- Phải chia rõ model khi get api

- Sử dụng OkHttp, Gson để getJson

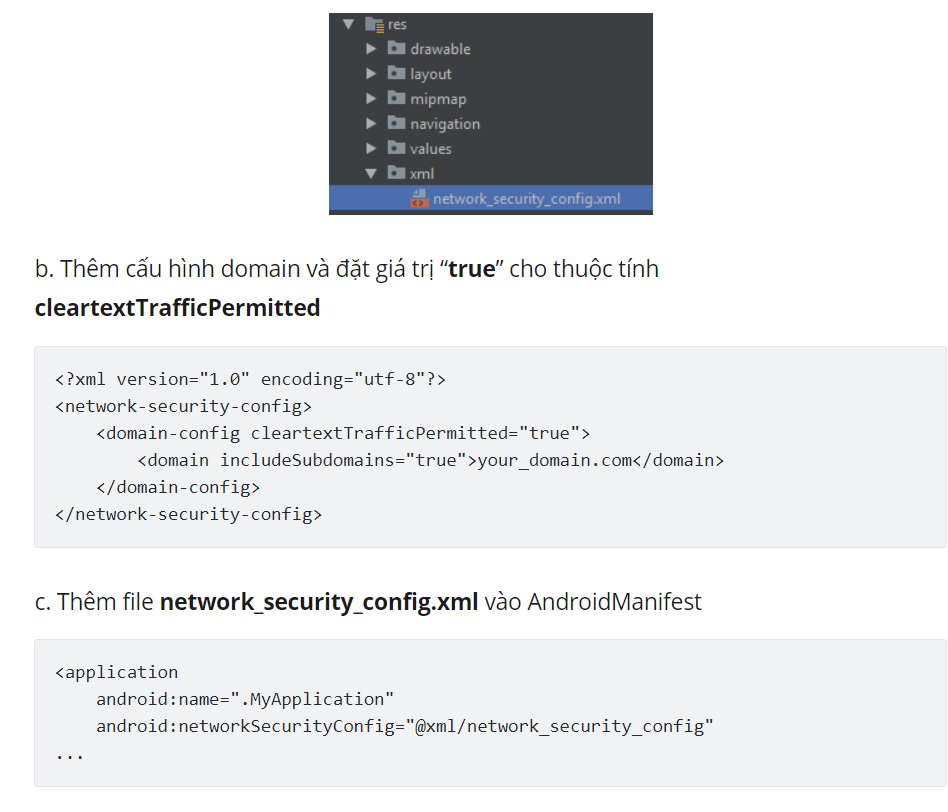
override fun onResponse(call: Call, response: Response) {  
 // Sử dụng thread để chạy ưu tiên, vì khi chạy bt thì val body = response?.body()?.string() chạy sau nên không lên dữ liệu  
 Thread(*Runnable* **{** runOnUiThread(**{** try {  
 //Đọc json  
 val body = response?.body()?.string()  
 // JSON - > OBJECT  
 // Bóc json sang model SearchResponse  
 val searchResponse = gson.fromJson(body, SearchResponse::class.*java*)  
 var arrayResponse: ArrayList<SearchResponse> = ArrayList()  
 arrayResponse.add(searchResponse);  
 lvYoutube.*adapter* = ListViewAdapter(this@ListYoutubeActivity,arrayResponse)  
 } catch (error: Exception) {  
 *println*("Error: " + error)  
 }  
 **}**)  
 **}**).start()  
}

- Cần thêm : mới show được ảnh từ url

android:usesCleartextTraffic="true"

vào trong application

cách khác để show ảnh:



- Sử dụng recycler view và loadmore khi kéo cuối trang  
Không đặt tên giống với các class và các hàm mặc định

class RecyclerViewAdapter(var context: Context, var responseList: ArrayList<MakeCloudModel>) :  
 RecyclerView.Adapter<RecyclerView.ViewHolder>(){  
  
 // Trạng thái còn dữ liệu hay không:  
 private val VIEW\_TYPE\_ITEM = 0  
 private val VIEW\_TYPE\_LOADING = 1  
  
  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecyclerView.ViewHolder {  
//Khi còn dữ liệu thì hiển thị, hết dữ liệu thì hiện ProgressBar  
 return if (viewType === VIEW\_TYPE\_ITEM) {  
 val view: View = LayoutInflater.from(parent?.*context*).inflate(R.layout.*recyclerview\_youtube*, parent, false)  
 return DataViewHolder(view)  
 } else {  
 val view: View = LayoutInflater.from(parent.*context*).inflate(R.layout.*progress\_bar*, parent, false)  
 return LoadingViewHolder(view)  
 }  
 }  
  
 override fun onBindViewHolder(holder: RecyclerView.ViewHolder, position: Int) {  
 if(holder is DataViewHolder){  
 showData(holder,position)  
 }else if(holder is LoadingViewHolder){  
 showLoadingView(holder,position)  
 }  
  
 }  
  
 override fun getItemCount(): Int {  
 return responseList.size +1  
 }  
  
  
  
 override fun getItemViewType(position: Int): Int {  
 // Khi item Chạy đến ô cuối +1, thì hiện progressbar  
 if (responseList.size == position) {  
 return VIEW\_TYPE\_LOADING;  
 }  
 return VIEW\_TYPE\_ITEM;  
 }  
  
 // Dành cho hàng trong list  
 class DataViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 var imgThumbnai: ImageView  
 var tvTitle: TextView  
 var tvSinger: TextView  
 var tvPlayCount: TextView  
  
 init {  
 imgThumbnai = itemView.findViewById(R.id.*imgThumbnai*)  
 tvTitle = itemView.findViewById(R.id.*tvTitle*)  
 tvSinger = itemView.findViewById(R.id.*tvSinger*)  
 tvPlayCount = itemView.findViewById(R.id.*tvPlayCount*)  
  
 }  
 }  
 // Dành cho progressbar  
 class LoadingViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 var progressBar: ProgressBar  
  
 init {  
 progressBar = itemView.findViewById(R.id.*progressbar*)  
 }  
 }  
  
 //  
 private fun showLoadingView(holder: LoadingViewHolder, position: Int) {  
 }  
  
 // Add data to row  
 private fun showData(holder: DataViewHolder, position: Int) {  
 var responseData: MakeCloudModel = responseList[position]  
 Glide.with(context).load(responseData.pictures.extra\_large).into(holder.imgThumbnai)  
 holder.tvTitle.*text* = responseData.name  
 holder.tvSinger.*text* = "Ca sĩ: " + responseData.user.name  
 holder.tvPlayCount.*text* = "Lượt nghe: " + responseData.play\_count.toString()  
 }  
}

- Bên act:

fun initScrollListener() {  
 rvList.addOnScrollListener(object : RecyclerView.OnScrollListener() {  
  
 override fun onScrolled(recyclerView: RecyclerView, dx: Int, dy: Int) {  
 super.onScrolled(recyclerView, dx, dy)  
 if (recyclerView.*adapter*!!.*itemCount* != 0) {  
 val lastVisibleItemPosition =  
 (recyclerView.*layoutManager* as LinearLayoutManager).findLastCompletelyVisibleItemPosition()  
 if (lastVisibleItemPosition != RecyclerView.*NO\_POSITION* && lastVisibleItemPosition >= recyclerView.*adapter*!!.*itemCount* - 1) {  
 // ADD MORE HERE  
 runHttp(linkSearch)  
 *println*("Link Loadmore"+linkSearch)  
  
  
 }  
 }  
 }  
 })  
}