**~~~ Utama ~~~**

1. set **manifest** tambahkan :

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

1. set gradle app tambahkan binding : **buildFeatures{**

**viewBinding true**

**}**

1. f

**~~~ drawable~~~**

1. membuat bg input :
2. setting ke **<shape>.**
3. tambahkan di dalam nya :

* **<solid color = (warna bg)/>**
* **<corners radius = (menambahkan radius)/>**
* **<stroke color = (warna garis) width = (tebal garis)/>**

1. s
2. membuat bg btn:
3. setting ke <shape>.
4. tambahkan di dalam nya :

* **<solid color = (warna bg)/>**
* **<corners radius = (menambahkan radius)/>**

1. s
2. d

**~~~ Response ~~~**

1. membuat class Response, digunakan untuk memanggil output dari db.
2. memasukan data(String) dan statusCode(int)

**~~~ conHelper ~~~**

1. Menambahkan String url dan context :
2. **private String url = “”;**
3. **private Context context;**
4. lalu memanggil construktornya.
5. membuat function untuk crud :

* untuk get data :

public Response getData(){

Response response = new Response(404, "");

try {

URL urll = new URL(url);

HttpURLConnection con = (HttpURLConnection) urll.openConnection();

con.setRequestMethod("GET");

con.setRequestProperty("Cotent-Type", "application/json; charset=UTF-8");

con.connect();

// Mengubah data ke String

BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream()));

StringBuilder sb = new StringBuilder();

String line = "";

while ((line = br.readLine()) != null){

sb.append(line+"\n");

}

response.data = sb.toString();

response.statusCode = con.getResponseCode();

br.close();

} catch (Exception e) {

Log.*d*("errorcon", e.getMessage());

}

return response;

}

* untuk post data :

// menambahkan object pd function post

public Response postData(JSONObject body){

Response response = new Response(404, "");

try {

URL urll = new URL(url);

HttpURLConnection con = (HttpURLConnection) urll.openConnection();

con.setDoInput(true);

con.setDoOutput(true);

con.setRequestMethod("POST");

con.setRequestProperty("Content-Type", "application/json; charset=UTF-8");

DataOutputStream dos = new DataOutputStream(con.getOutputStream());

dos.writeBytes(body.toString());

dos.flush();

dos.close();

con.connect();

BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream()));

StringBuilder sb = new StringBuilder();

String line = "";

while((line=br.readLine()) != null){

sb.append(line+"\n");

}

response.data = sb.toString();

response.statusCode = con.getResponseCode();

br.close();

} catch (Exception e) {

**// agar output menjadi string dan tidak crash saat kirim data**

Log.*d*("errorcon", e.getMessage());

}

return response;

}

1. s

**~~~ Api ~~~**

1. menambahkan API db, binding, Context dan SharedPrefrences :

* **private String API\_URL = “http://”;**
* **ActivityMainBinding bind;**
* **private Context context;**
* **SharedPrifrences sp;**

1. memanggil **construktor** : .
2. tambahkan construktor bind dan context
3. jika **show foto** maka tambahkan **int id :**

public ShowFoto(Context context, ItemHomeBinding bind, int id) {

this.context = context;

this.bind = bind;

API = "http://10.0.2.2:5000/api/Home/Item/Photo/"+id;

}

1. pada **doInBackground :**
2. jika **get gambar asynctask Response diganti ke Bitmap** :

protected Bitmap doInBackground(String... urls) {

Bitmap mIcon11 = null;

try {

URL url = new URL(API);

HttpURLConnection con = (HttpURLConnection) url.openConnection();

con.setRequestMethod("GET");

InputStream in = con.getInputStream();

mIcon11 = BitmapFactory.*decodeStream*(in);

} catch (Exception e) {

Log.*e*("Error", e.getMessage());

e.printStackTrace();

}

return mIcon11;

}

1. jika **post, update** : menggunakan JSONObject :

tambahkan object jsonobjct pada doinbackground :

JSONObject jo = new JSONObject();

try {

jo.put("email", strings[0]);

jo.put("password", strings[1]);

} catch (JSONException e) {

throw new RuntimeException(e);

}

ConnHelper ch = new ConnHelper(API\_URL, context);

return ch.postData(jo);

1. s
2. jika get data menggunakan **2 kolom** maka tambahkan pada onPostExecute :

* bind.rvHome.setHasFixedSize(true);
* bind.rvHome.setLayoutManager(new GridLayoutManager(context,2));

1. set **onPostExecute :**
2. **API LOGIN** meng validasi jika data kosong dan memasukan ke dlm sp output doInBackground di dalam onPostExecute :

Log.*d*("data", response.data);

if (response.data.isEmpty()){

Toast.*makeText*(context, "Data yang anda masukan salah!", Toast.*LENGTH\_SHORT*).show();

}else{

JSONObject jo = new JSONObject(response.data

);

sp = context.getSharedPreferences("LOGIN", Context.*MODE\_PRIVATE*);

SharedPreferences.Editor editor = sp.edit();

try {

editor.putString("id", String.*valueOf*(jo.getInt("id")));

editor.putString("email", jo.getString("email"));

editor.putString("name", jo.getString("name"));

editor.putString("brithday", jo.getString("brithday"));

editor.putString("phoneNumber", jo.getString("phoneNumber"));

editor.apply();

} catch (JSONException e) {

throw new RuntimeException(e);

}

Intent i = new Intent(context, HomeActivity.class);

context.startActivity(i);

}

1. **API GET DATA TO RV :**

protected void onPostExecute(Response response) {

super.onPostExecute(response);

bind.rvHome.setHasFixedSize(true);

bind.rvHome.setLayoutManager(new GridLayoutManager(context,2));

// mengconvert data dari db dan memasukan value ke dalam Model

try {

JSONArray ja = new JSONArray(response.data);

for (int i=0; i<ja.length(); i++){

JSONObject jo = ja.getJSONObject(i);

modelItems.add(new ModelItem(jo.getInt("id"), jo.getString("name"), jo.getString("description"), jo.getInt("price"), jo.getInt("stock")));

}

} catch (JSONException e) {

throw new RuntimeException(e);

}

// memasukan kedalam rv melalui adapter

bind.rvHome.setAdapter(new HomeAdapter(context, modelItems));

}

1. **API GET PICTURE :**

**protected void onPostExecute(Bitmap img) {**

**super.onPostExecute(img);**

**bind.imgViewHome.setImageBitmap(img);**

**}**

1. **s**

**~~~ Model~~~**

1. membuat model yg berisi berdasarkan tabel db.
2. s

**~~~ Cara MEmbuat SharedPrefrences ~~~**

1. coding menyimpan data shared prefrences :

**SharedPrefrences sp = getSharedPrefrences(“NAMA”, CONTEXT.MODE\_PRIVATE);**

**SharedPrefrences.Editor edt = sp.Edit();**

// menyimpan data menggunakan put

**edt.putString/int(“Terserah”, ~ISI~);**

// tutup sp

**edt.apply();**

1. coding memanggil data Shared prefrences :

**SharedPrefrences sp = getSharedPrefrences(“NAMA”, CONTEXT.MODE\_PRIVATE);**

// memanggil nama put harus sesuai

**String trsrh = sp.getString = (“Terserah”, “”);**

1. jika sp yg d panggil ingin di jadikan string maka harus menambahkan (+””), pada akhir nama **biar tidak crash**.
2. s

**~~~ Adapter ~~~**

1. membuat adapter dengan menambahkan extends :

public class HomeAdapter extends RecyclerView.Adapter<HomeAdapter.ViewHolder>

1. memanggil object Context dan List<ModelAnda> untuk menggambil data dri db.
2. lalu memanggil Construktor nya :

public HomeAdapter(Context context, List<ModelItem> data) {

this.context = context;

this.data = data;

}

1. memanggil ItemBinding ke dalam onCreateViewHolder agar dapat di masukan ke rv activity utama :

**ItemHomeBinding bind = ItemHomeBinding.inflate(LayoutInflater.from(parent.getContext), parent, false);**

**return new ViewHolder(bind);** // memanggil class viewHolder

1. pada getItemCount tambahkan :

**return data.size();**

1. coding class viewHolder :

public class ViewHolder extends RecyclerView.ViewHolder {

ItemHomeBinding bind;

public ViewHolder(@NonNull ItemHomeBinding i) {

super(i.getRoot());

bind = i;

}

}

1. s

**~~~ Session ~~~**

1. buat class Session pada file Helper.
2. untuk menambahkan array untuk menyimpan ke cart :

public class Session {

SharedPreferences sp;

SharedPreferences.Editor editor;

int PRIV\_MODE = 0;

Context context;

private static final String *PRIV\_NAME* = "itemCarts";

private static final ArrayList<ModelCart> *KEY\_CART* = new ArrayList<ModelCart>();

public Session(Context context) {

this.context = context;

sp = context.getSharedPreferences(*PRIV\_NAME*, PRIV\_MODE);

editor = sp.edit();

}

public void addToCart(String id, String judul, String count, String price){

*KEY\_CART*.add(new ModelCart(id, judul, count, price));

}

public ArrayList<ModelCart> getKeyCart() {return *KEY\_CART*;}

}

1. memanggil session :

private void setup() {

LinearLayoutManager linearLayoutManager = new LinearLayoutManager(CartActivity.this, bind.rvCart.*VERTICAL*, false);

bind.rvCart.setLayoutManager(linearLayoutManager);

Session session = new Session(this);

CartAdapter adapter = new CartAdapter(session.getKeyCart(), CartActivity.this);

bind.rvCart.setAdapter(adapter);

}

1. menggisi session :

String id = bind.productId.getText().toString();

String Judul = bind.productName.getText().toString();

String jmlh = bind.etJumlah.getText().toString();

String harga = bind.tvTotHargaProduct.getText().toString();

Session session = new Session(DetailActivity.this);

session.addToCart(id, Judul, jmlh, harga);