**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÀI TẬP LỚN ANDROID**

**ĐỀ TÀI: Ứng Dụng Nghe Nhạc**

**Giảng viên:**

**Thành viên:**

**Hà Nội, 2020**

**Mục Lục**

**Lời nói đầu**

1. **Tổng quát**
   1. Mô tả yêu cầu bài toán
   2. Biểu đồ use case
2. **Môi trường và các kỹ thuật hỗ trợ**
   1. Ngôn ngữ và môi trường lập trình
   2. Lưu trữ dữ liệu
   3. MediaPlayer trong Android
      1. Trạng thái phát nhạc
      2. Permission
   4. Các thư viện sử dụng thêm
3. **Kỹ thuật lập trình**
   1. Khai báo Mainfest
   2. Database
   3. File PHP
   4. Sử dụng retrofit để lấy dữ liệu từ API
      1. Sử dụng Annotations
      2. Xây dựng class APIService và APIRetrofitClient nhận url Server
      3. Thực hiện Request đến API trả về dữ liệu
   5. Xây dựng các Adapter
   6. Xây dựng các Fragment
   7. Xây dựng các Activity
      1. Main Activity
      2. Play Mp3 Ativity
4. **Giao diện chương trình**
   1. Đăng nhập
   2. Chọn bài và phát nhạc
   3. Tìm kiếm
   4. Thích bài hát
   5. Đọc nhạc từ bộ nhớ máy và 1 số tính năng khác
   6. Link video demo
5. **Phân công việc**

**LỜI NÓI ĐẦU**

Trước xu thế hội nhập ngày càng mở rộng, ngành dịch vụ Viễn thông nói chung và dịch vụ điện thoại di động nói chung đang bước vào giai đoạn cạnh tranh rất lớn. Nổi lên trong giai đoạn hiện nay là công nghệ hệ điều hành di động và nổi bật như một ngôi sao mới là hệ điều hành Android. 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, được nhà phát triển bởi hãng công nghệ rất nổi tiếng hiện nay là Google hậu thuẫn. Android 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, được nhiều người ưa chuộng nhất.

Vì vậy em đã quyết định chọn đề tài “Ứng dụng nghe nhạc trực tuyến trên hệ điều hành Android” cho bài tập lớn môn này. Trong quá trình triển khai đề tài em tập trung tìm hiểu về hệ điều hành Android và nền tảng phát triển phần mềm trên Android từ đó xây chương trình ứng dụng.

**Chương 1. Tổng quát**

* 1. **Mô tả yêu cầu bài toán**

1. Về cơ bản tổng quát hóa bài toán thì app cần có:

+ Giao diện đăng nhập và đăng ký tài khoản.

+ Giao diện hiển thị các danh sách playlist nhạc, album, quảng cáo, bài hát được yêu thích.

+ Giao diện danh sách nhạc lấy từ điện thoại

+ Giao diện hiển thị nghe nhạc

1. Về chức năng chính của app là:

+ Nghe nhạc từ diện thoại

+ Nghe nhạc nhạc trực tuyến mọi lúc.

+ Tìm kiếm bài hát.

+ Download bài hát

+ Chia sẻ lên Facebook

+ Xem thời tiết Hà Nội hiện tại

+ Tổng quan số liệu về Covid ở VN

* 1. **Biểu đồ use case**

1. Sơ đồ User-case tổng quát**A close up of a logo

   Description automatically generated**

*Hình 1.2: User-Case tổng quát*

1. Mô tả: người truy cập app có thể search bài hát và nghe bài hát.
2. Sơ đồ hoạt động của app:

A picture containing white

Description automatically generated

**Chương 2: Môi trường và các kỹ thuật hỗ trợ**

* 1. **Ngôn ngữ và môi trường lập trình**

1. Ngôn ngữ: Java (java version "12.0.2")
2. IDE: Android Studio
3. Sử dụng PHP.
   1. **Lưu trữ dữ liệu**
4. Sử dụng hosting (000webhost.com).
5. Xây dựng cơ sở dữ liệu trên phpAdmin.
6. Sử dụng webhost lưu chữ dữ liệu đi kèm.
7. Định dạng dữ liệu JSON.
   1. **MediaPlayer trong Android**
8. Media Player trong Android là một framework hỗ trợ để play những định dạng như video, nhạc, ảnh trong ứng dụng của mình.
   * 1. **Trạng thái phát nhạc**

A close up of a map

Description automatically generated

*Hình 2.3: Trạng thái của MediaPlayer*

Chúng ta có thể thấy được các trạng thái sau là các trạng thái mà media có thể ở đó:

1. **Idle** : Trạng thái ban đầu của Media Player. Khi muốn trở về trạng thái ban đầu của MediaPlayer thì có thể sử dụng phương thức reset().
2. **Initialized**: trạng thái đã khởi tạo được dữ liệu bên trong Media Player bằng phương thức setDataSource(Uri uri). Việc truyền vào Uri này có thể là ở local hoặc là ở server đều có thể play được.
3. **End**: Trạng thái kết thúc MediaPlayer khi hàm release() được gọi. Đến đây MediaPlayer hoàn toàn bị hủy bỏ khi không sử dụng nữa
4. **Error**: Trạng thái lỗi nếu việc play có vấn đề gì đó với Uri hoặc việc tương tác với nhạc trên internet gặp vấn đề. Lắng nghe listener OnErrorListener để xử lý khi không thể play nhạc.
5. **Prepared**: Trạng thái đã được chuẩn bị để sẵn sàng để có thể play hoặc là trạng thái khi stop hoặc là seekTo() đến khoảng nào đó trong toàn bộ độ dài của media. Việc đến được trạng thái này chúng ta phải trải qua hàm prepareAsync() (không đồng bộ) hoặc là prepare() (đồng bộ). Nếu gọi phương thức đồng bộ prepareAsync() thì sẽ có 1 listener được trả về nếu mà quá trình chuẩn bị thành công thì sẵn sàng để play. Chỗ này có thể gửi sự kiện chuẩn bị xong ra ngoài giao diện để cập nhật được giao diện phù hợp.
6. **Started**: Trạng thái khi mà media đang được chạy, lúc này có thể sử dụng pause() để tạm dừng nhạc hoặc stop() để dừng hẳn nhạc của media. Gọi isPlaying() để biết được media có đang được chạy không.
7. **Pause**: Khi media đang chạy, gọi hàm pause() để dừng media khi đang phát nhạc. Lúc này trạng thái media đang tạm dừng. Có thể dừng hẳn bằng hàm stop() hoặc start() để chạy media. Thường dùng cho nút play/pause gọi hàm này để thực hiện việc quản lý media.
8. **Stop**: Khi đang ở trạng thái dừng hẳn, có thể chạy lại media bằng prepareAsync() hoặc là prepare() để có thể trở về trạng thái Prepared và chuẩn bị để play lại từ đầu.
9. **Playback** **Complete**: Trạng thái mà chạy xong media sẽ nhảy vào đây, có thể lắng nghe sự kiện OnCompletionListener của nó để có thể thực hiện chuyển bài hoặc là vẫn play bài đó với các trường hợp của loop media.

~ Ngoài ra còn hàm **seekTo**() để có thể chuyển đến phần mà bạn chọn khi play media. Hàm này có thể được gọi ở nhiều trạng thái khác nhau nên phải chú ý tới từng trạng thái của Media khi gọi hàm này. Việc cho biết chạy đến đâu của tổng số thời gian được trả về thông qua hàm **getCurrentPosition**().

* + 1. **Permission**

1. Có một số trường hợp sử dung permission WAKE\_LOOK để giữ màn hình ở trạng thái on, ví dụ khi sử dụng với play video.

<uses-permission android:name="android.permission.WAKE\_LOCK" />

1. Nếu ứng dụng có sử dụng đến media trên mạng internet thì cần cấp quyền INTERNET để có thể tải được về và chạy.

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

1. Quyền đọc và ghi vào bộ nhớ điện thoại
2. <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" /> <usespermissionandroid:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />
   1. **Các thư viện sử dụng thêm**
3. Picasso: Hỗ trợ load ảnh từ server về app.
4. Gson: Thư viện java cho phép người dùng chuyển đổi sữ liệu từ JSON sang java và ngược lại.
5. Retrofit2: là một type-safe HTTP client cho Java và Android được phát triển bởi Square. Retrofit giúp dễ dàng kết nối đến một dịch vụ REST trên web bằng cách chyển đổi API thành Java Interface.
6. Facbook-Login: hỗ trợ từ đăng nhập từ tài khoản facebook
7. Các thư viện hỗ trợ hiển thị khác như: CardView, Recyclerview,…

**Chương 3: Kỹ thuật lập trình**

**3.1. Khai báo Mainfest**

- Khai báo Activity trong ứng dụng và các permission liên quan (INTERNET)

<?*xml version*="1.0" *encoding*="utf-8"?>  
<manifest *xmlns:android*="http://schemas.android.com/apk/res/android"  
 *xmlns:tools*="http://schemas.android.com/tools"  
 *package*="com.example.appzing">  
  
 <uses-permission *android:name*="android.permission.INTERNET" />  
 <uses-permission *android:name*="android.permission.READ\_EXTERNAL\_STORAGE" />  
 <uses-permission *android:name*="android.permission.WRITE\_EXTERNAL\_STORAGE" />  
  
 <application  
 *android:allowBackup*="true"  
 *android:icon*="@mipmap/ic\_launcher"  
 *android:label*="@string/app\_name"  
 *android:roundIcon*="@mipmap/ic\_launcher\_round"  
 *android:supportsRtl*="true"  
 *android:theme*="@style/AppTheme">  
 <activity *android:name*=".Activity.Weather"></activity>  
 <activity *android:name*=".Activity.Covid" />  
 <activity *android:name*=".Activity.PlayNhac3" />  
  
 <meta-data  
 *android:name*="com.facebook.sdk.ApplicationId"  
 *android:value*="@string/facebook\_app\_id" />  
 <meta-data  
 *android:name*="om.facebook.sdk.ApplicationId"  
 *android:value*="469435510525416" />  
  
 <provider  
 *android:name*="com.facebook.FacebookContentProvider"  
 *android:authorities*="com.facebook.FacebookContentProvider469435510525416"  
 *android:exported*="true" />  
  
 <activity  
 *android:name*="com.facebook.FacebookActivity"  
 *android:configChanges*="keyboard|keyboardHidden|screenLayout|screenSize|orientation"  
 *android:label*="@string/app\_name" />  
 <activity  
 *android:name*="com.facebook.CustomTabActivity"  
 *android:exported*="true">  
 <intent-filter>  
 <action *android:name*="android.intent.action.VIEW" />  
  
 <category *android:name*="android.intent.category.DEFAULT" />  
 <category *android:name*="android.intent.category.BROWSABLE" />  
  
 <data *android:scheme*="@string/fb\_login\_protocol\_scheme" />  
 </intent-filter>  
 </activity>  
 <activity *android:name*=".Activity.Mymp3" />  
 <activity *android:name*=".Activity.CreateLoginActivity" />  
 <activity *android:name*=".Activity.MainActivity" />  
 <activity *android:name*=".Activity.PlayNhacActivity" />  
 <activity *android:name*=".Activity.DanhsachtatcaAlbumActivity" />  
 <activity *android:name*=".Activity.DanhsachtheloaitheochudeActivity" />  
 <activity *android:name*=".Activity.DanhsachtatcachudeActivity" />  
 <activity *android:name*=".Activity.DanhsachcacplaylistActivity" />  
 <activity *android:name*=".Activity.DanhsachbaihatActivity" />  
 <activity *android:name*=".Activity.LoginActivity">  
 <intent-filter>  
 <action *android:name*="android.intent.action.MAIN" />  
  
 <category *android:name*="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

**3.2. Database**

- Các bảng trong cơ sở dữ liệu:

**A picture containing screenshot

Description automatically generated**

**-** Bảng bài hát:

A screenshot of a cell phone

Description automatically generated

- Dữ liệu bảng bài hát:

A picture containing light

Description automatically generated

**3.3. File PHP**

- Ở đây file albumhot.php trả về 4 các album ngẫu nhiên dưới dạng json A screenshot of a social media post

Description automatically generated

- Ở đây File search.php nhận từ khóa từ client, server trả về các ca khúc có chứa từ khóa

A screenshot of a social media post

Description automatically generated

**3.4. Sử dụng retrofit để lấy dữ liệu từ API**

a, Sử dụng Annotations

1. Annotations dể mô tả yêu cầu HTTP và hỗ trợ các tham số các URL và truyền tham số truy vấn
2. Mỗi phương thức phải có Annotations HTTP cung cấp request method và URL. Ứng dụng này, sử dụng các annotations:
3. public interface Dataservice {  
    @GET("songbanner.php")  
    Call<List<Quangcao>> GetDataBanner();  
     
    @GET("playlistforcurrentday.php")  
    Call<List<Playlist>> GetPlaylistCurrentDay();  
     
    @GET("chudevatheloaitrongngay.php")  
    Call<Theloaitrongngay> GetCategorMusic();  
     
    @GET("albumhot.php")  
    Call<List<Album>> GetAlbumHot();  
     
    @GET("baihatduocthich.php")  
    Call<List<Baihat>> GetBaiHatHot();  
     
    @FormUrlEncoded  
    @POST("danhsachbaihat.php")  
    Call<List<Baihat>> GetDanhsachbaihattheoquangcao(@Field("idquangcao") String idquangcao);  
     
    @FormUrlEncoded  
    @POST("danhsachbaihat.php")  
    Call<List<Baihat>> GetDanhsachbaitheotheoplaylist(@Field("idplaylist") String idplaylist);  
     
    @GET("danhsachcacplaylist.php")  
    Call<List<Playlist>> GetDanhsachcacPlayList();  
     
    @FormUrlEncoded  
    @POST("danhsachbaihat.php")  
    Call<List<Baihat>> GetDanhsachbaihattheotheloai(@Field("idtheloai") String idtheloai);  
     
    @GET("tatcachude.php")  
    Call<List<ChuDe>> GetAllChude();  
     
    @FormUrlEncoded  
    @POST("theloaitheochude.php")  
    Call<List<TheLoai>> GetTheloaitheochude(@Field("idchude") String idchude);  
     
    @GET("tatcaalbum.php")  
    Call<List<Album>> GetAllAlbum();  
     
    @FormUrlEncoded  
    @POST("danhsachbaihat.php")  
    Call<List<Baihat>> GetDanhsachbaihattheoalbum(@Field("idalbum") String idalbum);  
     
    @FormUrlEncoded  
    @POST("updateluotthich.php")  
    Call<String> UpdateLuotThich(@Field("luotthich") String luotthich, @Field("idbaihat") String idbaihat);  
     
    @FormUrlEncoded  
    @POST("search.php")  
    Call<List<Baihat>> GetSearch(@Field("tukhoa") String tukhoa);  
     
    @GET("getuser.php")  
    Call<List<User>> GetAllUser();  
     
    @FormUrlEncoded  
    @POST("insertuser.php")  
    Call<String> InsertUser(@Field("name") String username, @Field("pass") String password);  
   }

b, Xây dựng class APIService và APIRetrofitClient nhận url Server

public class APIRetrofitClient {  
 private static Retrofit *retrofit* = null;  
  
 public static Retrofit getClient(String base\_url) {  
 OkHttpClient okHttpClient = new OkHttpClient.Builder().readTimeout(10000, TimeUnit.*SECONDS*)  
 .writeTimeout(10000, TimeUnit.*MILLISECONDS*)  
 .connectTimeout(10000, TimeUnit.*MILLISECONDS*)  
 .protocols(Arrays.*asList*(Protocol.*HTTP\_1\_1*))  
 .build();  
  
 Gson gson = new GsonBuilder().setLenient().create();  
 *retrofit* = new Retrofit.Builder()  
 .baseUrl(base\_url)  
 .client(okHttpClient)  
 .addConverterFactory(GsonConverterFactory.*create*(gson))  
 .build();  
 return *retrofit*;  
 }  
}

public class APIService {  
 private static String *base\_url* = "https://canonized-emitter.000webhostapp.com/Server/";  
  
 public static Dataservice getService() {  
 return APIRetrofitClient.*getClient*(*base\_url*).create(Dataservice.class);  
 }  
}

c, Thực hiện Request đến API trả về dữ liệu

1. Ở đây là xử lý API Playlist

private void GetData() {  
 Dataservice dataservice = APIService.*getService*();  
 Call<List<Playlist>> callback = dataservice.GetPlaylistCurrentDay();  
 callback.enqueue(new Callback<List<Playlist>>() {  
 @Override  
 public void onResponse(Call<List<Playlist>> call, Response<List<Playlist>> response) {  
 mangplaylist = (ArrayList<Playlist>) response.body();  
 playlistAdapter = new PlaylistAdapter(getActivity(), android.R.layout.*simple\_list\_item\_1*, mangplaylist);  
 lvPlaylist.setAdapter(playlistAdapter);  
 setListViewHeightBasedOnChildren(lvPlaylist);  
 lvPlaylist.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
 Intent intent = new Intent(getActivity(), DanhsachbaihatActivity.class);  
 intent.putExtra("itemplaylist", mangplaylist.get(position));  
 startActivity(intent);  
 }  
 });  
 }  
  
 @Override  
 public void onFailure(Call<List<Playlist>> call, Throwable t) {  
  
 }  
 });  
}

**3.5. Xây dựng các Adapter**

- Class *ViewHolder* đỗ dữ liệu. Ở đây là 1 album trong list album:

public class ViewHolder extends RecyclerView.ViewHolder {  
  
 ImageView imgallalbum;  
 TextView txttenallalbum;  
  
 public ViewHolder(@NonNull View itemView) {  
 super(itemView);  
 imgallalbum = itemView.findViewById(R.id.*imageviewallalbum*);  
 txttenallalbum = itemView.findViewById(R.id.*textviewtenallalbum*);  
 itemView.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(context, DanhsachbaihatActivity.class);  
 intent.putExtra("album", albumArrayList.get(getPosition()));  
 context.startActivity(intent);  
 }  
 });  
 }  
}

1. Tiếp đến xây dựng class *AllAlbumAdapter* extends *RecyclerView.Adapter* đổ dữ liệu từ list nhận được:

public class AllAlbumAdapter extends RecyclerView.Adapter<AllAlbumAdapter.ViewHolder> {  
 Context context;  
 ArrayList<Album> albumArrayList;  
  
 public AllAlbumAdapter(Context context, ArrayList<Album> albumArrayList) {  
 this.context = context;  
 this.albumArrayList = albumArrayList;  
 }  
  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 LayoutInflater inflater = LayoutInflater.*from*((context));  
 View view = inflater.inflate(R.layout.*dong\_all\_album*, parent, false);  
  
 return new ViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull ViewHolder holder, int position) {  
 Album album = albumArrayList.get(position);  
 Picasso.*with*(context).load(album.getHinhanhAlbum()).into(holder.imgallalbum);  
 holder.txttenallalbum.setText(album.getTenAlbum());  
 }  
  
 @Override  
 public int getItemCount() {  
 return albumArrayList.size();  
 }

**3.6. Xây dựng các Fragment**

- Xây dựng các Fragment: Playlist, AlbumHot, Banner...

Ở đây Fragment\_Album\_Hot extends Fragment và hà GetData() đổ lại dữ liệu từ API

public class Fragment\_Album\_Hot extends Fragment {  
 View view;  
 RecyclerView recyclerViewalbum;  
 TextView txtxemthemalbum;  
 AlbumAdapter albumAdapter;  
  
 @Nullable  
 @Override  
 public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  
 view = inflater.inflate(R.layout.*fragment\_album\_hot*, container, false);  
 recyclerViewalbum = view.findViewById(R.id.*recyclerviewAlbum*);  
 txtxemthemalbum = view.findViewById(R.id.*textviewxemthemalbum*);  
 txtxemthemalbum.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(getActivity(), DanhsachtatcaAlbumActivity.class);  
 startActivity(intent);  
 }  
 });  
 GetData();  
 return view;  
 }  
  
 private void GetData() {  
 Dataservice dataservice = APIService.*getService*();  
 Call<List<Album>> callback = dataservice.GetAlbumHot();  
 callback.enqueue(new Callback<List<Album>>() {  
 @Override  
 public void onResponse(Call<List<Album>> call, Response<List<Album>> response) {  
 ArrayList<Album> albumArrayList = (ArrayList<Album>) response.body();  
 albumAdapter = new AlbumAdapter(getActivity(), albumArrayList);  
 LinearLayoutManager linearLayoutManager = new LinearLayoutManager(getActivity());  
 linearLayoutManager.setOrientation(linearLayoutManager.*HORIZONTAL*);  
 recyclerViewalbum.setLayoutManager(linearLayoutManager);  
 recyclerViewalbum.setAdapter(albumAdapter);  
 }  
  
 @Override  
 public void onFailure(Call<List<Album>> call, Throwable t) {  
  
 }  
 });  
 }  
}

- Thêm các Fragment vào Fragment\_Trang\_Chu qua file xml:

<?xml version="1.0" encoding="utf-8"?>  
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:fillViewport="true"  
 android:scrollbars="none">  
  
 <LinearLayout  
 android:id="@+id/linearLayout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <fragment  
 android:id="@+id/fragmentBanner"  
 android:name="com.example.appzing.Fragment.Fragment\_Banner"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </fragment>  
  
 <fragment  
 android:id="@+id/fragmentPlaylist"  
 android:name="com.example.appzing.Fragment.Fragment\_Playlist"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </fragment>  
  
 <fragment  
 android:id="@+id/fragmentTheloaitrongngay"  
 android:name="com.example.appzing.Fragment.Fragment\_Chude\_TheLoai\_Today"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </fragment>  
  
 <fragment  
 android:id="@+id/fragmentalbum"  
 android:name="com.example.appzing.Fragment.Fragment\_Album\_Hot"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </fragment>  
  
 <fragment  
 android:id="@+id/fragmentbaihat"  
 android:name="com.example.appzing.Fragment.Fragment\_Bai\_Hat\_Hot"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 </fragment>  
 </LinearLayout>  
  
</ScrollView>

**3.7. Xây dựng các Activity**

a, Main Activity

**-** MainActivity add các Fragment hiển thị trên giao diện chính:

private void init() {  
 MainViewPagerAdapter mainViewPagerAdapter = new MainViewPagerAdapter(getSupportFragmentManager());  
 mainViewPagerAdapter.addFragment(new Fragment\_Trang\_Chu(), "Trang chủ");  
 mainViewPagerAdapter.addFragment(new Fragment\_Tim\_Kiem(), "Tìm kiếm");  
  
 viewPager.setAdapter((mainViewPagerAdapter));  
 tabLayout.setupWithViewPager(viewPager);  
 tabLayout.getTabAt(0).setIcon(R.drawable.*icontrangchu*);  
 tabLayout.getTabAt(1).setIcon(R.drawable.*iconsearch*);  
}

class *MainViewPagerAdapter:*

public class MainViewPagerAdapter extends FragmentPagerAdapter {  
 private ArrayList<Fragment> arrayFragment = new ArrayList<>();  
 private ArrayList<String> arrayTitle = new ArrayList<>();  
  
 public MainViewPagerAdapter(FragmentManager fm) {  
 super(fm);  
 }  
  
 @Override  
 public Fragment getItem(int position) {  
 return arrayFragment.get(position);  
 }  
  
 @Override  
 public int getCount() {  
 return arrayFragment.size();  
 }  
  
 public void addFragment(Fragment fragment, String title) {  
 arrayFragment.add(fragment);  
 arrayTitle.add(title);  
 }  
  
 @Nullable  
 @Override  
 public CharSequence getPageTitle(int position) {  
 return arrayTitle.get(position);  
 }  
}

b, Play Mp3 Ativity

- Tạo class *PlayMp3* extends *AsyncTask* để xử lý việc phát nhạc

Phương thức ghi đè lấy link bài hát:

protected String doInBackground(String... strings) {  
 return strings[0];  
}

Phương thức xử lý phát nhạc:

protected void onPostExecute(String baihat) {  
 super.onPostExecute(baihat);  
 try {  
 mediaPlayer = new MediaPlayer();  
 mediaPlayer.setAudioStreamType(AudioManager.*STREAM\_MUSIC*);  
 mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {  
 @Override  
 public void onCompletion(MediaPlayer mp) {  
 mediaPlayer.stop();  
 mediaPlayer.reset();  
 }  
 });  
 mediaPlayer.setDataSource(baihat);  
 mediaPlayer.prepare();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 mediaPlayer.start();  
 TimeSong();  
 UpdateTime();  
}

**Chương 4: Giao diện chương trình**

1. **Đăng nhập**

a, Đăng nhập bằng tài khoản có sẵn

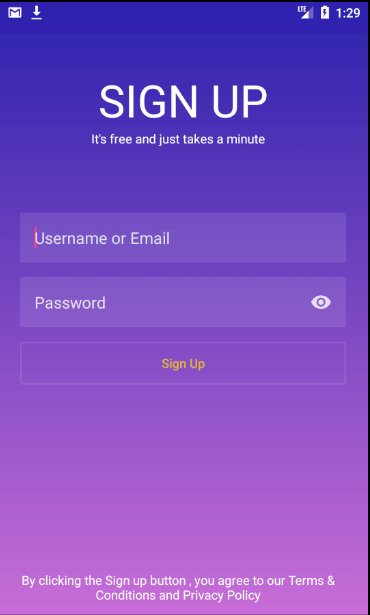
A screenshot of a cell phone

Description automatically generated

*Hình 4.1: Giao diện đăng nhập khi mở app.*

b, Tạo 1 tài khoản click “Don’t have an account?”

1. Hệ thống sẽ kiểm tra tài khoản đã tồn tại hay chưa. Đăng ký thành công trở lại Login.



*Hình 4.2: Giao diện đăng ký tài khoản.*

c, Đăng nhập bằng Facebook, hoặc Google

A screenshot of a cell phone

Description automatically generated

*Hình 4.3: Giao diện đăng nhập bằng tài khoản facebook.*

1. **Chọn bài và nghe nhạc**
2. Sau khi đăng nhập thành công

A screenshot of a cell phone

Description automatically generated

*Hình 4.2.1: Giao diện chính sau khi đăng nhập*

1. Chọn những phần khác nhau sẽ ra các list bài hát

Ở đây chọn vào phần playlist, chọn từng bài hát hoặc nghe cả list khi chọn biểu tượng icon zing mp3:

A screenshot of a cell phone

Description automatically generated

*Hình 4.4: Giao điện hiển thị các list nhạc*

1. Chọn “Xem Thêm Playlist” để hiển thị thêm các playlist khác

(thực hiện tương tự với thể loại và chủ đề, album)

A picture containing food

Description automatically generated

*Hình 4.5: Giao diện các playlist*

1. Chọn play list ta được, giao diện trình phát nhạc như sau:

A screen shot of a computer

Description automatically generated

1. Các chức năng được thực hiện khi chọn các nút: random bài hát, chuyển bài sau, dừng lại và tiếp tục, chuyển bài kế tiếp, lặp lại bài hát.
2. Chức năng share bài hát trên facebook, tải bài hát về máy dạng mp3.
3. **Tìm kiếm**
4. Nhập từ khóa trên thanh search, sau đó enter danh sách những bài hát liên quan sẽ hiển thị phía dưới. Chọn bài để phát nhạc.

A screenshot of a cell phone

Description automatically generated

1. **Thích bài hát**
2. Khi click icon love trái tim lượt thích tăng lên 1 gửi dữ liệu về server

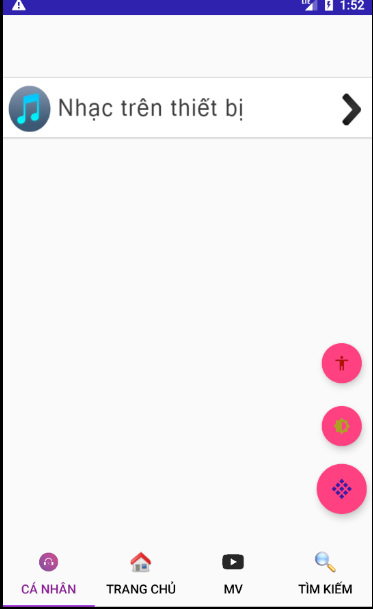
Top 5 bài hát được cập nhật có lượt thích cao nhất:

A screenshot of a cell phone

Description automatically generated

1. **Đọc nhạc từ bộ nhớ máy và 1 số tính năng khác**

**-**Giao diện menu

****

1. Một số tính nắng khác được gắn vào **Floating Action Button**

+ Xem thời thời tiết Hà Nội



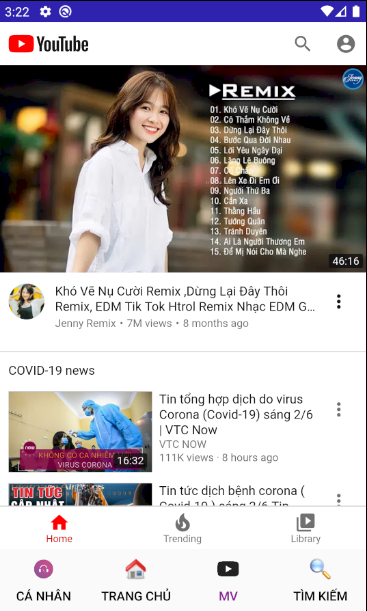
Dữ liệu được lấy từ OpenWeatherMap API trả về dạng JSON.

+Tra cứu thông tin về dịch Covid sử dụng API từ covid19api.com

A screenshot of a cell phone

Description automatically generated

+Ở tab MV sử dụng webview liên kết đến Youtube.com với mục đích xem MV



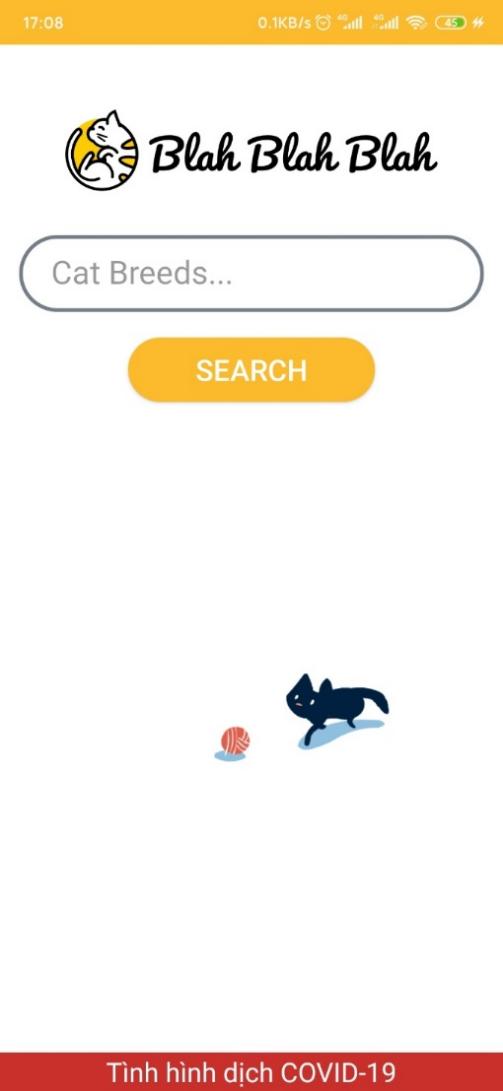
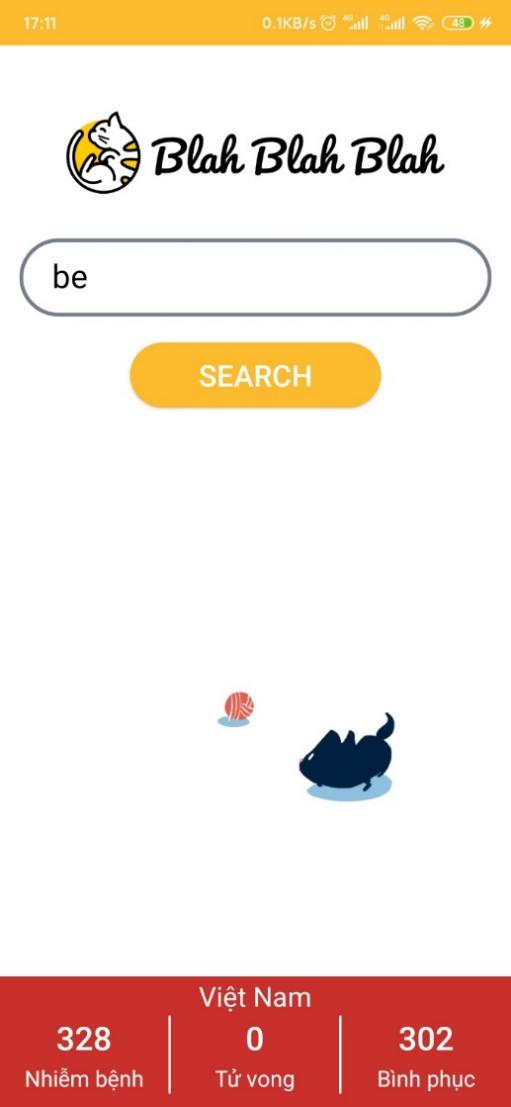
* ***Phần phụ:***

***Nhóm bọn em có làm thêm 2 app phụ, như là bản mobile của 2 chức năng trên website bài tập lớn ở môn web ạ.***

1. App Catpedia:

* Sử dụng API có sẵn để lấy dữ liệu về giống mèo theo từ khoá.
* Lấy số liệu về dịch covid-19 ở VN (chức năng phụ)
* Giao diện:

+ Trang chính:

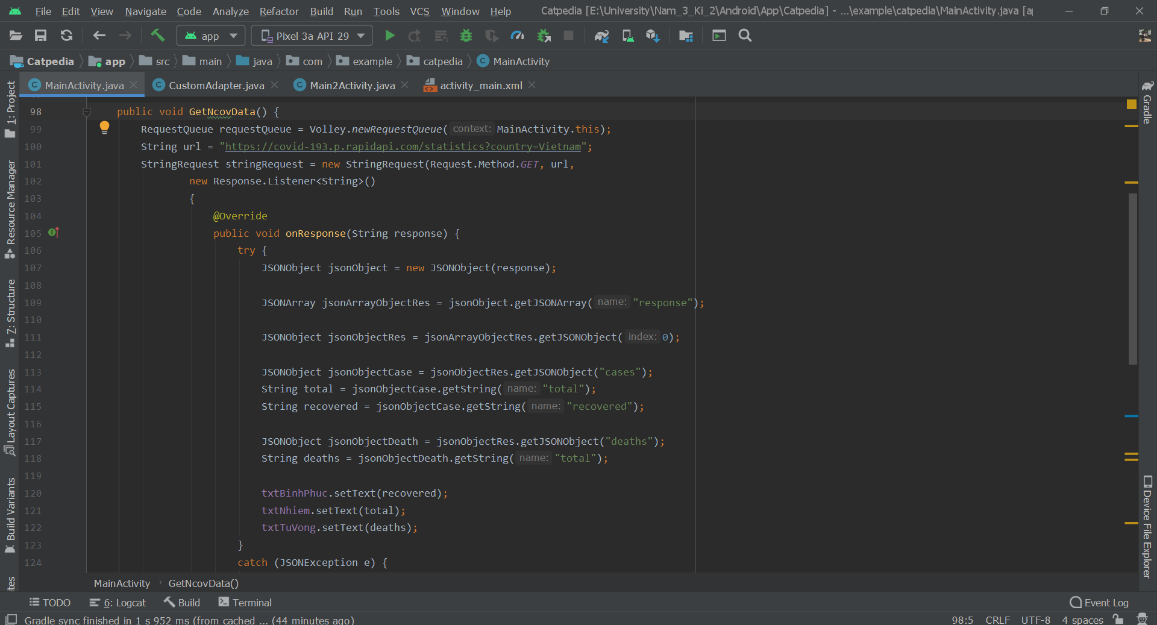
 

+ Trang kết quả

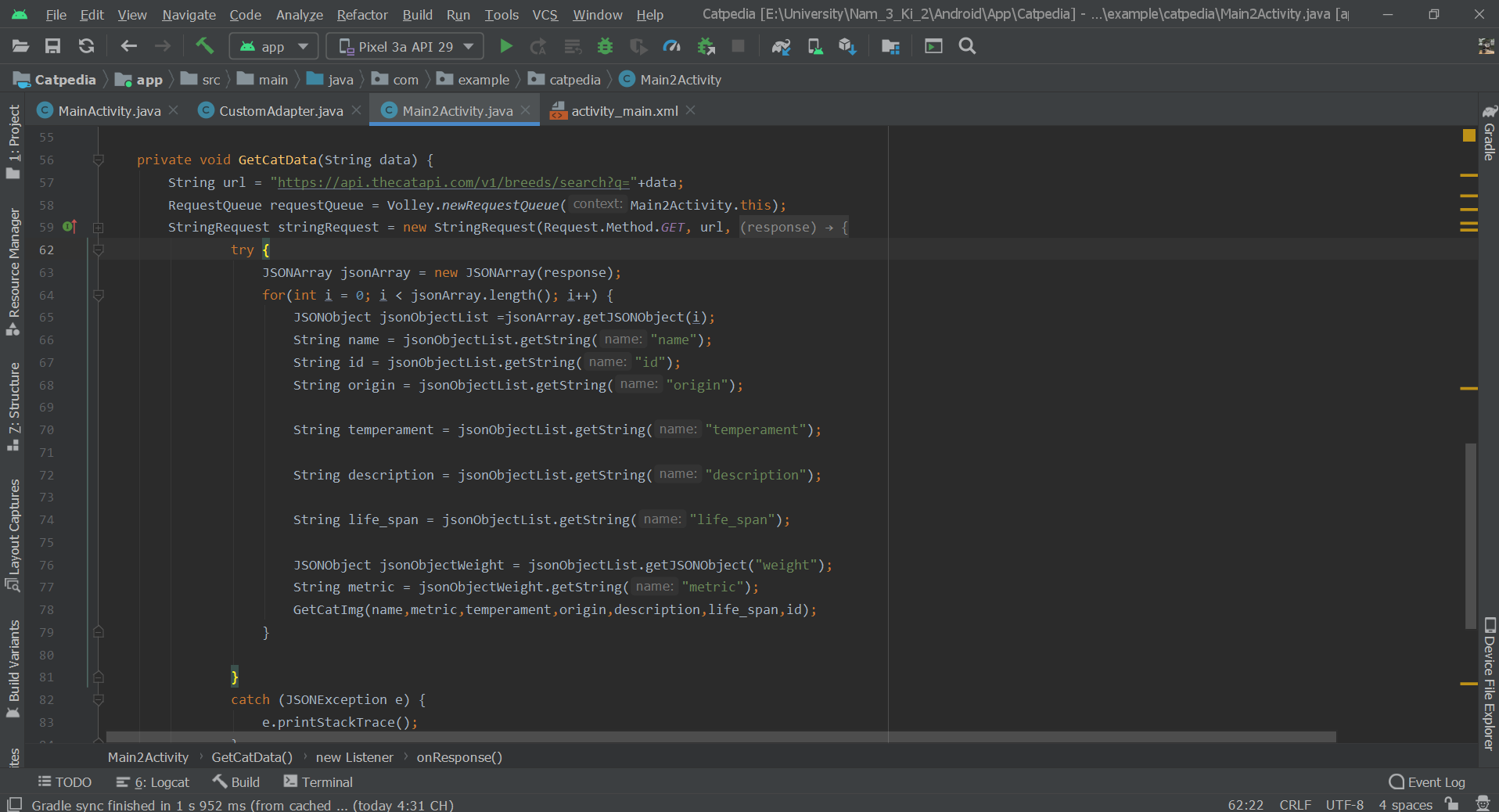


* Code:

+ Lấy dữ liệu api covid-19:



+ Lấy dữ liệu api về mèo:



1. App thời tiết:

* Sử dụng API có sẵn để lấy dữ liệu về thời tiết hiện tại và dự báo của 7 ngày tiếp theo.
* Hiển thị thanh thông báo để người dùng dễ dàng theo dõi.
* Giao diện:

+ Trang chính:



+ Trang thời tiết 7 ngày tới:

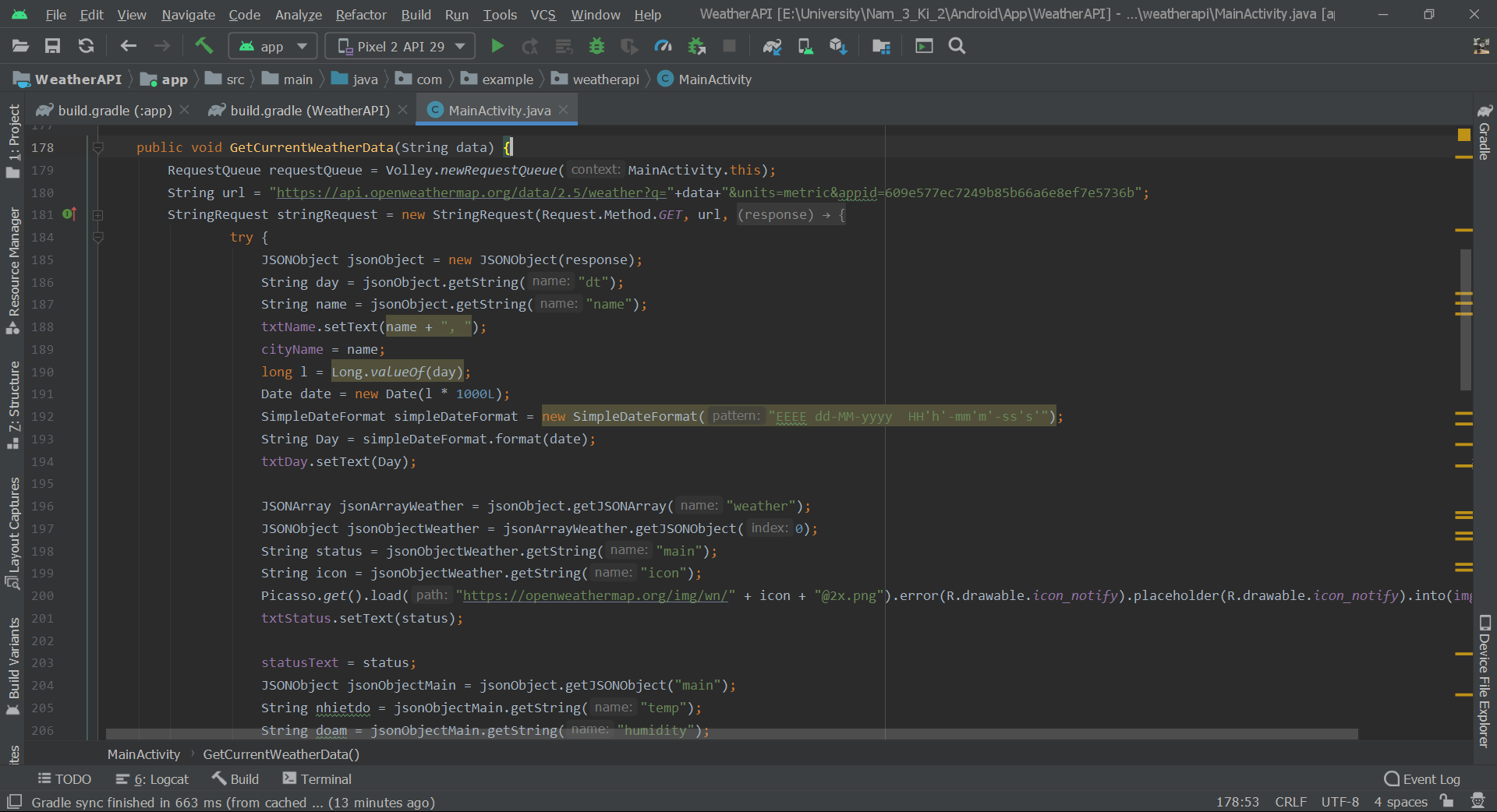


+ Thanh thông báo:

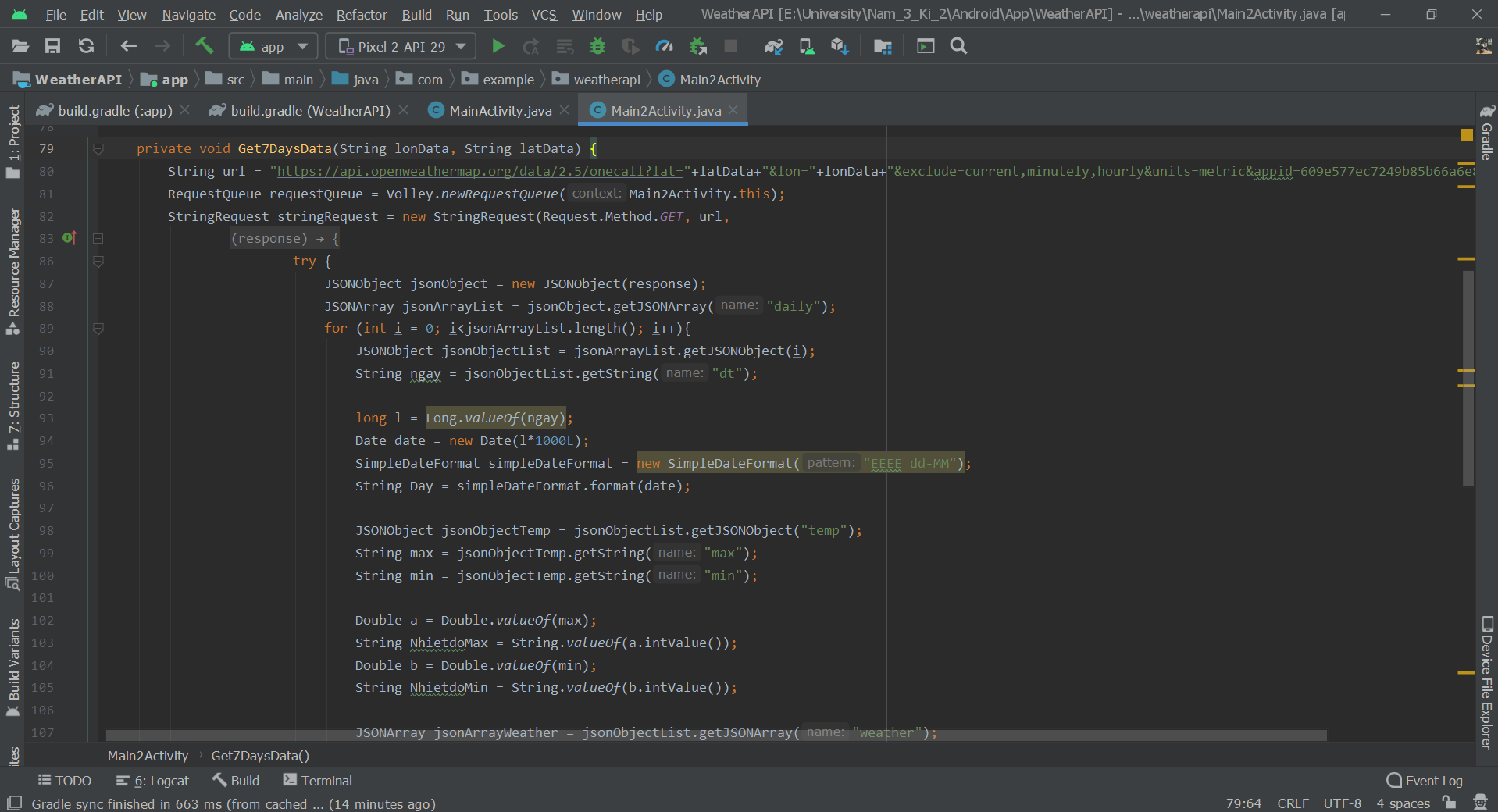


* Code:

+Lấy dữ liệu thời tiết hiện tại:



+Lấy dữ liệu thời tiết 7 ngày tới:



**Chương 5: Phân công việc**

* Phân công:

1. Ý tưởng, giao diện, chuẩn bị database, làm app catpedia và the weather: **Nguyễn Duy Hiển**
2. Code giao diện phát nhạc và trình phát nhạc online, thiết kế database:

**Lê Việt Hoàng**

1. Code trình phát nhạc offline, Sử dung API (Thống kê số liệu covid19 ,xem thời tiết Hà Nội, Đăng nhập GG,FB)

**Nguyễn Văn Long**

* Trong quá trình thực hiện các thành viên trong nhóm đều đã tham gia và hiểu toàn bộ chương trình.
* Kết thúc:

1. Do thời gian có hạn cùng với kiến thức của bản thân còn hạn chế nên chương trình vẫn còn rất nhiều hạn chế, nhóm em sẽ cố gắng phát triển để chương trình thân thiện và nhiều chức năng hơn.

Em trân thành cảm ơn!

*----------------------------------------------End-------------------------------------------------*