BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH

KHOA CÔNG NGHỆ THÔNG TIN

Môn Lập Trình Di Động Nâng Cao

Ứng Dụng Đọc Tin Tức Online

Giảng viên hướng dẫn: Giang Hào Côn

Sinh viên thực hiện: Nguyễn Đức Nhật

MSSV: 1611536220

Chuyên ngành: Kỹ thuật phần mềm

Môn học: Di động nâng cao

Khóa: 2016

Tp.HCM, tháng 09 năm 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH

KHOA CÔNG NGHỆ THÔNG TIN

Môn Lập Trình Di Động Nâng Cao

Ứng Dụng Đọc Tin Tức Online

Giảng viên hướng dẫn: Giang Hào Côn

Sinh viên thực hiện: Nguyễn Đức Nhật

MSSV: 1611536220

Chuyên ngành: Kỹ thuật phần mềm

Môn học: Di động nâng cao

Khóa: 2016

Tp.HCM, tháng 09 năm 2019

LỜI NÓI ĐẦU

Mạng điện thoại di động xuất hiện tại Việt Nam từ đầu những năm 1990 và theo thời gian số lượng các thuê bao cũng như các nhà cung cấp dịch vụ đi động tại Việt Nam ngày càng tăng. Do nhu cầu trao đổi thông tin ngày càng tăng và nhu cầu sử dụng sản phẩm công nghệ cao nhiều tính năng, cấu hình cao, chất lượng tốt, kiểu dáng mẫu mà đẹp, phong phú nên nhà cung cấp phải luôn luôn cải thiện, nâng cao những sản phẩm của mình. Do đó việc xây dựng các ứng dụng cho điện thoại di động đang là một ngành công nghiệp mới đầy tiềm năng và hứa hẹn nhiều sự phát triển vượt bậc của ngành khoa học kĩ thuật.

Cùng với sự phát triển của thị trường điện thoại di động là sự phát triển mạnh mẽ của xu hướng lập trình phần mềm ứng dụng cho các thiết bị di động. Phần mềm, ứng dụng cho điện thoại di động hiện nay rất đa dạng và phong phú trên các hệ điều hành di động cũng phát triển mạnh mẽ và đang thay đổi từ ngày. Các hệ điều hành J2ME, Android, IOS, Hybrid, Web based Mobile Application đã có rất phát triển trên thị trường truyền thông di động.

Ngày nay, với sự phát triển nhanh chóng của xã hội, nhu cầu cập nhật tin tức xã hội mọi lúc mọi nơi là rất cần thiết, vì vậy em đã chọn đề tài "Xây dựng ứng dụng đọc tin tức online" với mục đích nghiên cứu, tìm hiểu về hệ điều hành Android và xây dựng ứng dụng đọc tin tức online để có thể đáp ứng được nhu cầu cập nhật tin tức xã hội hiện nay một cách nhanh chóng và tiện lợi.

LỜI CẢM ƠN

Trong học ký vừa qua, dưới sự dạy dỗ, giúp đỡ của các thầy cô trường Đại học Nguyễn Tất Thành nói chung, các thầy cô trong khoa Công Nghệ Thông Tin nói riêng, đã truyền đạt cho em nhiều kiến thức và kinh nghiệm quý báu góp phần không nhỏ vào quá trình học tập và thực hiện đồ án của em.

Với sự giúp đỡ của thầy cô trong khoa Công Nghệ Thông Tin, đặc biệt là sự hướng dẫn tận tình của thầy Giang Hào Côn – GV Trường Đại Học Nguyễn Tất Thành, em đã hoàn thành được đồ án môn di động nâng cao của mình. Và em cũng xin biết ơn bố mẹ và bạn bè đã ủng hộ, giúp đỡ và động viên em trong những lúc khó khăn. Tuy đồ án đã hoàn thành, song không tránh khỏi thiếu sót, em rất mong được sự đóng góp, giúp đỡ của các thầy cô và các bạn.

Cuối cùng, em xin gửi lời cảm ơn đến các thầy cô trong khoa Công nghệ thông tin, em xin được bày tỏ lòng cảm ơn sâu sắc đến thầy Giang Hào Côn đã tận tình giúp đỡ để em có thể hoàn thành được đồ án này.

Em xin chân thành cảm ơn.

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

Điểm đồ án: ……

……

……

……

TPHCM, Ngày …… tháng …… năm

Giáo viên hướng dẫn

MỤC LỤC

DANH MỤC CÁC BẢNG BIỂU

DANH MỤC CÁC BẢNG HÌNH

DANH MỤC CÁC TỪ VIẾT TẮT

Từ viết tắt	**Ý nghĩa**
XML	Extensible Markup Language
JSON	JavaScript Object Notation
CSDL	Cơ sở dữ liệu
SQL	Structured Query Language
UI	User Interface
URL	Uniform Resource Locato
GCM	Google Cloud Message
HTTP	HyperText Transfer Protocol
SSL	Secure Sokets Layer
STT	Số thứ tự

TỔNG QUAN VỀ ĐỀ TÀI

I. Cơ sở khoa học và tính thực tiễn của đề tài

1. Cơ sở khoa học

Trong vài năm trở lại đây, hệ điều hành Android ra đời với sự kế thừa những ưu việt của các hệ điều hành ra đời trước và sự kết hợp của nhiều công nghệ tiên tiến nhất hiện nay, đã được nhà phát triển công nghệ rất nổi tiếng hiện nay là Google. Android đã nhanh chóng là đối thủ cạnh tranh mạnh mẽ với các hệ điều hành trước đó và đang là hệ điều hành di động của tương lai và được nhiều người ưa chuộng nhất.

Cùng với các tin tức và bài báo được cập nhật liên tục hàng ngày, cho nên việc xem trên các trang báo giấy truyền thống khiến chúng ta mất nhiều thời gian và công sức, do đó nhu cầu cập nhật tin tức một cách nhanh nhất được xem là giải pháp ưu tiên nhất.

2. Tính thực tiễn

Việc xem tin tức với nhiều người qua các trang báo giấy hoặc trên các phương tiện truyền thông vẫn chưa đáp ứng được nhu cầu và thông tin người đọc, do các tin tức được cập nhật liên tục hàng giờ nên sẽ rất khó để các trang báo giấy truyền thống có thể đáp ứng kịp.

Vì vậy, việc triển khai một ứng dụng đọc tin tức trên di động là hết sức cần thiết để đáp ứng nhu cầu cập nhật tin tức nhanh nhất và thuận lợi cho mọi người.

II. Mục tiêu đề tài

1. Lý thuyết

- Nghiên cứu ngôn ngữ java.
- Nghiên cứu xây dựng giao diện xml trong Android
- Nghiên cứu lập trình trên hệ điều hành Android

2. Thực tiễn

- Vận dụng thành thạo, nắm bắt được ưu và nhược diểm của ngôn ngữ java.
- Ứng dụng đọc tin tức online chạy trên hệ điều hành Android gồm có 2 mục tiêu:
- Xây dựng được ứng dụng đọc tin tức online trên Android.
- Đáp ứng nhu cầu của người dùng:
    - Xem tin tức được trên smart phone
    - Cập nhật tin tức nhanh nhất
    - Đăng nhập, đăng ký tài khoản qua Firebase

III. Phương pháp nghiên cứu

1. Về lý thuyết

- Tìm hiểu cách thức hoạt động và cấu trúc của một trang báo.
- Dựa trên sự tìm hiểu về lập trình Android trên thiết bị di động.
- Tìm hiểu cách thức xây dựng ứng dụng trên Android.

2. Triển khai

- Xây dựng ứng dụng chạy trên Android theo các bước:
    - Khảo sát các website tin tức.
    - Khảo sát nhu cầu người dùng điện thoại về các yêu cầu của trang báo.
    - Nghiên cứu các công nghệ phát triển trên nền tảng Android
- Sau đây là các công nghệ áp dụng trong đồ án này:
    - Android SDK
    - Java

IV. Nội dung nghiên cứu và bố cục đồ án

1. Nội dung nghiên cứu

- ✓ Phân tích yêu cầu.
- ✓ Lựa chọn công nghệ.
- ✓ Nghiên cứu cơ sở lý thuyết công nghệ đã chọn.
- ✓ Áp dụng các cơ sở lý thuyết vào ứng dụng thực tiễn.
- ✓ Kiểm tra, tham khảo và tối ưu ứng dụng.

2. Bố cục đồ án

1. Cơ sở lý thuyết
2. Phân tích và thiết kế
3. Triển khai ứng dụng
4. Hạn chế và hướng phát triển

Chương 1. Cơ Sở Lý Thuyết

1.1. SQLITE

1.1.1. Giới thiệu

SQLITE là một loại CSDL quan hệ, nhỏ gọn, nó là hệ CSDL được sử dụng rất rộng rãi trên nhiều nền tảng (Mobile, Destop, Webserver ...), nó có các ưu điểm có thể kế ra như: Là hệ CSDL SQL nhúng, sử dụng ngay mà không cần cấu hình, không cần có một Server SQL riêng (khác với MySQL, MS SQL Server ...) ...

Mặc định mỗi ứng dụng sẽ được cấp phát một thư mục cho việc lưu trữ cơ sở dữ liệu và nó chỉ có thể được dùng bởi ứng dụng đó. Nếu muốn chia sẻ dữ liệu dùng chung giữa các ứng dụng ta có thể sử dụng Content Provider.

1.1.2. XÂY DỰNG CƠ SỞ DỮ LIỆU VỚI SQLITE

- Tạo cơ sở dữ liệu thông qua lớp SQLiteOpenHelper.
    - SQLiteOpenHelper là một lớp ảo, SQLiteOpenHelper giúp tạo các cơ sở dữ liệu dùng SQLite (vì SQLite không hỗ trợ các phương thức khởi tạo cơ sở dữ liệu). Vậy làm sao để sử dụng được SQLiteOpenHelper? Vì SQLiteOpenHelper là một lớp ảo nên ta cần khai báo một lớp khác kế thừa lớp này.
    - SQLiteOpenHelper thực hiện các phương thức cần thiết cho phép khởi tạo, nâng cấp cơ sở dữ liệu. Tạo đối tượng để truy cấp cơ sở dữ liệu (Read và Write).
    - SQLiteDatabase cung cấp phương thức insert(), update(), delete(), hoặc execSQL() cho phép thực hiện truy xuất dữ liệu.
        - Override phương thức onCreate() để tạo cơ sở dữ liệu.
        - Override phương thức onUpgrade() để nâng cấp cơ sở dữ liệu.
- Lớp SQLiteOpenHelper cung cấp 2 phương thức getReadableDatabase() và getWriteableDatabase() để trả về đối tượng SQLiteDatabase.

1.2. Thread handler_Asynctask

1.2.1. Thread

1.2.1.1. Khái niệm

Thread là một tiến trình đơn vị xử lý của máy tính có thể thực hiện một công việc riêng biệt. Multi-thread là khái niệm cho nhiều tiến trình chạy đồng thời. Một ứng dụng Java ngoài luồng chính có thể có các luồng khác thực thi đồng thời làm ứng dụng chạy nhanh và hiệu quả hơn.

1.2.1.2. Ưu điểm của đa luồng

Mỗi luồng có thể dùng chung và chia sẻ nguồn tài nguyên trong quá trình chạy, nhưng có thể thực hiện một cách độc lập. Ứng dụng trách nhiệm có thể được tách + Luồng chính chạy giao diện người dung + Các luồng phụ nhiệm gửi đến luồng chính.

1.2.1.3. Nhược điểm của đa luồng:

Càng nhiều luồng thì xử lý càng phức tạp Cần phát hiện tránh các luồng chết, luồng chạy mà không làm gì trong ứng dụng cả

- Ví dụ

```
    Thread(new Runnable() {
        new public void run() {
        final Bitmap bitmap =
        loadImageFromNetwork("http://example.com/image.png");
    mImageView.post(new Runnable() {
        public void run() {
        mImageView.setImageBitmap(bitmap);
        }
});
}
}).start();
```

1.2.2. Handler

- Handler là đối tượng cho phép gửi, xử lí các thông điệp và các đối tương Runable có liên quan đến hàng đợi thông điệp. Mỗi Handler có thể liên kết với một thread và hàng đợi thông điệp của thread đó.
- Bạn có thể tạo ra một thread của riêng bạn và giao tiếp ngược với main thread của ứng dụng thông qua một Handler. Điều này được thực hiện bằng cách gọi sendMessage nhưng từ thread mới của bạn.

▪ **Khởi tạo**

- Tạo một đối tượng Handler và ghi đè lại phương thức handleMessage để bắt và xử lý các thông điệp liên lạc.

```
        Handler handler = new Handler(new Handler.Callback() {
        @Override
            public boolean handleMessage(Message msg) {
```

```
        //Kiểm tra message
        if(msg != null){
          //Xử lý message nhận được từ Thread
      }
          return true;
        }
      });
```

- Thread sẽ gửi tin nhắn lên Handler và Handler đóng vai trò kiểm tra tin nhắn là gì và làm những việc tương ứng.
- Ta chú ý các phương thức sendMessage sau:
    - sendMessage(): gửi tin nhắn lên Handler ngay lập tức.
    - sendMessageAtFrontOfQueue(): gửi tin nhắn lên Handler và tin nhắn đó sẽ ưu tiên giải quyết trước.
    - sendMessageAtTime(): gửi tin nhắn vào thời điểm chỉ định trước.
    - sendMessageDelayed(): gửi tin nhắn vào Handler sau một khoảng thời gian nhất
- định.

```
Thread t1 = new Thread(new Runnable() {
  @Override
    public void run() {
      Message mss = new Message();
      mss.arg1 = 1;
      handler.sendMessage(mss);
        }
        });
    t1.start();
```

1.2.3. AsynTask

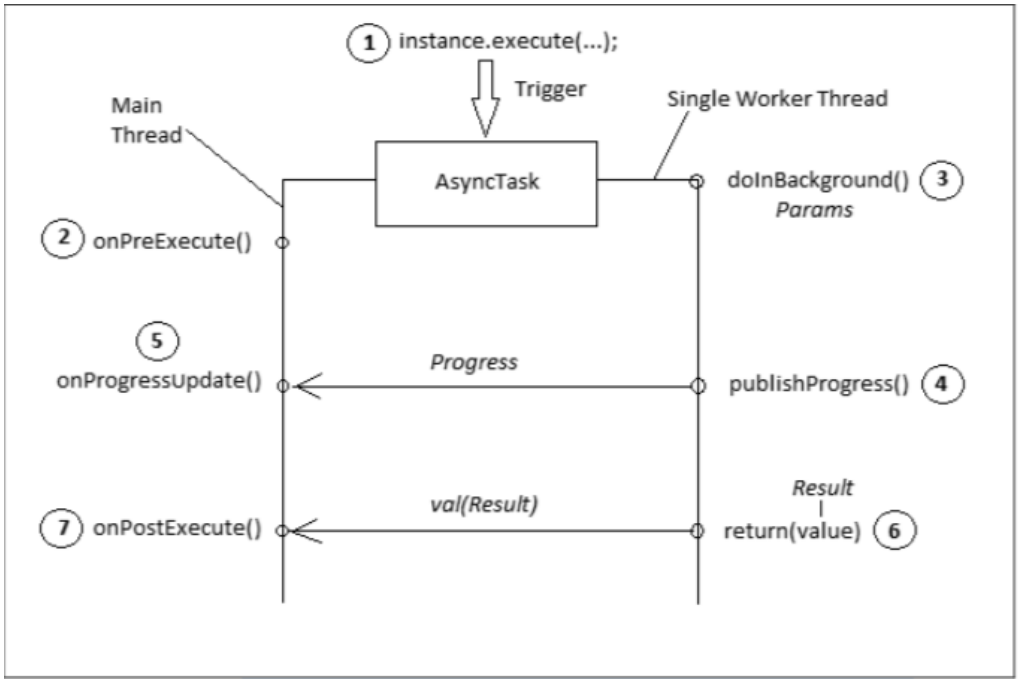
1.2.3.1. Khái niệm

- Lớp trừu tượng cho phép thực hiện khai báo các tiến trình xử lý ngầm và cập nhật giao diện một cách tự động.
- Bao gồm cơ chế tự động tính toán một cách bất đồng bộ để đưa ra thời gian thực thi cho mỗi tiến trình xử lý.

- Thời gian xử lý được tính toán thông qua ba tham số:
    - Params: các thông số đầu vào.
    - Progress: dữ liệu trong quá trình xử lý.
    - Result: kết quả cuối cùng của tiến trình xử lý.

1.2.3.2. Các phương thức quan trọng

- Các phương thức cho phép nắm giữ dữ liệu ở nhiều giai đoạn khác nhau của tiến trình xử lý bao gồm 4 bước:
    - onPreExcute
        - Tiến trình tiền xử lý.
    - doInBackground
        - Tiến trình đang ở chế độ ngầm.
    - onProgessUpdate
        - Tiến trình cập nhật dữ liệu.
    - onPostExcute
        - Tiến trình cập nhật dữ liệu trên UI.
- Các qui tắc quan trọng khi xử lý trên AsyncTask:
- AsyncTask phải được tạo, gọi và thực thi trên tiến trình UI
- Tự động hoá từ phiên bản JellyBean
- Không được gọi tường minh các phương thức xử lý onPostExcute, doInBackground…
- Mỗi AsyncTask chỉ được gọi một lần duy nhất.
- Có thể thực hiện huỷ AsyncTask thông qua phương thức onCancel(boolean)
    - Khi đó phương thức onCancel(Object) sẽ được gọi sau phương thức doInBackground thay vì onPostExcute.



Hình 1.1 Thứ tự hoạt động của các hàm trong AsyncTask

1.3. Khai thác tài nguyên internet

1.3.1. Tổng quan tài nguyên internet

1.3.1.1. Tài nguyên internet trên thiết bị di động

- Ứng dụng trực tuyến trên thiết bị di động được xây dựng nhằm mục đích hiển thị dữ liệu đầu cuối được xử lý từ các máy chủ cũng như gửi các yêu cầu về truy xuất dữ liệu.
- Khi xây dựng các ứng dụng trực tuyến cần xác định rõ nguồn tài nguyên được cung cấp:
    - Định dạng tài nguyên (văn bản, hình ảnh…)
    - Giao thức kết nối đến máy chủ (http, https, tcp, rtsp…)
    - Sử dụng công nghệ kết nối không dây (Wifi, Internet Mobile…)

1.3.1.2. Các vấn đề kết nối

- Xây dựng ứng dụng thực hiện kết nối Internet trên thiết bị di động cần đảm bảo các vấn đề sau:
    - **Vấn đề về băng thông**: khi sử dụng trình duyệt để duyệt đến web ứng dụng, tất cả các tài nguyên hầu như bắt buộc phải tải lại từ đầu, ví dụ: hình ảnh, giao diện, âm thanh... điều này tạo ra một lượng băng thông lớn để truy xuất các dữ liệu tĩnh, thay vào đó xây dựng ứng dụng có thể giúp chúng ta lưu trữ những dữ liệu này một cách dễ dàng.
    - **Lưu Cache**: các trình duyệt chỉ có thể lưu giữ một lượng thông tin nhỏ về trang web mà chúng làm việc, trong khi đó ứng dụng hoàn toàn có thể lưu trữ bất kỳ

dữ liệu nào mà lập trình viên mong muốn từ dữ liệu truy xuất đến thao tác người dùng. Khi có sự cố về kết nối, ứng dụng sẽ lưu trữ lại thông tin tại thời điểm mất kết nối và tự động xử lý khi thiết bị có lại kết nối.

- **Năng lượng**: các kết nối Internet khi được thực hiện sử dụng một lượng lớn năng lượng cho mỗi lần kết nối, tuy nhiên việc xây dựng ứng dụng sẽ cho phép chúng ta tổ chức quản lý kết nối cũng như duy trì các kết nối khi cần thiết.
- **Tính năng thiết bị**: các ứng dụng nền web hầu như không thể nào truy xuất đến phần cứng cũng như các đặc tính quan trọng trên một thiết bị di động. Ví dụ: định vị, máy ảnh, cảm biến... đương nhiên, khi xây dựng ứng dụng chúng ta hoàn toàn có thể dễ dàng truy xuất các đặc tính thiết bị thông qua các API mà bộ SDK cung cấp.

1.3.1.3. Các hình thức kết nối

- Các thiết bị di động ở thời điểm hiện tại hỗ trợ khá nhiều hình thức kết nối đến Internet, tuy nhiên ta có thể quy chung về dạng sau:
  - Mobile Internet: một dạng kết nối thông qua hình thức cung cấp dịch vụ băng thông của nhà mạng, bao gồm một số chuẩn phổ biến sau: GPRS, EDGE, 3G, 4G, LTE...
  - Wifi: kết nối không dây và truyền dữ liệu ra Internet thông các thiết bị mạng (router, switch...), ngoài ra thiết bị di động Android có khả năng phát tín hiệu mạng cho các thiết bị khác.
- Việc xây dựng ứng dụng thực hiện kết nối đến Internet phụ thuộc vào khá nhiều hình thức kết nối mà thiết bị người dùng đang sử dụng. Do đó, để tối ưu hóa kết nối và tăng trải nghiệm người dùng cần thực hiện sử dụng hình thức kết nối một cách hợp lý. Ta có thể thấy các mạng di động thường có băng thông thấp và chi phí cao để sử dụng, trong khi đó Wifi lại tùy thuộc và băng thông của nguồn kết nối.

1.3.1.4. Lớp khai báo kết nối

- Việc khai báo kết nối tùy thuộc vào chuẩn giao thức kết nối và địa chỉ URL của máy chủ cần kết nối:
  - URL: lớp giải quyết phân giải tên miền thành địa chỉ IP.
  - Định dạng: http://username:password@host:8080/directory/file?query#ref:
  - Phương thức kết nối
    - openConnection

- URLConnection: lớp thực hiện kết nối đến URL được chỉ định cho việc đọc hoặc ghi dữ liệu, hỗ trợ các giao thức:
  - File: URIs
- FTP
- HTTP & HTTPS
- Jar
- URLConnection: Các phương thức quan trọng:
- connect()
- getContent()
- getContentType()
- getInputStream()
- getOutputStream()
- setDoInput-getDoInput
- setDoOutput-getDoOutput

- Các lớp giao thức HTTP & HTTPS:
  - HttpURLConnection
  - HttpsURLConnection
- Các phương thức quan trọng:
  - disconnect()
  - getContentEncoding()
  - getResponseCode()
  - getResponseMessage()

1.3.1.5. Thực hiện kết nối Internet (HTTP)

- Bao gồm các bước sau:
  - Thực hiện mở kết nối đến địa chỉ URL được chỉ định → HttpURLConnection.
  - Thực hiện khai báo các header, content-type, cookies, ... o Gọi phương thức setDoOutput(true) thực hiện xây dựng phần dữ liệu cần gửi lên máy chủ, dữ liệu được thiết lập được trả thông qua phương thức getOutputStream(). (Optional).
  - Thực hiện truy xuất dữ liệu thông qua phương thức getInputStream(), để lấy các thông tin về nội dung được trả, độ dài, thời gian...
  - Gọi phương thức disconnect() để đóng các kết nối khi kết thúc phiên làm việc và giải phóng tài nguyên.
- Một số lưu ý khi thực hiện:
  - Cần thực hiện xin cấp quyền truy cập Internet cho ứng dụng trong tập tin

AndroidManifest.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

- Trong một vài trường hợp cần xin cấp quyền kiểm soát trạng thái Internet

```
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

- Không được thực hiện kết nối Internet trực tiếp trên tiến trình chính của ứng dụng. Sử dụng AsyncTask hoặc Thread để thay thế.

- Ví dụ:

```
URL url = new URL("http://t3h.vn");  URLConnection urlConnection =
url.openConnection(); HttpURLConnection httpURLConnection =

    (HttpURLConnection) urlConnection; int responseCode =
    httpURLConnection.getResponseCode();

    if (responseCode == HttpURLConnection.HTTP_OK) { InputStream
    inputStream =

    httpURLConnection.getInputStream();

    // Xử lý đọc dữ liệu từ InputStream
    }
```

1.3.1.6. Thực hiện kết nối Internet (HTTPS)

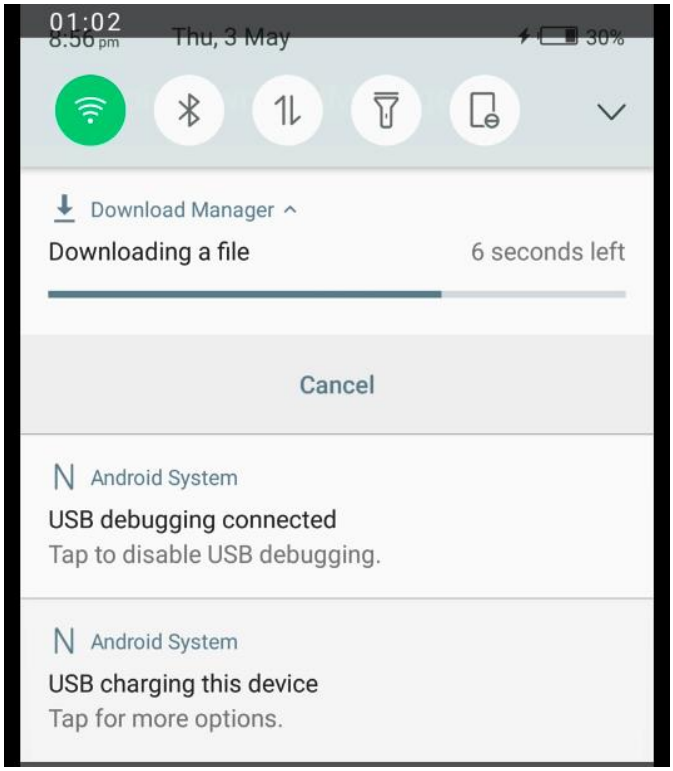
- Bao gồm các bước sau:
  - Khai báo KeyStore dùng để chứng thực.
  - Chứng thực KeyStore thông qua **X509TrustManager** hoặc **SSLSocketFactory**.
  - Thực hiện mở kết nối đến địa chỉ URL được chỉ định → **HttpsURLConnection**.
  - Thực hiện gắn SSLContext cho việc chứng thực.
  - Gọi phương thức **setDoOutput**(true) thực hiện xây dựng phần dữ liệu cần gửi lên máy chủ, dữ liệu được thiết lập được trả thông qua phương thức **getOutputStream**().
  - Thực hiện truy xuất dữ liệu thông qua phương thức **getInputStream**(), để lấy các thông tin về nội dung được trả, độ dài, thời gian...
  - Gọi phương thức disconnect() để đóng các kết nối khi kết thúc phiên làm việc và giải phóng tài nguyên.
- Ví dụ:

```
KeyStore keyStore = ...;  String algorithm =
TrustManagerFactory.getDefaultAlgorithm(); TrustManagerFactory tmf =
TrustManagerFactory.getInstance(algorithm); tmf.init(keyStore);

SSLContext context = SSLContext.getInstance("TLS"); context.init(null,
tmf.getTrustManagers(), null);

URL url = new URL("https://www.example.com/"); HttpsURLConnection
urlConnection = (HttpsURLConnection) url.openConnection();
urlConnection.setSSLSocketFactory(context.getSocketFactory()); InputStream in =
urlConnection.getInputStream();
```

1.3.2. Sử dụng dịch vụ Download Manager

1.3.2.1. Giới thiệu dịch vụ Download Manager

- Download Manager là trình quản lý các tác vụ tải trên thiết bị Android được Google tích hợp từ phiên bản Android 2.3 (API 9). Download Manager được thiết kế hoạt động như một dịch vụ chạy ngầm, cho phép quản lý và giám sát sự thay đổi của các kết nối HTTP cũng như các hoạt động khởi động lại của thiết bị để đảm bảo các nội dung được hoàn thành một cách triệt để.
- Download Manager được sử dụng trong các trường hợp cần thực hiện tải một nội dung trong một thời gian dài ở chế độ ngầm giữa các phiên tương tác của người dùng và việc hoàn thành nội dung tải được đặt lên hàng đầu.
- Tất cả các tập tin được tải thông qua Download Manager được quản lý bởi ứng dụng Download trên thiết bị.



Hình 1.2 Ví dụ minh họa Download Manager

1.3.2.2. Khai báo và sử dụng Download Manager

- Việc truy xuất Download Manager được thông qua phương thức getSystemService().

```
DownloadManager
manager=(DownloadManager)getSystemService(DOWNLOAD_SERVICE);
```

- Để khởi tạo yêu cầu tải một nội dung, tiến hành tạo đối tượng Request với tham số truyền vào là một Uri, một dạng địa chỉ kết nối đến máy chủ chứa nội dung cần tải. Đối tượng này sau đó sẽ được xếp vào hàng đợi để bắt đầu tải thông qua phương thức **enqueue**().

```
DownloadManager manager = (DownloadManager)
getSystemService(DOWNLOAD_SERVICE);

Uri uri = Uri.parse("http://developer.android.com/downloads/design/roboto-1.2.zip");
Request request = new Request(uri);

long id_reference = manager.enqueue(request);
```

- Khi phương thức enqueue() được gọi, việc tải nội dung sẽ được bắt đầu thực hiện khi kết nối mạng khả dụng và hàng đợi của Download Manager đang rỗng.

- Phương thức enqueue() cũng trả về một biến tham chiếu id_reference, được sử dụng để thực hiện các thao tác liên quan đến yêu cầu tải đang được thực hiện trong hàng đợi. Ví dụ, ta có thể thực hiện truy vấn xem trạng thái của yêu cầu tải đã hoàn thành chưa hoặc đơn giản có thể hủy yêu cầu tải dựa trên tham chiếu này.

- Khi thực hiện tải một nội dung, chúng ta có thể cân nhắc về dung lượng của nội dung và cho phép sử dụng các kết nối phù hợp nội dung đó thông qua phương thức setAllowedNetworkTypes(), lựa chọn tải bằng Wi-fi hay mạng di động.

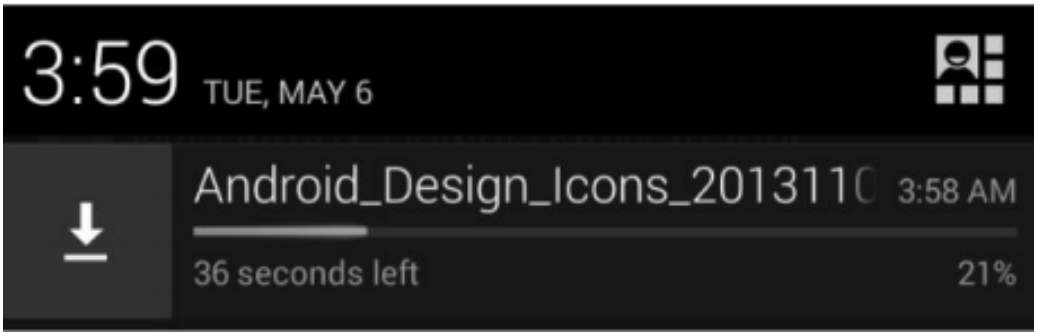
- Từ phiên bản Android 3.0, phương thức getRecommendedMaxBytesOverMobile() được bổ sung cho phép ứng dụng có thể nhận biết dung lượng lớn nhất có thể mà thiết bị có thể thực hiện tải thông qua kết nối của mạng di động.

- Để nhận biết việc hoàn thành tải nội dung, ta tiến hành xây dựng một BroadcastReceiver để bắt lại hành động ACTION_DOWNLOAD_COMPLETE được truyền thông bởi dịch vụ chạy ngầm DownloadMananager. Trong dữ liệu bắt được, cũng bao gồm tham số EXTRA_DOWNLOAD_ID chứa biến tham chiếu đến nội dung tải. Có thể xem đoạn code sau để hiểu thêm.

```
IntentFilter filter = new IntentFilter
(DownloadManager.ACTION_DOWNLOAD_COMPLETE);

BroadcastReceiver receiver = new BroadcastReceiver() {@Override

public void onReceive(Context context, Intent intent) {long id_reference =

intent.getLongExtra(DownloadManager.EXTRA_DOWNLOAD_ID, -1); Log.d("HTSI",
"File's id: " + id_reference);

}};  registerReceiver(receiver, filter);
```

1.3.2.3. Tùy chỉnh thông báo

- Tuỳ chỉnh thông báo: mặc định trên thanh trạng thái (Status Bar) của thiết bị sẽ xuất hiện một hộp thoại thông báo hoạt động hiện tại, hiển thị các thông tin như: tên tập tin được tải, ngày giờ tải, dung lượng đã tải…



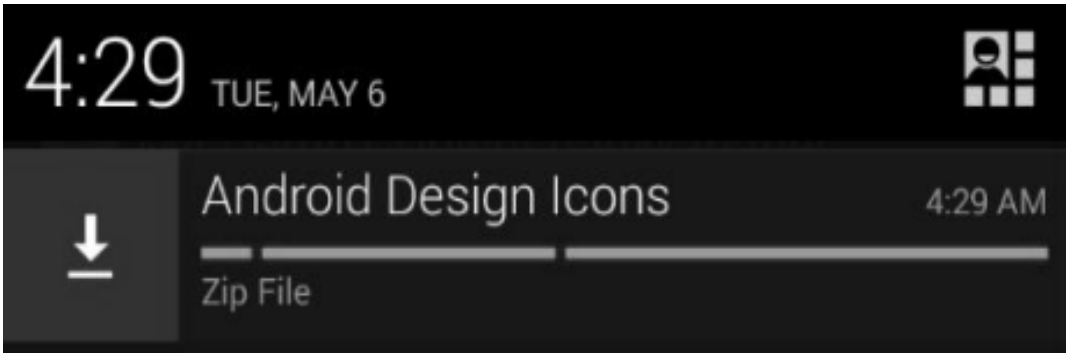
Hình 1.3 Thông báo nội dung tải

- Tuy nhiên chúng ta hoàn toàn có thể tùy chỉnh thông tin hiển thị trên hộp thoại thông báo này tường minh hơn thông qua các phương thức hỗ trợ trong đối tượng Request.

- Ví dụ:

```
request.setTitle("Android Design Icons");
request.setDescription("Zip File");
```

- Kết quả:



Hình 1.4 Kết quả nội dung tải

Hộp thoại thông báo mặc định sẽ xuất hiện khi bắt đầu tải và tự đóng khi hoàn tất quá trình tải, để thiết lập lại trạng thái này chúng ta có thể sử dụng phương thức setNotificationVisibility() với các biến cờ tương ứng sau:

- Request.VISIBILITY_VISIBLE: tham số mặc định, chỉ hiển thị trong quá trình tải.
- Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED: hiển thị trong quá trình tải và cả khi hoàn tất.
- Request.VISIBILITY_VISIBLE_NOTIFY_ONLY_COMPLETION: chỉ hiển thị khi hoàn tất quá trình tải.
- Request.VISIBILITY_HIDDEN: không hiển thị hộp thoại thông báo. Đối với biến cờ VISIBILITY_HIDDEN cần cấp quyền trong AndroidManifest.xml

```
<uses-permission
android:name="android.permission.DOWNLOAD_WITHOUT_NOTIFICATION"/>
```

1.3.2.4 Chỉ định nơi lưu trữ

- Mặc định các nội dung tải được hệ thống lưu trữ trong Content Providers của ứng dụng Download:

```
data/user/0/com.android.providers.downloads/cache/<Tên tập tin>
```

- Các tập tin trong thư mục này sẽ không thể truy xuất bằng đường dẫn của tập tin, chỉ có thể truy xuất thông qua Uri của tập tin đó.
- Tuy nhiên ta có thể thiết lập lại vị trí lưu trữ nội dung tải thông qua các phương thức sau:
  - setDestinationUri(Uri uri): chỉ định vị trí lưu trữ bất kỳ thông qua Uri.
    - Ví dụ:

```
File f = new File(Environment.getExternalStorageDirectory(),
uri.getLastPathSegment());
 request.setDestinationUri(Uri.fromFile(f));
```

- Đường dẫn lưu trữ

```
/storage/emulated/0/<file-name>
```

- setDestinationInExternalFilesDir(Context context, String dirType, String subPath): chỉ định vị trí lưu trữ trong thư mục ứng dụng trên bộ nhớ ngoài.
  - Ví dụ:

```
request.setDestinationInExternalFilesDir(this, "My Files",
uri.getLastPathSegment());
```

- Đường dẫn lưu trữ

```
/storage/emulated/0/Android/data/<pakage-name>/files/My Files/<file-name>
```

- setDestinationInExternalPublicDir(String dirType, String subPath): chỉ định vị trí lưu trữ trong thư mục ứng dụng trên bộ nhớ ngoài.
  - Ví dụ:

```
request.setDestinationInExternalPublicDir  (Environment.DIRECTORY_DOWNLOADS, uri.getLastPathSegment());
```

- Đường dẫn lưu trữ

```
/storage/emulated/0/Download/<file-name>
```

- Các hành động truy xuất đến bộ nhớ ngoài cần cấp quyền trong tập tin AndroidManifest.xml

```
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

1.3.2.5. Truy vấn nội dung tải

- Các cột dữ liệu quan trọng trong bảng dữ liệu Download.
  - COLUMN_ID
  - COLUMN_LOCAL_FILENAME
  - COLUMN_LOCAL_URI
  - COLUMN_MEDIA_TYPE
  - COLUMN_REASON
  - COLUMN_STATUS
  - COLUMN_BYTES_DOWNLOADED_SO_FAR
  - COLUMN_TOTAL_SIZE_BYTES
- Việc truy vấn thông qua đối tượng Query, dữ liệu trả về là đối tượng Cursor.
  - Ví dụ:

```
Query query = new Query(); query.setFilterById(id_reference); Cursor cursor =
```

```
manager.query(query); if (cursor.moveToFirst()) { String path =
cursor.getString(cursor.getColumnIndex
(DownloadManager.COLUMN_LOCAL_FILENAME)); if (path != null)
imageView.setImageURI(Uri.parse(path)); }
```

1.4. Web service Parse XML và Parser Json

1.4.1. Tìm hiểu về dịch vụ Web service

1.4.1.1. Web service là gì?

Là những thành phần ứng dụng dùng để chuyển đổi một ứng dụng thông thường sang một ứng dụng web. Đồng thời nó cũng xuất bản các chức năng của mình để mọi người dùng internet trên thế giới đều có thể sử dụng thông qua nền tảng web.

Web Serviece truyền thông bằng cách sử dụng các giao thức mở, tài nguyên phần mềm có thể xác định bằng địa chỉ URL, thực hiện các chức năng và đưa ra các thông tin người dùng yêu cầu, các ứng dụng độ lập và tự mô tả chính nó.

Nó bao gồm các modun độc lập cho hoạt động của khác hàng và doanh nghiệp và bản thân nó được thực thi trên server. Nền tảng cơ bản của WS là XML + HTTP. Bất cứ một ứng dụng nào cũng đều có thể có một thành phần WS. WS có thể được tạo ra bằng bất kỳ một ngôn ngữ lập trình nào.

Các Web Services cho phép các tổ chức thực hiện truyền thông dữ liệu mà không cần phải có kiến thức về hệ thống IT phía sau tường lửa. Một số Web Services hiện nay có sẵn miễn phí và càng ngày càng hướng dần vào các doanh nghiệp.

Một ví dụ về Web Service sẵn có là dịch vụ được cung cấp bởi PayPal cho phép những người có tài khoản có thể thanh toán hoặc trả một phần hoặc thực hiện các giao dịch tìm kiếm, và lấy lại các thông tin của từng giao dịch cụ thể.

1.4.1.2. Các chuẩn dịch vụ web

Việc tìm hiểu về các chuẩn dịch vụ WEB giúp chúng ta có thể đưa ra các lựa chọn phù hợp với hệ thống triển khai, dữ liệu và các thiết bị truy xuất dữ liệu đầu cuối đầu cuối. Có thể liệt kê một số chuẩn chính sau:

- UDDI (Universal Description, Discovery and Integrated).
- WSDL (Web Services Description Language).
- WSIL (Web Services Inspection Language).
- SOAP (Simple Object Access Protocol).

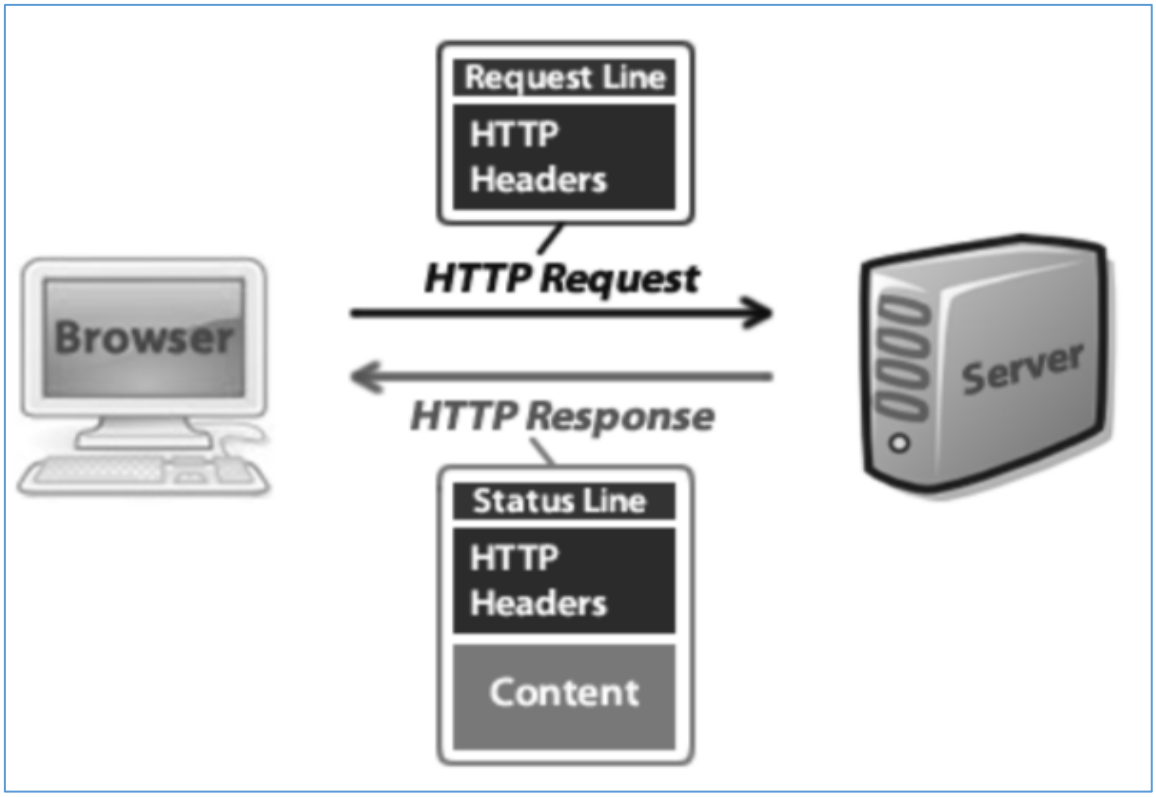
1.4.1.3. Các loại web service

1.4.1.3.1. Giao thức HTTP

- HTTP (Hypertext Transfer Protocol) là giao thức mạng cho phép các hệ thống thông

tin phân phối và cộng tác với nhau. HTTP là nên tảng giao tiếp dữ liệu cho WWW.

- HTTP hoạt động trên cơ chế giao thức request – response trong mô hình điện toán client – server.



Hình 1.5 Cơ chế hoạt động của HTTP

- Các định nghĩa đi kèm:
    - URI (Uniform Resource Identifier) là một chuỗi để xác định một tài nguyên trên internet.
    - URL (Uniform Resource Locator) là một URI cho biết sự tồn tại của một tài nguyên và cách thức để nhận tài nguyên đó.
    - URN (Uniform Resource Name) là một URI nhằm xác định tài nguyên bằng tên và độc lập với vị trí lưu trữ.
- HTTP status code
    - 2xx Success (Trạng thái thành công)
        - Ví dụ:
            - 200 – OK.
            - 201 – Created.
    - 3xx Redirection (Báo chuyển hướng)
        - Ví dụ:
            - 304 – Not Modified
    - 4xx Client Error (Báo lỗi client)
        - Ví dụ:

- 403 Forbidden
- 404 – Not Found

- 5xx Server Error (Báo lỗi server)
  - Ví dụ:
    - 503 – Service Unavailable
    - 504 – Gateway Timeout

1.4.1.3.2. SOAP

- SOAP là một giao thức dựa trên nền XML cho phéo ứng dụng trao đổi các thông tin thông qua HTTP. Nói cụ thể hơn thì SOAP là một giao thức dùng để truy cập các dịch vụ web.
- SOAP là viết tắt của Simple Object Access Protocol.
- SOAP là một giao thức truyền thông.
- SOAP là một định dạng dùng để gửi đi các thông điệp.
- SOAP được thiết kế để giao tiếp thông qua internet.
- SOAP là một nền tảng XML.
- SOAP rất đơn giản và có thể mở rộng.
- SOAP cho phép chúng ta vượt qua bức tường lửa (firewall)
- SOAP là một tiêu chuẩn W3C.
- SOAP là giao thức sử dụng XML để định nghĩa dữ liệu dạng thuần văn bản (plaint text) thông qua HTTP. SOAP là cách mà web service sử dụng để truyền tải dữ liệu. Vì dựa trên XML nên SOAP là một giao thức không phụ thuộc platform cũng như bất kì ngôn ngữ lập trình nào.
- Một thông điệp SOAP được chia thành hai phần là header và body. Phần header chỉ ra địa chỉ web service, host, Content-Type, Content-Length tương tự như một thông điệp HTTP.

1.4.1.3.2. REST

- REST (Representational State Transfer) đã được chọn sử dụng rộng rãi thay cho Web service dựa trên SOAP và WSDL. Bằng chứng quan trọng của sự thay đổi này chính là việc các công ty dẫn đầu trong lĩnh vực cung cấp dịch vụ mạng 2.0 như Yahoo, Google và Facebook đã phản đối các giao thức dựa trên SOAP hoặc WSDL và ủng hộ phương thức hướng đến tài nguyên và dễ sử dụng đối với các dịch vụ của họ. Trong bài viết này, chúng ta sẽ tìm hiểu các nguyên lý cơ bản của REST.

- REST (Representational State Transfer) là một kiểu kiến trúc phần mềm cho các hệ thống phân tán siêu truyền thông như là WWW.
- Đặc trưng của REST
    - Là dạng client – server.
    - Phân tách giao diện của client ra khỏi dữ liệu.
    - Cho phép mỗi thành phần phát triển độc lập.
    - Hỗ trợ đa nền tảng.
- Mỗi yêu cầu từ client phải có đủ thông tin cần thiết để server có thể hiểu được mà không cần phải lưu trữ thêm thông tin nào trước đó.
- Tất cả tài nguyên được truy cập thông qua một interface thống nhất (HTTP
    - GET, PUT, POST, DELETE, ...).

1.4.1.3.3. RESTful Service

- Là một web service đơn giản sử dụng HTTP và tính chất của REST.
- Là một tập tài nguyên các thành phần được định nghĩa:
    - URI gốc cho web service.
    - MIME type hỗ trợ bởi web service.
    - Tập hành động hỗ trợ bởi web service sử dụng HTTP method (GET, POST, PUT, DELETE).

1.4.2. Parser XML

1.4.2.1. Định dạng XML

- XML (viết tắt từ tiếng Anh eXtensible Markup Language, "Ngôn ngữ Đánh dấu Mở rộng") là ngôn ngữ đánh dấu với mục đích chung do W3C đề nghị, để tạo ra các ngôn ngữ đánh dấu khác. Đây là một tập con đơn giản của SGML, có khả năng mô tả nhiều loại dữ liệu 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. Các ngôn ngữ dựa trên XML (thí dụ: RDF, RSS, MathML, XHTML, SVG, GML và 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.

- Cú pháp sơ lược:

```
<tên thẻ thuộc tính= “Giá trị”>Nội dung thẻ</tên thẻ>
```

- Ví dụ:

```
<item>
    <title>Tên</title>
    <description>Nội dung</description>
```

```
</item>
```

1.4.2.2. Đọc ghi dữ liệu XML

- Để thực hiện đọc dữ liệu từ XML trong Android, ta có thể sử dụng DOM (Document Object Model) Parser hoặc XML Pull Parser.

1.4.2.3. Ghi dữ liệu XML

- **DOM Parser**: giao diện lập trình ứng dụng (API) có dạng một cây cấu trúc dữ liệu, các đối tượng cần khởi tạo khi sử dụng:
  - **Element**: đại diện cho một thẻ trong XML.
  - **Document**: tập tin tài liệu được khởi tạo từ dữ liệu XML thông qua DocumentBuilder.
  - **DocumentBuilder**: đối tượng hỗ trợ chuyển đổi dữ liệu XML thành cấu trúc tập tin XML cho việc đọc ghi dữ liệu.
  - **DocumentBuilderFactory**: khởi tạo đối tượng DocumentBuilder.
  - **Transformer**: đối tượng cho phép thực hiện chuyển đổi dữ liệu sang nhiều dạng XML khác nhau.
  - **TransformerFactory**: khởi tạo đối tượng Transformer.
  - **DOMSource**.
- **XML Pull Parser**: cho phép trình bày các thành phần trong tập tin theo dạng chuỗi các thẻ (tag) và các đánh dấu (event), để làm việc với XML Pull Parser cần khảo sát các thuộc tính và các đối tượng sau:
  - **XmlPullParserFactory**: khởi tạo đối tượng XmlPullParser từ tập tin tài liệu XML.
  - **XmlPullParer**: đối tượng kiểm soát việc duyệt và truy xuất dữ liệu.
  - **START_DOCUMENT**: điểm đánh dấu bắt đầu của tập tin XML.
  - **END_DOCUMENT**: điểm đánh dấu kết thúc của tập tin XML.
  - **START_TAG**: điểm đánh dấu bắt đầu cặp thẻ XML.
  - **END_TAG**: điểm đánh dấu kết thúc cặp thẻ XML.

1.4.3. Parse JSON

1.4.3.1. Định dạng JSON

- JSON (JavaScript Object Notation) được định nghĩa dữ theo ngôn ngữ JavaScript, tiêu chuẩn ECMA-262 năm 1999, cấu trúc là một định dạng văn bản đơn giản với các trường dữ liệu được lồng vào nhau. JSON được sử dụng để trao đổi dữ liệu giữa

các thành phần của một hệ thống tương thích với hầu hết các ngôn ngữ C, C++, C#, Java, JavaScript, Perl, Python...

- JSON được xây dựng dựa trên hai cấu trúc chính:
    - Tập hợp cặp giá trị name/value, trong nhiều ngôn ngữ khác nhau cặp giá trị này có thể là object, record, struct, dictionary, hash table, keyed list...
    - Tập hợp danh sách các giá trị, có thể là array, vector, list hay sequence.
- Tuỳ thuộc vào dữ liệu cần trao đổi, JSON có thể có nhiều dạng khác nhau, tuy nhiên có thể tổng hợp ở những hai dạng chính sau:
    - Một đối tượng Object chứa các cặp giá trị string/value không cần thứ tự, được bao trong cặp "{}", các giá trị bên trong được định dạng "string:value" và chia cách nhau bởi dấu ",". Value ở đây có thể là chuỗi, số, true- false, null...
- Ví dụ

```
{
        "username" : "thehalfheart",
        "email" : "thehalfehart@gmail.com",
}
```

1.4.3.2. Đọc ghi dữ liệu JSON

- Việc thực hiện đọc ghi dữ liệu JSON trong Android có thể thông qua nhiều thư viện khác nhau như GSON, Json.Smart, Jackson, ... Tuy nhiên, trong tài liệu này, chúng ta sẽ khảo sát các lớp JSON trong gói org.json được tích hợp sẵn trong Android SDK. Trong gói này bao gồm bốn lớp chính:
    - **JSONObject**: đối tượng quản lý JSON ở dạng một Object.
    - **JSONArray**: đối tượng quản lý JSON ở dạng tập hợn các Object hoặc Array.
    - **JSONStringer**: đối tượng chuyển dữ liệu JSON thành dạng chuỗi.
    - **JSONTokener**: chuyển đổi đối tượng JSON (chuẩn RFC-4627) mã hoá chuỗi
- một thành đối tượng tương ứng.

1.4.3.3. Ghi dữ liệu JSON

- Để thực hiện ghi dữ liệu JSON, cần xác định rõ cấu trúc của dữ liệu cần lưu trữ. Nếu dữ liệu cần ghi là một đối tượng, dữ liệu sẽ được ghi vào một JSONObject; nếu dữ liệu là một mảng, dữ liệu sẽ được ghi vào một JSONArray.
- Ví dụ 1: với dạng JSONObject

```
{
  "name":"Nhat",
```

```
    // Khởi tạo đối tượng JSONObject
    JSONObject jsonObject = new JSONObject();

    // Thiết lập các cặp giá trị name/value
    jsonObject.put("name", "Nhat");

    // Chuyển dữ liệu thành chuỗi để sử dụng
    String jsonString = jsonObject.toString();
}
```

- Ví dụ 2: với dạng JSONArray

```
    // Khởi tạo đối tượng JSONArray
    JSONArray jsonArray = new JSONArray();

    // Khởi tạo và đặt giá trị đối tượng JSONObject "source"
    JSONObject jsonObjectSource = new JSONObject();
    jsonObjectSource.put("source", "vnexpress.vn");
    // Khởi tạo đối tượng JSONArray "data"
    JSONArray jsonArrayData = new JSONArray();
    // Khởi tạo và đặt giá trị phần tử JSONObject
    JSONObject jsonObjectDate1 = new JSONObject(); jsonObjectDate1.put("date",
    "17-06-2015");
    jsonObjectDate1.put("tempMaxC", 13);
    // Đặt giá trị vào mảng "data"
    jsonArrayData.put(jsonObjectDate1);
    // Đặt giá trị mảng vào "data"
    JSONObject jsonObjectData = new JSONObject();
    jsonObjectData.put("data", jsonArrayData);
    // Đặt giá trị vào mảng
    jsonArray.put(jsonObjectSource);
    jsonArray.put(jsonObjectData);
    // Xuất giá trị mảng
    Log.d("HTSI", jsonArray.toString());
```

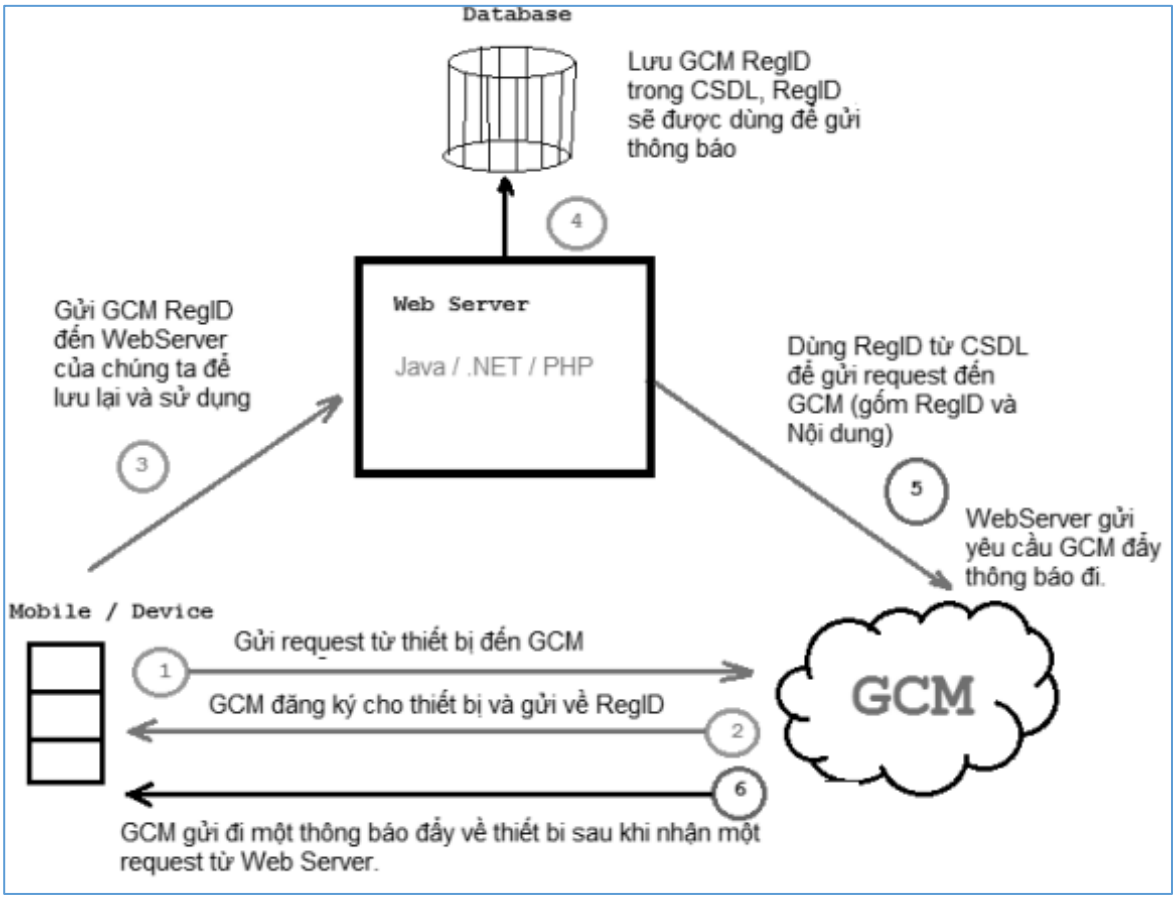
1.5. Google Cloud Message

1.5.1. Giới thiệu

- Google Cloud Messaging (GCM) là một dịch vụ cho phép gửi dữ liệu từ máy chủ của bạn đến các thiết bị Android của người dùng, và ngược lại.
- GCM được sử dụng cho hoạt động giao dịch giữa các ứng dụng và máy chủ hỗ trợ đầu cuối. Cloud Messaging hiện đang được tích hợp vào Google Play Services và có những cải tiến mạnh mẽ.

Hình 1.6 Logo của Google Cloud Message

- GCM là hoàn toàn miễn phí và không giới hạn băng thông. Dịch vụ hoạt động trên các gói dữ liệu có dung lượng nhỏ hơn 4kb và tin nhắn tới thiết bị Android là tức thời (Push-notification).
- GCM là phù hợp cho việc xây dựng các ứng dụng nhắn tin tức thời hoặc tương tác giữa người dùng và nhà phát triển ứng dụng.
- Ví dụ:
  - Thông báo tới thiết bị Android của người khi có một email mới. GCM gửi thông báo tới người quản lý khi có một đơn đặt hàng trên trang Web của họ.
- Gửi tin nhắn tới cả một cộng đồng như: cồng đồng nhân viên thuộc cùng một công ty, học sinh hoặc phụ huynh của một trường học với chi phí thấp nhất và hiệu quả nhất.
- Nguyên tắc hoạt động: Hãy cùng xem hình dưới để hiểu rõ hơn.



Hình 1.7 Nguyên tắc hoạt động khi Web Service gửi thông báo về thiết bị của GCM

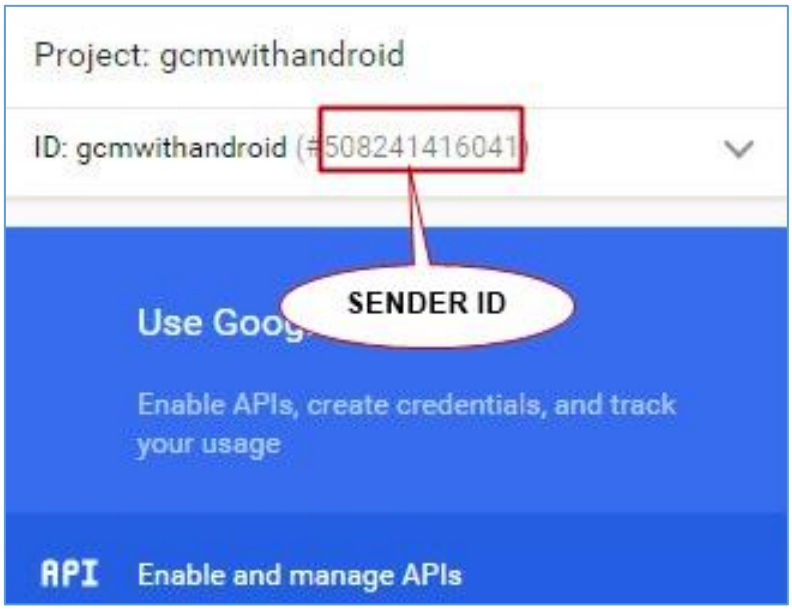
1.5.2. Cấu hình cho Google Cloud Message

1.5.2.1. Đăng ký dịch vụ GCM

- Đầu tiên, bạn cần đăng ký Google Cloud Messaging bằng tài khoản Google của bạn.
  - Truy cập Google Console tại: https://cloud.google.com/console/project
  - Nếu bạn chưa tạo một dự án API nào, nhấp vào “Create Project”.

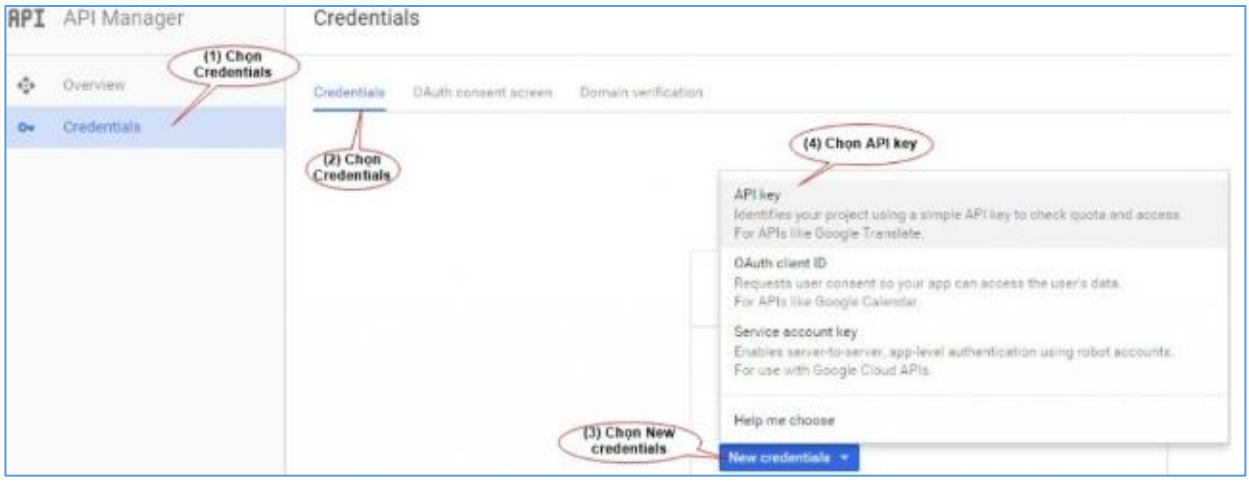
Hình 1.8 Tạo một project trên Google Developers Console

- Điền tên project và chọn create
- Kết quả tạo xong một project và nhận Sender ID



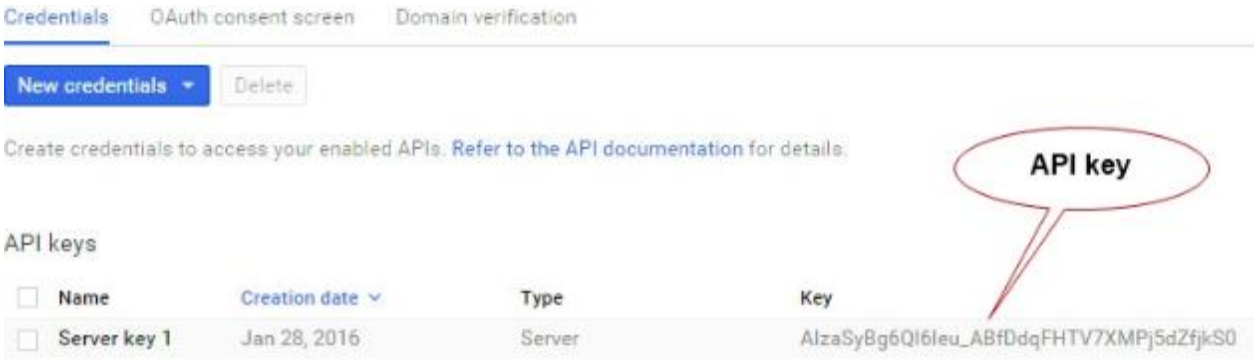
Hình 1.9 Nhận Sender ID khi tạo xong project

- Đăng ký **API Key**
- Chọn **Credentials** -> chọn **API key**



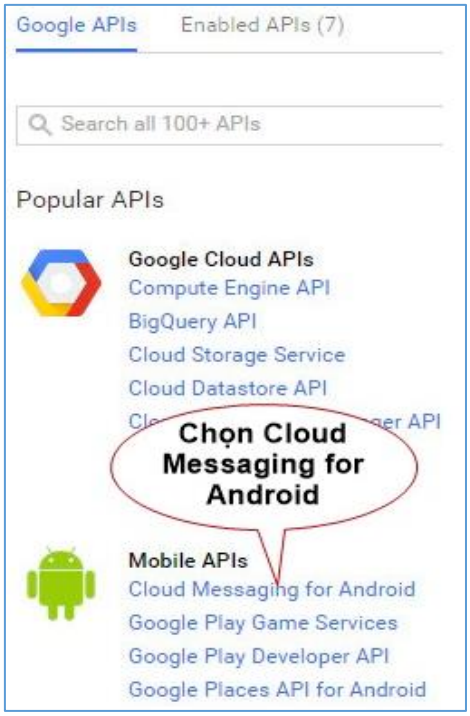
Hình 1.10 Tạo Credentials cho project

- Tiếp đến, chọn **Android Key -> Server Key**
- Điền tên -> Chọn Create
- API Key sau khi tạo thành công



Hình 1.11 API key trong project

- Kích hoạt dịch vụ Cloud Messaging for Android
- Chọn **Google APIs** -> chọn **Cloud Messaging for Android**



Hình 1.12 Kích hoạt GCM cho project của bạn 1

- Chọn **Enable API**

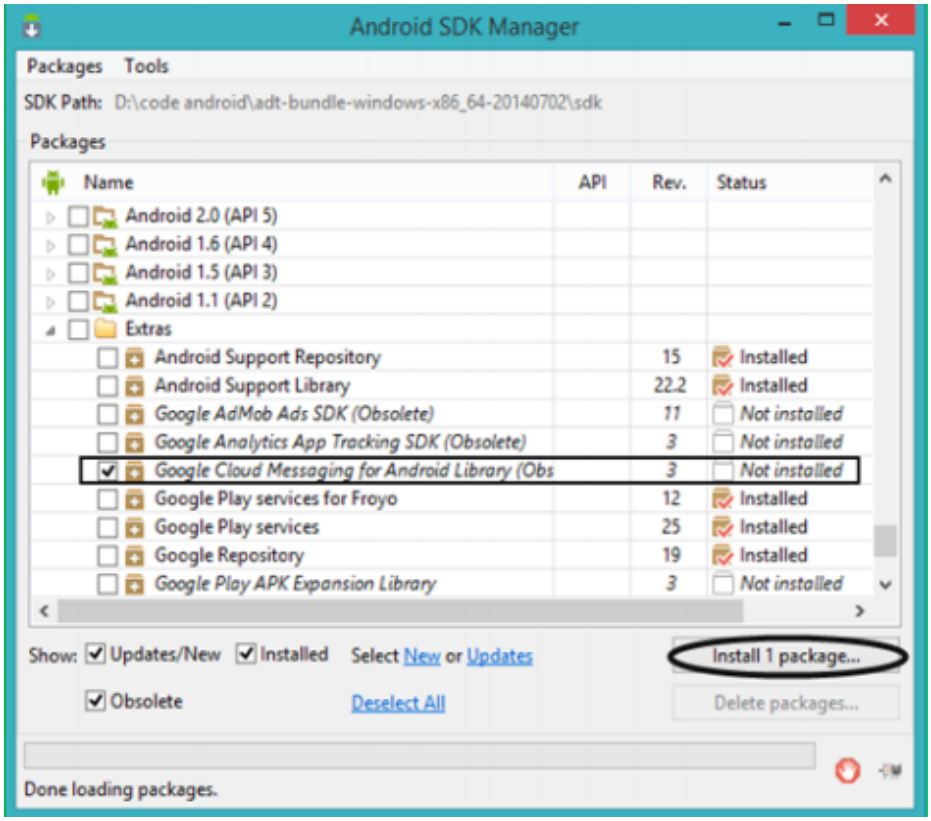
Hình 1.13 Kích hoạt GCM cho project của bạn 2

- Mở tập tin appengine-web.xml và tìm đến dòng lệnh bên dưới và thay thế "YOUR_KEY_HERE" là API key ở trên:

```
<property name="gcm.api.key" value="YOUR_KEY_HERE" />
```

1.5.2.2 Cấu hình môi trường hoạt động cho GCM

- Download thư viện hỗ trợ GCM
- Mở SDK Manager lên và tiến hành tải thư viện hỗ trợ GCM.
- Vào mục Extras → Chọn mục Google Cloud Messaging for Android Library:



Hình 1.14 Tải GCM helper library

1.6. TelePhone & SMS

1.6.1 TelePhone

Tạo cuộc gọi bằng các sử dụng Intent để gọi Dialer có sẵn trong thiết bị.

Ví dụ:

```
Intent intent = new Intent(Intetn.ACTION_DIAL, Uri.parse(“tel:123456”));
startActivity(intent);
```

- Thực hiện hành động gọi ACTION_CALL bằng cách nhấn nút Call trong thiết bị.
- Một số thông tin cần thiết cho việc quản lý chức năng Phone trên thiết bị:
  - PhoneType (GSM - CDMA) o UniqueID (IMEI – MIED)
  - Sofware version o Number
- Để truy xuất được các thông tin này cần thực hiện đăng ký trong AndroidManifest.xml

```
<u<users-permission android:name=
“android.permission.READ_PHONE_STATE”/>
```

- Việc truy xuất quản lý bởi lớp TelephoneManager với phương thức getSystemService().
- Ví dụ:

```
String srvcName = Context.TELEPHONY_SERVICE; TelephonyManager
telephonyManager = (TelephonyManager)getSystemService(srvcName);
int phoneType = telephonyManager.getPhoneType();
```

```
switch (phoneType) {
        case (TelephonyManager.PHONE_TYPE_CDMA): break;
        case (TelephonyManager.PHONE_TYPE_GSM) : break;
        case (TelephonyManager.PHONE_TYPE_NONE): break;
        default: break;
}
String deviceId = telephonyManager.getDeviceId();
String softwareVersion = telephonyManager.getDeviceSoftwareVersion();
String phoneNumber = telephonyManager.getLine1Number();
```

- Quản lí trạng thái:
- Ví dụ:

```
class MyPhoneStateListener extends PhoneStateListener {
@Override  public void onCallStateChanged(int state, String incomingNumber) {
super.onCallStateChanged(state, incomingNumber);
switch(state)  {
        case TelephonyManager.CALL_STATE_IDLE: break;
        case TelephonyManager.CALL_STATE_RINGING: break;
        case TelephonyManager.CALL_STATE_OFFHOOK: break;
        default: break;
} }
}
```

- Đọc dữ liệu kết nối và các thay đổi trạng thái thông qua các phương thức getDataSet() và getDataActivity()

```
int dataActivity = telephonyManager.getDataActivity(); int dataState =
telephonyManager.getDataState(); switch (dataActivity)  {
      case TelephonyManager.DATA_ACTIVITY_INT : break;
      case TelephonyManager.DATA_ACTIVITY_OUT : break;
      case TelephonyManager.DATA_ACTIVITY_INOUT : break;
      case TelephonyManager.DATA_ACTIVITY_NONE : break;
}
```

- Đọc các dữ liệu Network cần thiết:

```
String networkCountry = telephonyManager.getNetworkCountryIso();
String networkOperatorId = telephonyManager.getNetworkOperator();
String networkName = telephonyManager.getNetworkOperatorName();
int networkType = telephonyManager.getNetworkType();
switch (networkType) {
            case (TelephonyManager.NETWORK_TYPE_1xRTT) : break;
            case (TelephonyManager.NETWORK_TYPE_CDMA) : break;
            case (TelephonyManager.NETWORK_TYPE_EDGE) : break;
            case (TelephonyManager.NETWORK_TYPE_EVDO_0) : break;
            case (TelephonyManager.NETWORK_TYPE_EVDO_A) : break;
            case (TelephonyManager.NETWORK_TYPE_GPRS) : break;
```

```
            case (TelephonyManager.NETWORK_TYPE_HSDPA) : break;
            case (TelephonyManager.NETWORK_TYPE_HSPA) : break;
            case (TelephonyManager.NETWORK_TYPE_HSUPA) : break;
            case (TelephonyManager.NETWORK_TYPE_UMTS) : break;
            case (TelephonyManager.NETWORK_TYPE_UNKNOWN) : break;
            default: break; }
}
```

- Đọc các dữ liệu SIM:

```
int simState = telephonyManager.getSimState(); switch (simState) {
case (TelephonyManager.SIM_STATE_ABSENT): break;  case
(TelephonyManager.SIM_STATE_NETWORK_LOCKED): break;
        case (TelephonyManager.SIM_STATE_PIN_REQUIRED): break;
        case (TelephonyManager.SIM_STATE_PUK_REQUIRED): break;
        case (TelephonyManager.SIM_STATE_UNKNOWN): break;
        case (TelephonyManager.SIM_STATE_READY): {
String simCountry = telephonyManager.getSimCountryIso();
String simOperatorCode = telephonyManager.getSimOperator();
String simOperatorName =telephonyManager.getSimOperatorName();
String simSerial = telephonyManager.getSimSerialNumber();
break;
} }
default: break;
```

- Các thay đổi trạng thái Phone được quản lý bởi lớp PhoneStateListener

```
PhoneStateListener phoneStateListener = new PhoneStateListener() {
        public void onCallForwardingIndicatorChanged(boolean cfi) {}
        public void onCallStateChanged(int state, String incomingNumber) {}
        public void onCellLocationChanged(CellLocation location) {}
        public void onDataActivity(int direction) {}
        public void onDataConnectionStateChanged(int state) {}
        public void onMessageWaitingIndicatorChanged(boolean mwi) {}
        public void onServiceStateChanged(ServiceState serviceState) {}
        public void onSignalStrengthChanged(int asu) {}
}
```

- Hủy đăng ký bằng LISTEN_NONE

```
telephonyManager.listen(phoneStateListener, PhoneStateListener.LISTEN_NONE);
```

- Xây dựng ứng dụng thay thế ứng dụng cuộc gọi có sẵn trong ứng dụng. Bao gồm 2 bước:

o Chặn những Intent đến của ứng dụng Dialer và xử lý.

o Tổ chức quản lí các cuộc gọi ra bên ngoài. Cần đăng ký các Intent cho Activity xử lý

o Intent.ACTION_CALL_BUTTON o Intent.ACTION_DIAL

o Intent.ACTION_VIEW

- Ví dụ đăng kí trong Intent trong AndroidManifest

```
<activity  android:name= ".MyDialerActivity" android:label ="@string/app_name">
<intent-filter>  <action android:name = "android.intent.action.CALL_BUTTON"/>
<category android:name = "android.intent.category.DEFAULT"/
</intent-filter>
<intent-filter>
<action android:name = "android.intent.action.VIEW"/>  <action android:name =
"android.intent.action.DIAL" /> <category android:name =
"android.intent.category.DEFAULT" /> <category android:name =
"android.intent.category.BROWSANLE" /> <data android:scheme = "tel" />
</intent-filter>
</activity>
```

- Ví dụ xử lý cuộc gọi tới

```
phoneStateListner callStateListener = new PhoneStateListner()
{
public void onCallStateChanged( int state, String incomingNumber)
 {  }
};
telephonyManager.listen(callStateListener,
phoneStateListener.LISTEN_CALL_STATE;
```

1.6.2. SMS

- Gửi tin nhắn bằng cách tạo một Intent để gọi ứng dụng Message trong thiết bị.
- Ví dụ:
    - Tạo tin nhắn SMS

```
Intent smsIntent = new Intent(Intent.ACTION_SENDTO,Uri.parse("sms:123456"));
smsIntent.putExtra("sms_body", "Press send to send me"); startActivity(smsIntent);
```

- Gửi tin nhắn MMS có chứa tập tin Media.
- Ví dụ:

```
//Tạo tin nhắn MMS
Uri attached_Uri = Uri.parse("content://media/external/images/media/1"); Intent
mmsIntent = new Intent(Intent.ACTION_SEND,attached_Uri);
mmsIntent.putExtra("sms_body", "Please see the attached image");
mmsIntent.putExtra("address", "07912355432");
mmsIntent.putExtra(Intent.EXTRA_STREAM, attached_Uri);
mmsIntent.setType("image/png");
startActivity(mmsIntent);
```

- SMS tin nhắn được điều khiển bởi SmsManager
    - Tạo đối tượng SmsManager

```
SmsManager smsManager = SmsManager.getDefault();
```

    - Yêu cầu quyền truy cập khi gửi tin nhắn

```
<uses-permission android:name= "android.permission.SEND_SMS"/>
```

- Việc gửi tin nhắn được thực hiện bởi phương thức sendTextMessage của lớp

SmsManager

- Cú pháp:

```
sendTextMessage(Stringdestination Address,
        String scAddress,
        String text,
        PendingIntent sentIntent,
        PendingIntent deliveryIntent)
```

- Ví dụ:

```
String sendTo = “03149”;
String myMessage = “Android supports programmatic SMS messaging!”;
smsManager.sendTextMessage(sendTo, null, myMessage, null, null);
```

- Theo dõi, xác nhận SMS tin nhắn đã được gửi bằng cách đăng ký BroadcastReceiver để lắng nghe các hoạt động từ Pending Itent được tạo khi gửi tin nhắn. o sentItent được gọi lên khi tin nhắn được gửi thành công hay lỗi.
  - Activity.RESULT_OK.
  - SmsManager.RESULT_ERROR_GENERIC_FAILURE.
  - SmsManager.RESULT_ERROR_RADIO_OFF.
  - SmsManager.RESULT_ERROR_NULL_PDU.
  - deliveryIntent được gọi lên một khi thiết bị đích nhận được tin nhắn.
- -Ví dụ về kiểm soát gửi tin nhắn
  - Tạo Pending Intent

```
String SENT_SMS_ACTION = "SENT_SMS_ACTION " ;
String DELIVERED_SMS_ACTION = "DELIVERED_SMS_ACTION " ;
Intent sentIntent = new Intent(SENT_SMS_ACTION);
PendingIntent sentPI =
PendingIntent.getBroadcast(getApplicationContext(),0,sentIntent,0);
Intent deliveryIntent = new Intent(DELIVERED_SMS_ACTION);
PendingIntent deliverPI = PendingIntent.getBroadcast(getApplicationContext(),0,
deliveryIntent,0);
```

  - Tạo Broadcast Receiver cho Send_Action

```
registerReceiver(new BroadcastReceiver() { @Override
public void onReceive(Context _context, Intent _intent){ switch (getResultCode()) {
        case Activity.RESULT_OK: break;
        case SmsManager.RESULT_ERROR_GENERIC_FAILURE: break;
        case SmsManager.RESULT_ERROR_RADIO_OFF: break;
        case SmsManager.RESULT_ERROR_NULL_PDU:break;
        } }
},  new IntentFilter(SENT_SMS_ACTION));
```

  - Tạo Broadcast Receiver cho Delivery_Action

```
registerReceiver(new BroadcastReceiver() { @Override
public void onReceive(Context _context, Intent _intent){ [ ... SMS delivered actions ...
```

```
]
} },
new IntentFilter(DELIVERED_SMS_ACTION));
smsManager.sendTextMessage(sendTo, null, myMessage, sentPI, deliverPI);
```

- Xử lý các tin nhắn dài hơn 160 ký tự bằng phương thức divideMessage() và sendMultipartTextMessage().
    - Cú pháp:

```
sendMultipartTextMessage(Stringdestination Address, String scAddress,
ArrayList<String> parts, ArrayList<PendingIntent> sentIntents,
ArrayList<PendingIntent> deliveryIntents)
```

    - Ví dụ:

```
ArrayList<String> messageArray = smsManager.divideMessage(myMessage);
ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(); for (int i =
0; i < messageArray.size(); i++)
sentIntents.add(sentPI);
smsManager.sendMultipartTextMessage(sendTo,null,messageArray, sentIntents,null);
```

- Gửi tin nhắn chứa dữ liệu bằng mảng byte o Cú pháp:
    - Vídụ:

```
sendDataMessage(Stringdestination Address, String scAddress, short destination Port,
byte[] data,
PendingIntent sentIntent, PendingIntent deliveryIntent)
Intent sentIntent = new Intent(SENT_SMS_ACTION); PendingIntent sentPI =
PendingIntent.getBroadcast(getApplicationContext(),0,sentIntent,0); short
destinationPort = 80;  byte[] data = [... your data ... ];
smsManager.sendDataMessage(sendTo,null,destinationPort,data, sentPI,null);
```

- Để ứng dụng nhận được SMS, cần đăng ký:
    - BroadcastReceiver với action

```
<receiver android:name="MySMSMonitor"> <intent-filter>  <action
android:name="android.provider.Telephony.SMS_RECEIVED"/> </intent-filter>
</receiver>
```

- Cung cấp quyền truy cập nhận SMS tin nhắn

```
<uses-permission android:name= “android.permission.RECEIVE_SMS”/>
```

- Làm việc với các thư mục tin nhắn, cần đăng ký quyền trong AndroidManifest.xml

```
<uses-permission android:name=”android.permission.READ_SMS”/>
```

- Thực hiện truy vấn để lấy ra các thông tin về tin nhắn trong các thư mục với URI tương ứng:
    - All: content://sms/all
    - Inbox : content://sms/inbox
    - Sent: content://sms/sent

- Draft: content://sms/draft
- Outbox: content://sms/outbox
- Failed : content://sms/failed
- Queued: content://sms/queued
- Undelivered: content://sms/undelivered
- Conversations: content://sms/conversations

1.7. SENSOR

1.7.1. Giới thiệu về cảm biến

- Cảm biến là phần cứng được tích hợp trên các thiết bị, có tác dụng phản hồi lại các hành động ở thế giới thực vào trong môi trường ứng dụng chạy trên thiết bị đó.
- Các hành động trong cảm biến được thực hiện một chiều và chúng chỉ cho phép thực hiện các hành động mặc định sẵn (trừ NFC).
- GPS cũng là một bộ cảm biến nhưng không được tích hợp vào nền tảng cảm biến trong Android. Các loại cảm biến có thể có trong một thiết bị:
  - Light Sensor.
  - Proximity Sensor.
  - Temparature Sensor.
  - Pressure Sensor.
  - Gyroscope Sensor.
  - Accelerometer Sensor.
  - Magnetic Field Sensor.
  - Orientation Sensor.
  - Gravity Sensor (Android 2.3).
  - Linear Accelerometer Sensor (Android 2.3).
  - Rotation Vector Sensor (Android 2.3).
  - Near Field Communication Sensor (Android 2.3).
- Có 2 cách để nhận biết thiết bị hổ trợ những loại cảm biến nào:
  - Sử dụng đối tượng thuộc lớp SensorManager để thực hiện thao tác lấy về danh sách các loại cảm biến trên thiết bị.
  - Thiết lập trong tập tin AndroidManifest để chỉ định các tính năng thiết bị cần có cho ứng dụng.

```
uses-feature android:name= “android.hardware.sensor.proximity”/>
```

1.7.2. Lấy thông tin và điều khiển cảm biến

- Cần đăng ký một bộ lắng nghe (Listener) sự thay đổi của cảm biến để thực hiện lấy các thông tin.
- Override các onResume và onPause để thực hiện các thiết lập trên cảm biến.

- Trong phương thức registrerListener() ta thiết lập thông số để nắm bắt các giá trị thay đổi của cảm biến.
    - SENSOR_DELAY_NORMAL
    - SENSOR_DELAY_UI
    - SENSOR_DELAY_GAME
    - SENSOR_DELAY_FASTEST
- Theo dõi các hoạt động của cảm biến
- Ví dụ: thực hiện callback 2 phương thức onAccuracyChanged() và onSensorChanged().

```
public void onAccuracyChanged(Sensor sensor, int accuracy) {
msg.insert(0, sensor.getName() + “accuracy changed: “ + accuracy + (accuracy +
(accuracy==1?” (LOW)” : (accuracy == 2?” (MED)”: “(HIGH)”)) + “\n”);
    text.setText(msg);
    text.invalidate();
}
public void onSensorChangeed(SensorEvent event) {
msg.insert(0, “Got a sensor event: “ + event.values[0] + “SI lux units\n”);
    text.setText(msg);
    text.invalidate();}
```

1.8.Firebse

1.8.1. Giới thiệu về Firebase

- **Firebase** là một dịch vụ cơ sở dữ liệu thời gian thực hoạt động trên nền tảng đám mây được cung cấp bởi Google nhằm giúp các lập trình phát triển nhanh các ứng dụng bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu.

1.8.2. Các chức năng chính của Firebase

- Với Google Firebase, bạn chỉ có thể tạo ra các ứng dụng chat như Yahoo Message của ngày xưa hoặc như Facebook Messager của ngày nay trong thời gian cực ngắn như khoảng một ngày thậm chí là vài giờ bởi đơn giản là bạn chỉ cần lo phần client còn phần server và database đã có firebase lo.
- Firebase là sự kết hợp giữa nền tảng cloud với hệ thống máy chủ cực kì mạnh mẽ tới từ Google, để cung cấp cho chúng ta những API đơn giản, mạnh mẽ và đa nền tảng trong việc quản lý, sử dụng database. Cụ thể hơn Google Firebase cung cấp tới chúng ta những chức năng chính sau:

1.8.2.1. Realtime Database – Cơ sở dữ liệu thời gian thực

- Firebase lưu trữ dữ liệu database dưới dạng JSON và thực hiện đồng bộ database tới tất cả các client theo thời gian thực. Cụ thể hơn là bạn có thể xây dựng được client đa

nền tảng (cross-platform client) và tất cả các client này sẽ cùng sử dụng chung 1 database đến từ Firebase và có thể tự động cập nhật mỗi khi dữ liệu trong database được thêm mới hoặc sửa đổi.

- Ngoài ra Firebase còn cho phép bạn phân quyền một các đơn giản bằng cú pháp tương tự như javascript.

1.8.2.2.Firebase Authentication – Hệ thống xác thực của Firebase

- Với Firebase bạn có thể dễ dàng tích hợp các công nghệ xác thực của Google, Facebook, Twitter, … hoặc một hệ thống xác thực mà bạn tự mình tạo ra vào trong ứng dụng của bạn ở bất kì nền tảng nào như Android, iOS hoặc Web.

1.8.2.3. Firebase storage

- Với firebase storage các lập trình viên có thể lưu trữ dữ liệu trực tiếp trên server của firebase như: hình ảnh, video, tập tin một cách dễ dàng.
- Firebase còn bổ sung Google security để tải lên và tải về các ứng dụng firebase của bạn. Bạn có thể sử dụng nó để lưu trữ hình ảnh, âm thanh, video hoặc nội dung do người dùng tạo ra .
- Đây là một dịch vụ lưu trữ đối tượng mạnh mẽ, đơn giản và hiệu quả chi phí.

1.8.2.4. Firebase Hosting

- Firebase hosting cung cấp nhanh chóng và an toàn cho việc lưu trữ ứng dụng web của bạn. Với Hosting, bạn có thể nhanh chóng và dễ dàng triển khai các ứng dụng web và các nội dung tĩnh cho một mạng nội dung phân phối toàn cầu (CDN) với một lệnh duy nhất.
- Điểm nổi bật:

Server với kết nối an toàn	Cấu hình SSL được thêm vào Firebase Hosting nên nội dung luôn được phân phối một cách an toàn.
Tốc độ cao	Mỗi tập tin bạn tải lên sẽ được lưu trữ trên ổ SSD tại CDN . Chính vì vậy mànội dung được cung cấp nhanh chóng.
Triển khai nhanh chóng	Sử dụng Firebase CLI, bạn có thể có được ứng dụng và chạy trong vài giây. Chỉ với một vài lệnh command line để thêm vào ứng dụng của bạn.

One-click rollbacks	Triển khai nhanh chóng là rất tốt, nhưng có thể lùi lại sai lầm thậm chí còn tốt hơn. Firebase Hosting cung cấp versioning đầy đủ cho phép rollbacks một cú nhấp chuột.

- Các ứng dụng sẽ được cấp 1 tên miền dạng *.firebaseapp.com hoặc bạn có thể trả tiền để sử dụng tên miền của riêng mình.

1.8.2.5. Firebase Cloud Messaging

- Không chỉ vậy, hiện nay firebase còn cung cấp dịch vụ Cloud Messaging thay thế cho Google Cloud Messaging đã cũ kỹ.
- Một giải pháp đa nền tảng, đáng tin cậy cho phép bạn cung cấp các thông điệp miễn phí dạng tin nhắn đến người dùng.
- Sử dụng FCM, bạn có thể thông báo cho một ứng dụng client email mới hoặc dữ liệu khác có sẵn để đồng bộ hóa. Bạn có thể gửi tin nhắn thông báo cho lái xe reengagement sử dụng và duy trì. Đối với trường hợp sử dụng như tin nhắn tức thời, tin nhắn có thể chuyển một tải trọng lên đến 4KB để một ứng dụng client.

1.8.3. Những lợi ích từ việc sử dụng Google Firebase

- Ở phía trên là các chức năng của google firebase, vậy các chức năng đó sẽ đem lại cho bạn những lợi ích gì, có lẽ một số bạn đã mường tượng ra rồi nhưng cũng có bạn có lẽ vẫn còn mơ hồ vì vậy hãy đọc kĩ phần này sẽ biết câu trả lời chính xác nhất.

1.8.3.1. Triển khai ứng dụng cực nhanh

- Với Firebase bạn có thể giảm bớt rất nhiều thời gian cho việc viết các dòng code để quản lý và đồng bộ cơ sở dữ liệu, mọi việc sẽ diễn ra hoàn toàn tự động với các API của Firebase. - Không chỉ có vậy Firebase còn hỗ trợ đã nền tảng nên bạn sẽ càng đỡ mất thời gian rất nhiều khi ứng dụng bạn muốn xây dựng là ứng dụng đa nền tảng.
- Không chỉ nhanh chóng trong việc xây dựng database, Google Firebase còn giúp ta đơn giản hóa quá trình đăng kí và đăng nhập vào ứng dụng bằng các sử dụng hệ thống xác thực do chính Firebase cung cấp.

1.8.3.2. Bảo mật

- Firebase hoạt động dựa trên nền tảng cloud và thực hiện kết nối thông qua giao thức bảo mật SSL, chính vì vậy bạn sẽ bớt lo lắng rất nhiều về việc bảo mật của dữ liệu cũng như đường truyền giữa client và server.

- Không chỉ có vậy, việc cho phép phân quyền người dùng database bằng cú pháp javascipt cũng nâng cao hơn nhiều độ bảo mật cho ứng dụng của bạn, bởi chỉ những user mà bạn cho phép mới có thể có quyền chỉnh sửa cơ sở dữ liệu.

1.8.3.3. Tính linh hoạt và khả năng mở rộng

- Sử dụng Firebase sẽ giúp bạn dễ dàng hơn rất nhiều mỗi khi cần nâng cấp hay mở rộng dịch vụ. Ngoài ra firebase còn cho phép bạn tự xây dựng server của riêng mình để bạn có thể thuận tiện hơn trong quá trình quản lý.
- Việc Firebase sử dụng NoSQL, giúp cho database của bạn sẽ không bị bó buộc trong các bảng và các trường mà bạn có thể tùy ý xây dựng database theo cấu trúc của riêng bạn.

1.8.3.4. Sự ổn định

- Firebase hoạt động dựa trên nền tảng cloud đến từ Google vì vậy hầu như bạn không bao giờ phải lo lắng về việc sập server, tấn công mạng như DDOS, tốc độ kết nối lúc nhanh lúc chậm, … nữa, bởi đơn giản là Firebase hoạt động trên hệ thống server của Google. Hơn nữa nhờ hoạt động trên nền tảng Cloud nên việc nâng cấp, bảo trì server cũng diễn ra rất đơn giản mà không cần phải dừng server để nâng cấp như truyền thống.

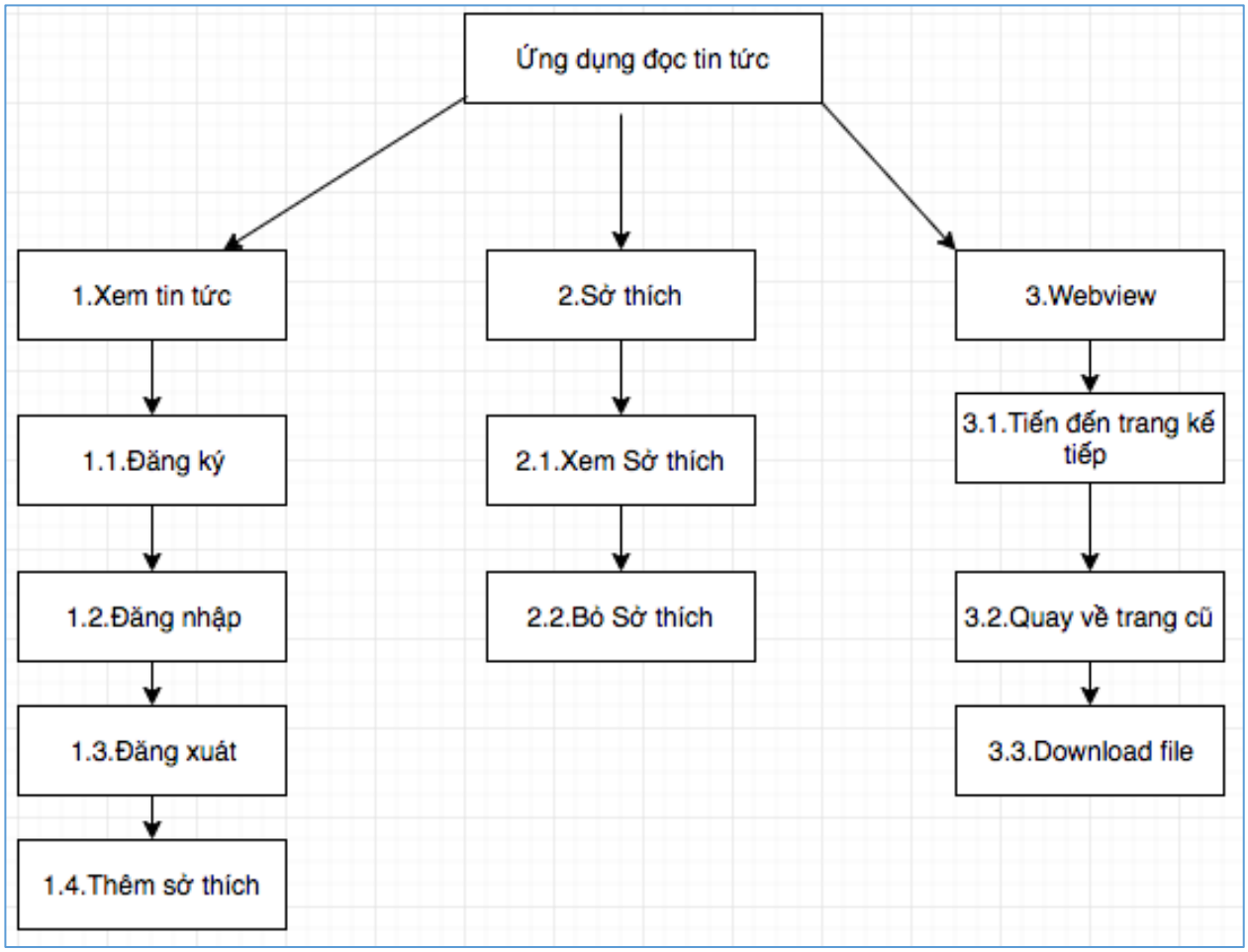
1.8.3.5. Giá thành

- Google Firebase có rất nhiều gói dịch vụ với các mức dung lượng lưu trữ cũng như băng thông khác nhau với mức giá dao động từ Free đến $1500 đủ để đáp ứng được nhu cầu của tất cả các đối tượng.
- Chính vì vậy bạn có thể lựa chọn gói dịch vụ phù hợp nhất với nhu cầu của mình. Điều này giúp bạn tới ưu hóa được vốn đầu tư và vận hành của mình tùy theo số lượng người sử dụng. Ngoài ra bạn còn không mất chi phí để bảo trì, nâng cấp, khắc phục các sự cố bởi vì những điều này đã có Firebase lo.
- Trên đây chính là 5 lợi ích dễ thấy nhất khi bạn sử dụng Firebase thay cho việc sử dụng các database theo cách truyền thống, ngoài ra có lẽ còn nhiều lợi ích khác mà chỉ khi xây dựng ứng dụng với Firebase bạn mới có thể tự trải nghiệm được.

CHƯƠNG 2. Thiết Kế Và Cài Đặt Ứng Dụng

2.1. Thiết Kế

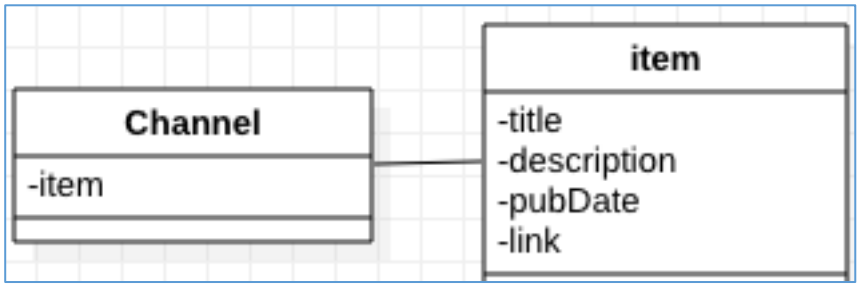
2.1.1. Mô hình chức năng



Hình 1.15 Mô hình chức năng

2.1.2. Mô hình dữ liệu

2.1.2.1 Cấu trúc tin tức Rss



Hình 1 16 Cấu trúc tin tức rss

2.1.2.2. Mô tả cấu trúc Rss

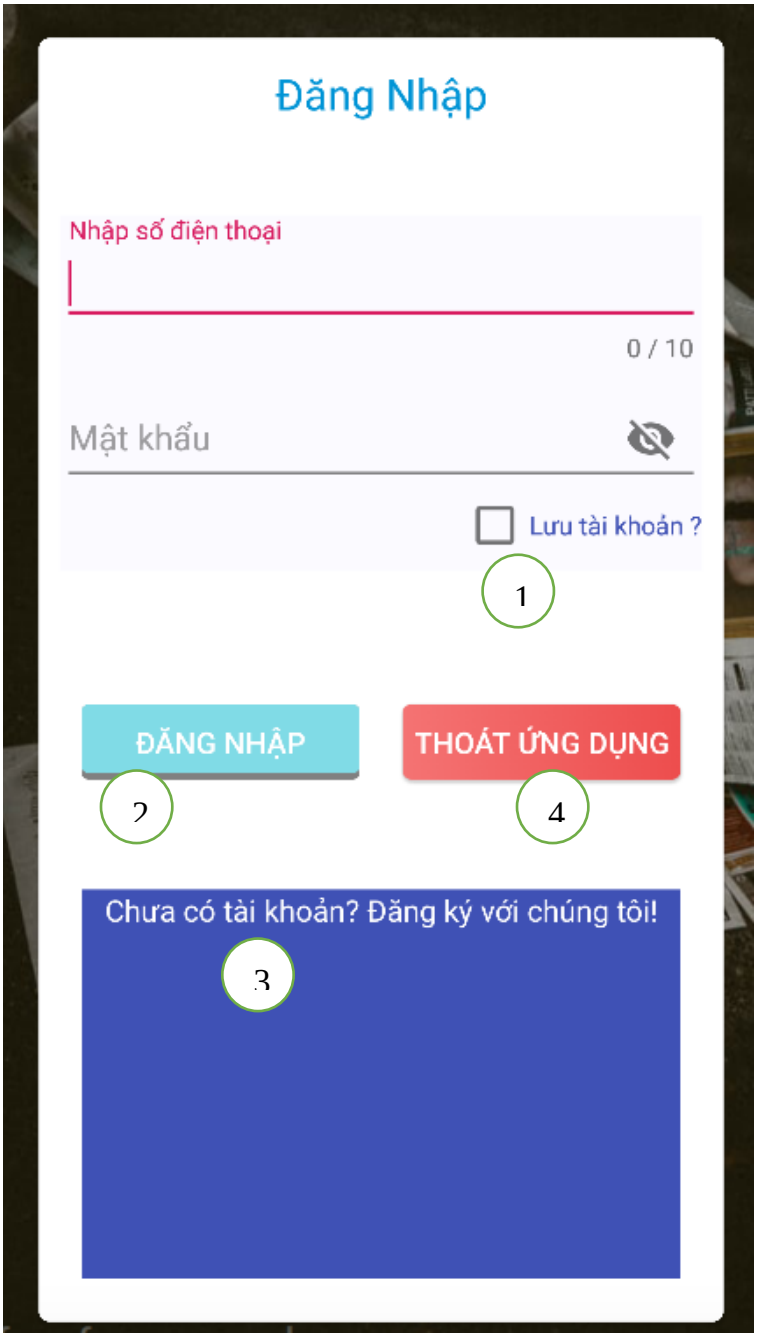
Tên	Mô tả

<channel>	Mỗi channel chứa một hoặc nhiều item
<item>	Thành phần chứa các mục của rss
<title>	Tiêu đề của rss
<description>	Mô tả nội dung của rss
<pubDate>	Ngày đăng của rss
<link>	Đường dẫn đến trang rss

Bảng 1.1 Bảng mô tả cấu trúc rss

2.2. Cài đặt ứng dụng

- Khi app khởi động thì màn hình đang nhập sẽ hiển thị.



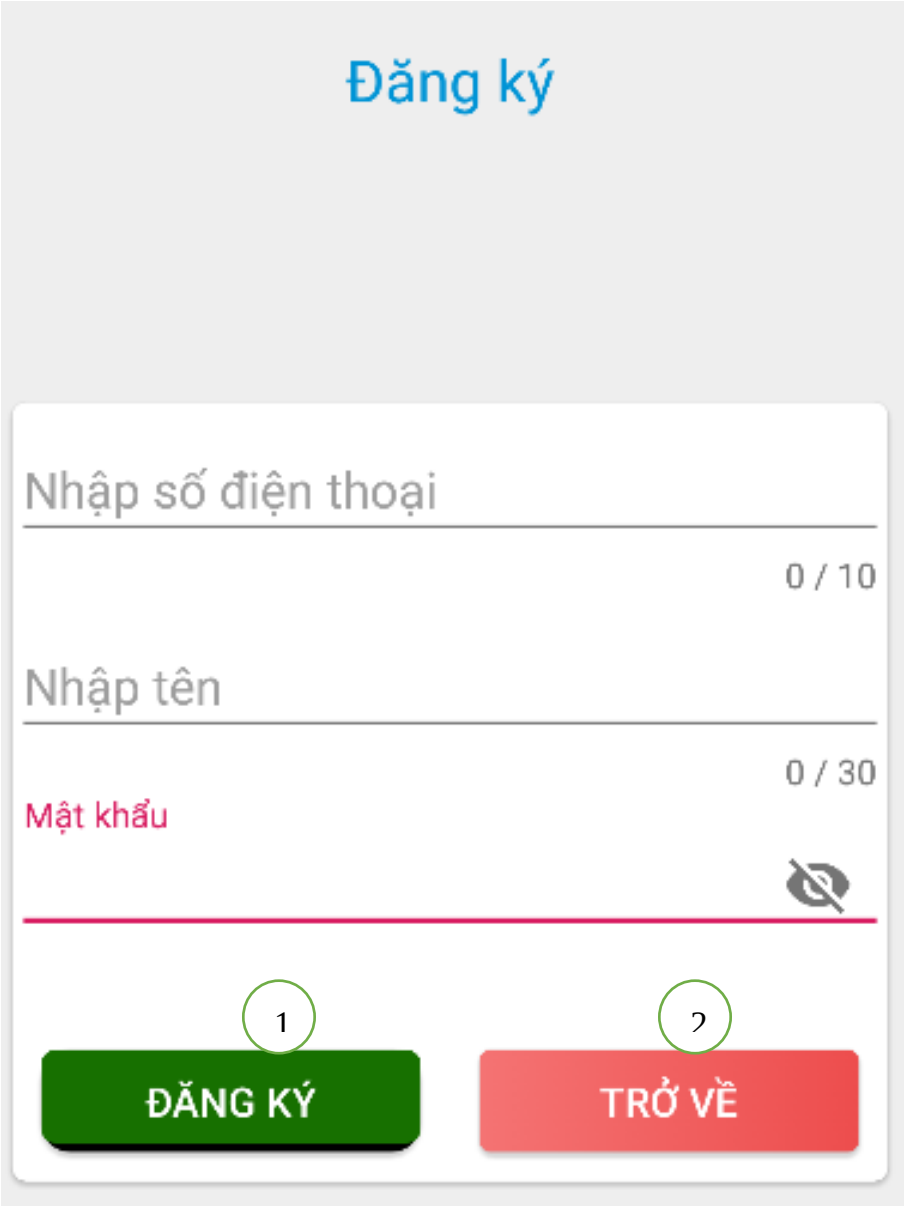
Hình 1.17 Màn hình đăng nhập

Mô tả hình 1.17

STT	Tên chức năng	Ý nghĩa
1	Lưu tài khoản	Chức năng cho phép lưu lại tài khoản và tự đăng nhập cho lần tiếp theo
2	Đăng nhập	_Tài khoản:+Số diện thoại: 0909130464 +Password: 123456 _Khi điền đủ 10 số và password đầy đủ thì button đăng nhập sẽ enable cho pháp vào màn hình chính, ngược lại sẽ hiện lỗi
3	Link đăng ký	Chức năng cho phép tới màn hình đăng ký
4	Thoát	Thoát ứng dụng

Bảng 1.2 Mô tả chức năng màn hình đăng nhập

- Màn hình đăng ký



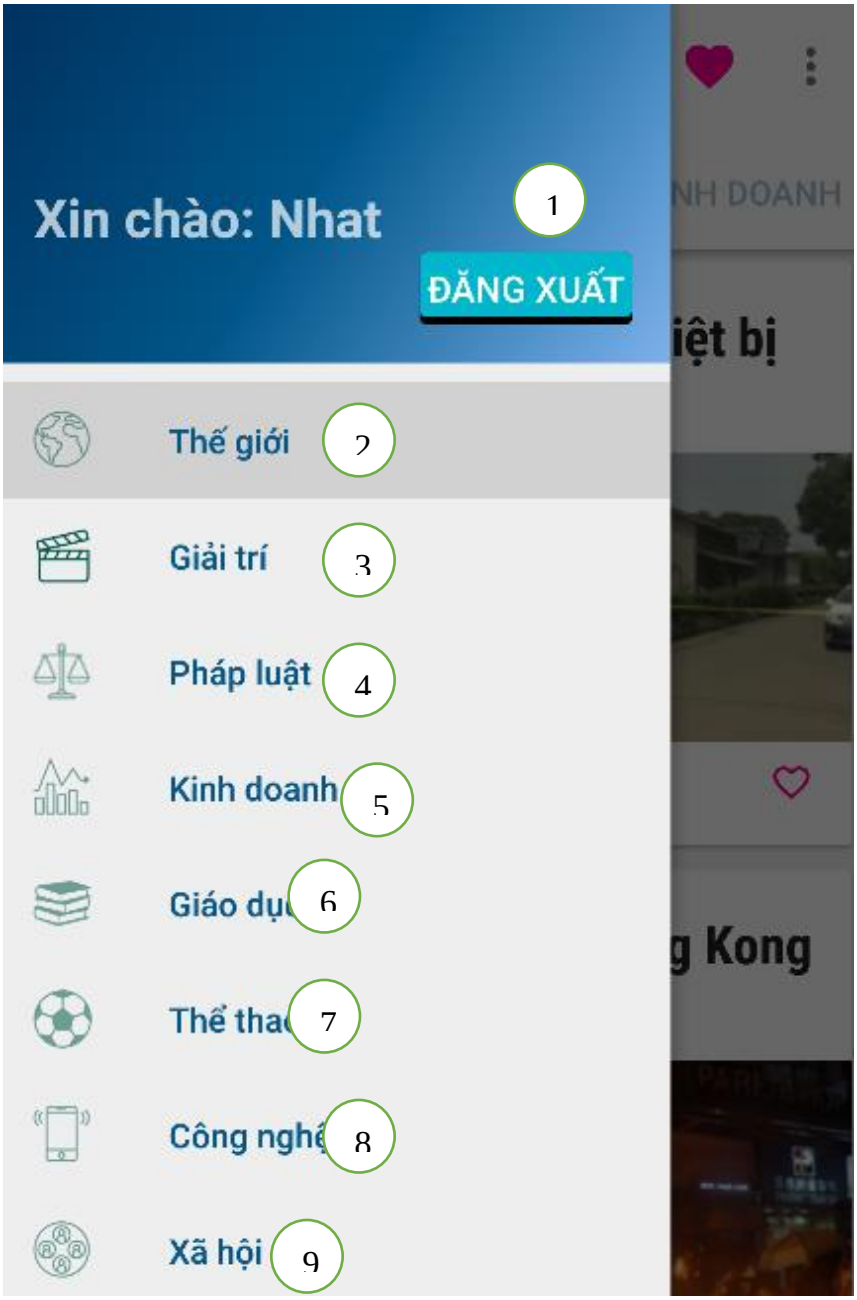
Hình 1.18 Màn hình đăng ký

Mô tả hình 1.18

STT	Tên chức năng	Ý nghĩa
1	Đăng ký	Khi nhập thì số điện thoại tối đa 10 số và nhập tên và mật khẩu không được bỏ trống thì chức năng sẽ đăng ký thành công, tài khoản có thể đăng nhập vào ứng dụng, ngược lại sẽ hiện lỗi
2	Trở về	Trở lại màn hình đăng nhập

Bảng 1.3 Mô tả chức năng màn hình đăng kí

- Màn hình Drawer Layout



Hình 1.19 Màn hình drawer layout

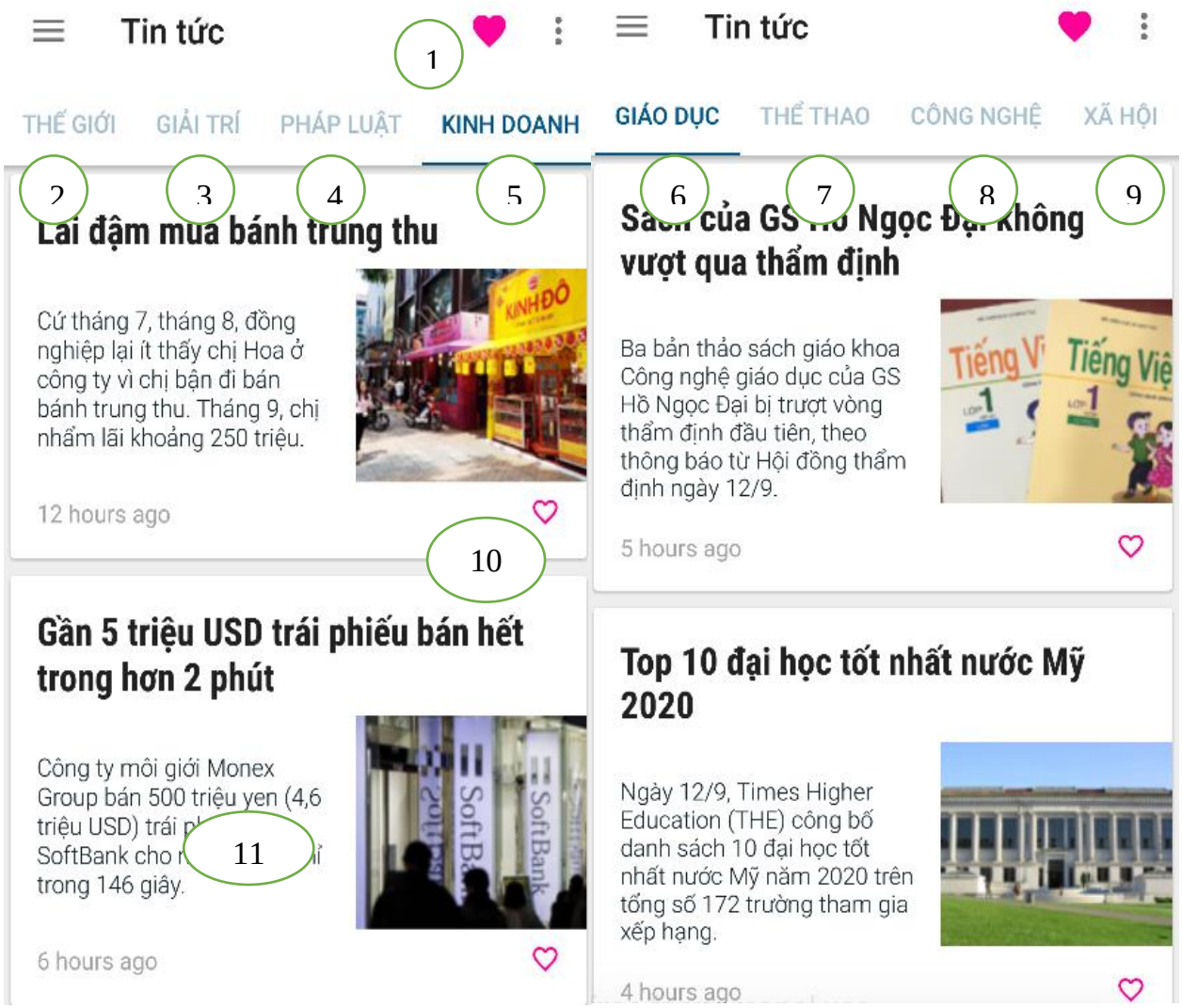
Mô tả hình 1.17

STT	Tên chức năng	Ý nghĩa
1	Đăng xuất	Đăng xuất tài khoản ra khỏi ứng dụng
2	Thế giới	Đi đến Viewpaper và hiện thông tin trang Thế giới
3	Giải trí	Đi đến Viewpaper và hiện thông tin trang Giải trí
4	Pháp luật	Đi đến Viewpaper và hiện thông tin trang Pháp luật
5	Kinh doanh	Đi đến Viewpaper và hiện thông tin trang Kinh doanh
6	Giáo dục	Đi đến Viewpaper và hiện thông tin trang Giáo dục
7	Thể thao	Đi đến Viewpaper và hiện thông tin trang Thể thao

8	Công nghệ	Đi đến Viewpaper và hiện thông tin trang Công nghệ
9	Xã hội	Đi đến Viewpaper và hiện thông tin trang Xã hội

Bảng 1.4 Mô tả chức năng màn hình Drawer Layout

- Màn hình chính của trang báo



Hình 1.20 Màn hình chính trang báo

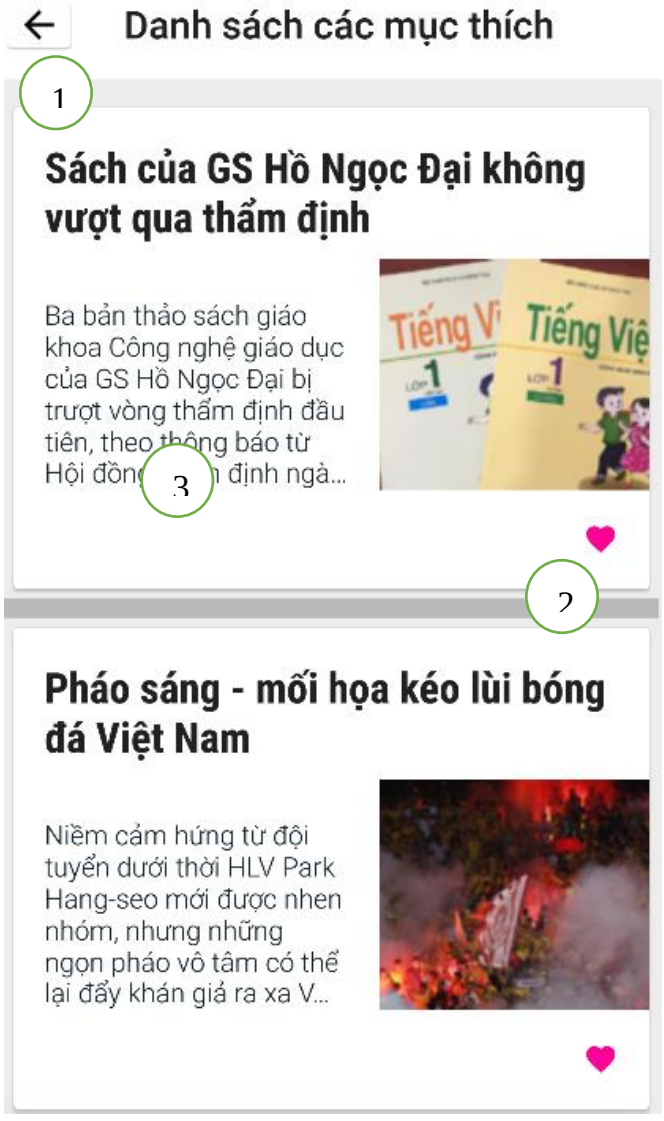
Mô tả hình 1.20

STT	Tên chức năng	Ý nghĩa
1	Mục sở thích	Vào màn hình sở thích
2	Thế giới	Đi đến Viewpaper và hiện thông tin trang Thế giới
3	Giải trí	Đi đến Viewpaper và hiện thông tin trang Giải trí
4	Pháp luật	Đi đến Viewpaper và hiện thông tin trang Pháp luật
5	Kinh doanh	Đi đến Viewpaper và hiện thông tin trang Kinh doanh
6	Giáo dục	Đi đến Viewpaper và hiện thông tin trang Giáo dục

7	Thể thao	Đi đến Viewpaper và hiện thông tin trang Thể thao
8	Công nghệ	Đi đến Viewpaper và hiện thông tin trang Công nghệ
9	Xã hội	Đi đến Viewpaper và hiện thông tin trang Xã hội
10	Sở thích	Cho phép thêm và bỏ thích vị trí báo hiện tại vào danh mục thích
11	Link trang	Đi đến đường link của trang đó trên webview

Bảng 1.5 Mô tả chức năng màn hình chính

- Màn hình sở thích



Hình 1.21 Màn hình sở thích

Mô tả hình 1.21

STT	Tên chức năng	Ý nghĩa
1	Trở về	Trở về màn hình chính
2	Sở thích	Bỏ thích

3	Link trang	Đi đến đường link của trang đó trên webview

Bảng 1.6 Mô tả chức năng màn hình sở thích

- Giao diện webview

Hình 1.22 Màn hình webview

Mô tả hình 1.20

STT	Tên chức năng	Ý nghĩa
1	Trở về	Trở về màn hình chính
2	Trang trước	Về trang trước
3	Trang sau	Về trang sau
4	Chia sẻ	Có thể chia sẻ liên kết cho mọi người
5	Download	Có thể sử dụng webview để tải file

Bảng 1.7 Mô tả chức năng màn hình WebView

Chương 3. Kết Luận

❖ **Kết quả đạt được:**

- Cơ bản nắm được các bước xây dựng một áp tin tức trên di động và hỗ trợ Android 5.0 trở lên.
- Có thể lưu lại các bài báo đã thích một cách dễ dàng.
- Đồ án giúp em hiểu về các kỹ thuật Parser XML và sử dụng tính kế thừa của các Fragment.

❖ **Hạn chế:**

- App nhỏ, chỉ mô phỏng một phần nhỏ
- Chưa xử lý được các giao diện phức tạp.
- Hạn chế trong việc phân trang cũng như chưa tối ưu được ứng dụng trong khi hoạt động.

❖ **Hướng phát triển:**

- Xây dựng app tin tức có thể xem nhiều trang báo khác nhau.
- Sẽ tiếp tục nghiên cứu để phát triển phần mềm hoàn thiện hơn nữa sửa chữa các sai xót.
- Tiếp tục nghiên cứu để làm cho phần mềm thêm nhiều tính năng hơn.
- Nghiên cứu và tìm hiểu sâu về các kiến thức liên quan tới hệ điều hành Android để có thể thêm các chức năng xem giá vàng, chứng khoáng…

Tài Liệu Tham Khảo

[1] Bài giảng của thầy Giang Hào Côn trường đại học Nguyễn Tất Thành.

[2] John Wiley & Sons, Java Programing for Android Developers, published in Canada, năm 2017

[3] https://www.journaldev.com/9412/android-shared-preferences-example-tutorial, ngày 5/09/2019