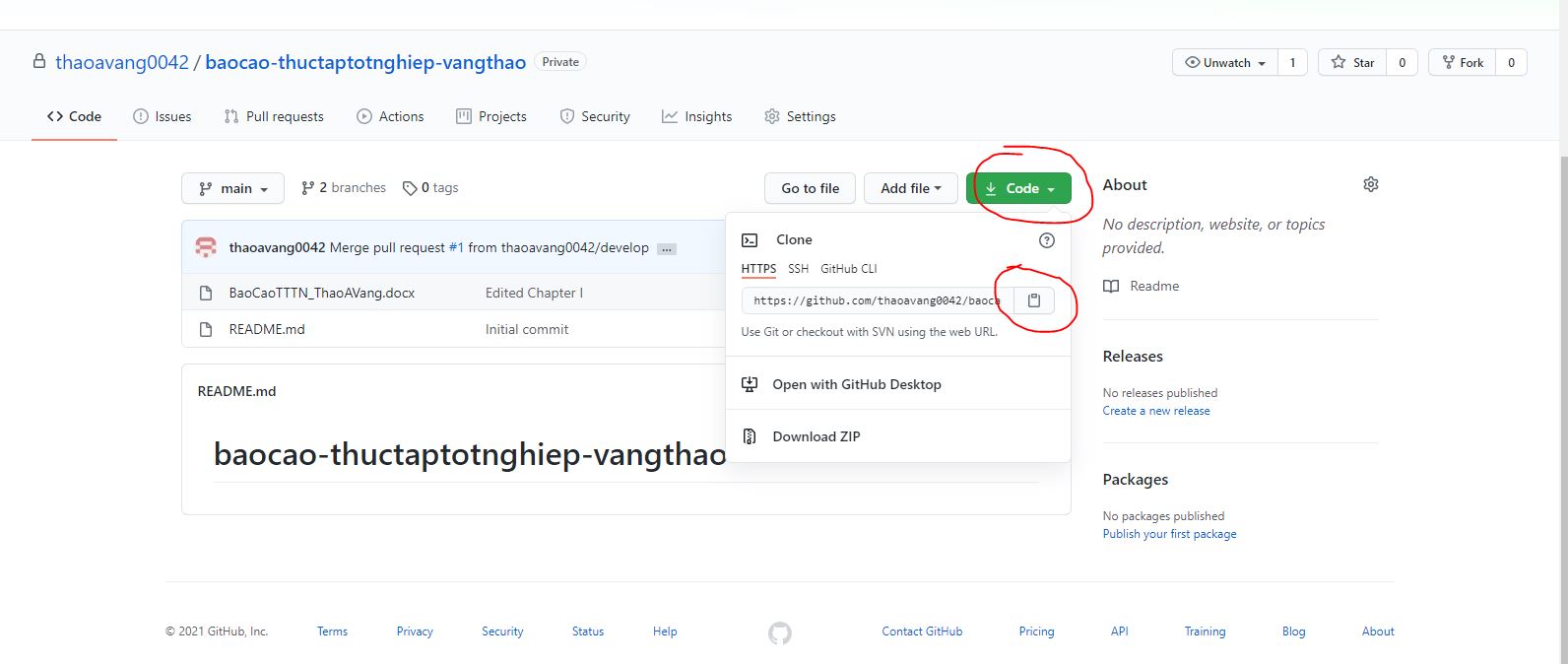
**Bước 3:** Nhấn vào nút **Create Repository**



*Hình 22. Tạo một repository mới bước 3*

**5.2 Clone một repository**

**Bước 1:** Truy cập vào repo đã tạo, chọn **Code** và copy đường link đến reop đó.



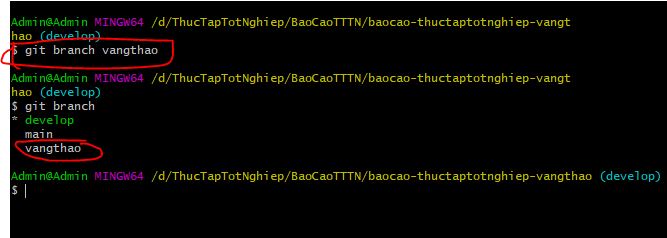
*Hình 23. Clone một repository*

**Bước 2:** Chúng ta gõ lệnh “*git clone <đường link vừa copy*>” bằng git bash. Hoặc chúng ta có thể clone bằng một IDE nào đó có hỗ trợ clone repo. VD như Android Studio IDE

**5.3 Làm việc với nhánh trong git**

**Tạo nhánh mới:**

- git branch <tên nhánh>. VD: git branch vangthao

****

*Hình 24. Tạo nhánh mới trong git*

**Checkout một nhánh:**

- git checkout <tên nhánh>

**5.4 Một số lệnh hay dùng để làm việc với git**

**Thêm các file được thay đổi:**

* + git add .

**Commit code:**

* git commit –m “Messge of commit”

**Push code:**

* + git push origin <branch name>: để push code lên 1 nhánh
  + git push: để push code lên nhánh đang được checkout.

**Pull code:**

* + git pull : sẽ pull (lấy về) mã nguồn mới nhất từ nhánh chính.
  + git pull <branch name> : sẽ pull về từ nhánh bất kỳ với tên nhánh đó.

# CHƯƠNG III KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## **I. Kết luận**

### 1. Kỹ năng mềm

Sau khoảng thời gian tham gia thực tập tại Công ty Cổ Phần Quốc Tế TDA Việt Nam giúp em học hỏi được rất nhiều điều cũng như rèn luyện thêm những kỹ năng mềm trong công việc

* Kỷ luật và chấp hành đúng giờ làm việc

Trong quá trình thực tập ở công ty, kỷ luật là một điều rất quan trọng. Nếu bản thân không thực hiện tốt được những kỷ luật đó, công việc dù giỏi đến đâu cũng không thể hoàn thành và đánh giá cao.

Ở TDA Việt Nam, mỗi ngày làm việc 8 tiếng, bắt đầu từ 8h30-18h00 các ngày từ thứ 2 đến thứ 6. Thời gian chính là yếu tố cần thiết nhất cho công việc. Vì thế, việc sắp xếp thời gian hợp lý là một điều hết sức cần thiết. Và TDA Việt Nam đã giúp em trưởng thành hơn trong vấn đề giờ giấc.

* Kỹ năng giao tiếp, tác phong ứng xử chuyên nghiệp hơn

Kỹ năng giao tiếp luôn được xem là yếu tố then chốt đối với sự phát triển toàn diện của một người. Nhất là làm việc trong môi trường công nghệ thông tin, cần phải biết trình bày và giải thích rõ ràng, cùng những người khác tìm ra và thực hiện giải pháp, giao nhiệm vụ cho cả nhóm một cách hiệu quả.

TDA Việt Nam cũng giúp thay đổi những tác phong ứng xử trong môi trường công sở:

- Tạo được mối quan hệ thân thiết với những anh chị đồng nghiệp, được anh chị nhiệt tình chỉ bảo.

- Hòa nhã với các nhân viên nơi thực tập, tham gia sôi nổi các hoạt động ngoài giờ làm việc cùng mọi người trong dự án.

- Chủ động tiếp cận công việc và sẵn sàng hỗ trợ đồng nghiệp để có thể hoàn thành công việc chung, tự khẳng định năng lực bản thân.

- Phong cách, trang phục luôn chỉnh tề, phù hợp, lịch sự

* Khả năng làm việc nhóm

Làm việc theo nhóm giúp cho bản thân em rất nhiều. Giúp em có thể lắng nghe ý kiến từ nhiều phía khác nhau, trợ giúp và tôn trọng ý kiến của mọi người. Bên cạnh đó, làm việc nhóm cũng giúp em rèn luyện khả năng Thuyết phục- thuyết phục các thành viện phải trao đổi, suy xét những ý tưởng đã đưa ra. Đồng thời em cũng nâng cao khả năng tự bảo vệ và thuyết phục người khác đồng tình với ý kiến của mình.

* Lên kế hoạch công việc và quản lý thời gian

Lên kế hoạch cho công việc mình làm là một trong những kỹ năng em học được ở công ty. Để làm tốt công việc được giao, bản thân em đã biết lập kế hoạch cho các vấn đề, những phần quan trọng cần giải quyết được sắp xếp lên đầu cứ như thế theo thứ tự giảm dần

Việc này không chỉ giúp bản thân em điều chỉnh thời gian cho phù hợp mà còn giúp tiến độ công việc được nâng cao.

### 2. Áp dụng thực tiễn

Bài báo cáo đã trình bày được lý thuyết , demo về các thành phần cơ bản trong Android, các công cụ hỗ trợ làm việc theo team.

Tìm hiểu và áp dụng thực tế công cụ Android Studio. Có thể nói, Android Studio là công cụ để lập trình Android mạnh mẽ, hỗ trợ nhiều cho các lập trình viên để họ xây dựng các ứng dụng tiện lợi, nhanh chóng hơn.

### 3. Hạn chế

Do thời gian còn hạn chế và kinh nghiệm thực tế chưa có nhiều nên không thể tránh khỏi những thiếu sót về kiến thức, kỹ năng cũng như kết quả đạt được. Em sẽ cố gắng khắc phục những hạn chế đó để có thể hoàn thành tốt bài báo cáo trong đợt Bảo vệ đồ án sắp tới.

## **II. Hướng phát triển**

**-** Tích cực phát huy kỹ năng mềm đã tích lũy được trong môi trường làm việc thực tế

**-** Áp dụng những kiến thức đã học được để tiếp tục học hỏi, làm các dự án phần mềm thực tế

# TÀI LIỆU THAM KHẢO

[1] Tài liệu phát triển Android chính thức của Google https://developer.android.com/docs

[2] Trang codepath: https://guides.codepath.com/android

[3] https://mindx.edu.vn/blog/post/lap-trinh-android

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………