BAB IV

Android Studio #2

4.1 Tujuan

1. Praktikan mampu mengetahui lingkungan pengembangan Sistem Operasi *Mobile* Android.
2. Praktikan mampu memahami model pengembangan MVP (*Model – View – Presenter)*
3. Praktikan memahami konsep penggunaan API *(Application Program Interface)*
4. Praktikan memahami konsep penggunaan retrofit.
5. Praktikan mampu membuat aplikasi Android sederhana menggunakan model pengembangan MVP.
6. Praktikan mampu melakukan deploy program ke device android masing-masing.

4.2 Alat dan Bahan

1. PC / Notebook (minimal RAM 4GB)

   PC / Notebook yang dugunakan yaitu yang memiliki RAM minimal 4 GB, hal ini dilakukan demi menunjang kelancaran saat praktikum.

Gambar 4.1. PC / Notebook

2. Android Studio 3.2, JDK, Android SDK

   Android Studio adalah Integrated Development Enviroment (IDE) untuk sistem operasi Android, yang dibangung diatas perangkat lunak JetBrains IntelliJ IDEA dan didesain khusus untuk pengembangan Android. IDE ini merupakan pengganti dari Eclipse Android Development Tools (ADT) yang sebelumnya merupakan IDE utama untuk pengembangan aplikasi android.

Gambar 4.2. Android Studio

JDK (Java Development Kit) adalah Perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode java ke bytecode yang dapat dimengerti dan dapat dijalankan oleh JRE (Java Runtime Envirotment). JDK

wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis java, namun tidak wajib terinstall di komputer yang akan menjalankan aplikasi yang dibangun dengan java.

Gambar 4.3. Logo Java

Android SDK (Software Development Kit) adalah satu set alat pengembangan yang digunakan untuk mengembangkan aplikasi untuk platform Android.

Gambar 4.4. Android SDK

3. Device Android dan kabel USB

Device yang digunakan di praktikum BAB IV adalah perangkat berjalan yang memiliki operasi sistem Android yang memiliki dukungan kabel data USB.

Gambar 4.5. Device Android dan kabel USB

4.3 Dasar Teori

4.3.1. API

API adalah singkatan dari *Application Program Interface*, dan memungkinkan *developer* untuk mengintegrasikan dua bagian dari aplikasi atau dengan aplikasi yang berbeda secara bersamaan. API terdiri dari berbagai elemen seperti function, protocols, dan tools lainnya yang memungkinkan *developer* untuk membuat aplikasi. Tujuan penggunaan API adalah untuk mempercepat proses *development* dengan menyediakan *function* secara terpisah sehingga developer tidak perlu membuat fitur yang serupa. Penerapan API akan sangaat terasa jika fitur yang diinginkan sudah sangat kompleks, tentu membutuhkan waktu untuk membuat yang serupa dengannya. Misalnya: integrasi dengan *payment gateway*. Terdapat berbagai jenis sistem API yang dapat digunakan, termasuk sistem operasi, *library, dan web*.

Gambar 4.6. API

(Sumber: codepolitan.com)

4.3.2. Retrofit

Retrofit adalah library Android yang dibuat oleh Squre yang digunakan sebagai REST Client pada Android, yang pasti akan memudahkan kita dalam programing. Karena kita tidak perlu lagi untuk membuat method-method sendiri untuk menggunakan REST Client API dari backend.

Gambar 4.7. Retrofit

(sumber:medium.com)

4.3.3. MVP

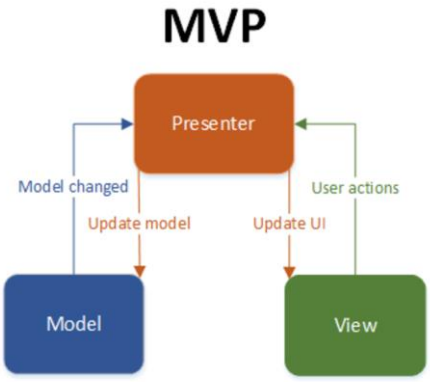
MVP (Model-View-Presenter) adalah pola arsitektur antarmuka pengguna dirancang untuk memfasilitasi unit testing otomatis dan meningkatkan pemisahan keprihatinan di logika presentasi:

a) Model adalah antarmuka yang mendefinisikan data yang akan ditampilkan atau ditindaklanjuti dalam antarmuka pengguna.

b) View adalah antarmuka pasif yang menampilkan data (model) dan mengarahkan perintah pengguna (kejadian) ke presenter untuk bertindak atas data tersebut.

c) Presenter bertindak berdasarkan model dan pandangan. Ini mengambil data dari repositori (model), dan memformatnya untuk ditampilkan dalam tampilan.

Biasanya, implementasi tampilan instantiates objek presenter beton, menyediakan referensi untuk dirinya sendiri.



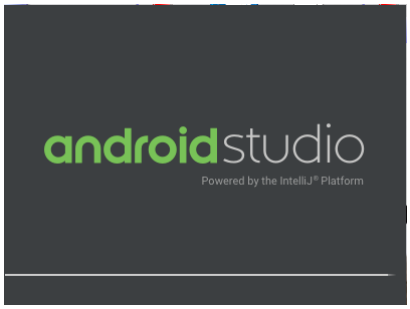
Gambar 4.8. MVP

(sumber:medium.com)

4.4 Langkah Kerja

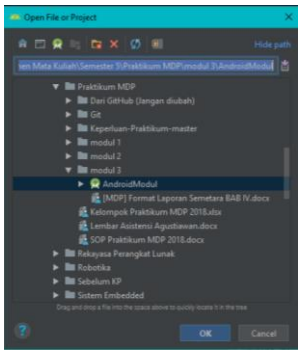
4.4.1 Persiapan Package

1. Buka Android Studio



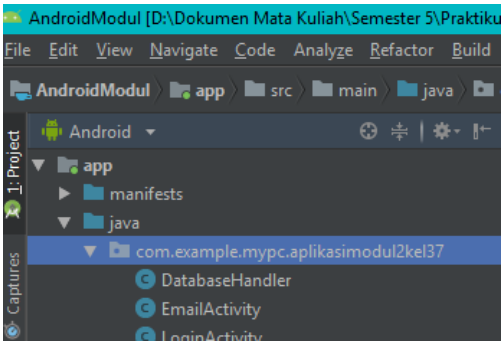
Gambar 4.9. Android Studio

2. Buka backup file project Modul 2, dengan cara klik **File – Open** – Arahkan ke File Project Android Modul 2



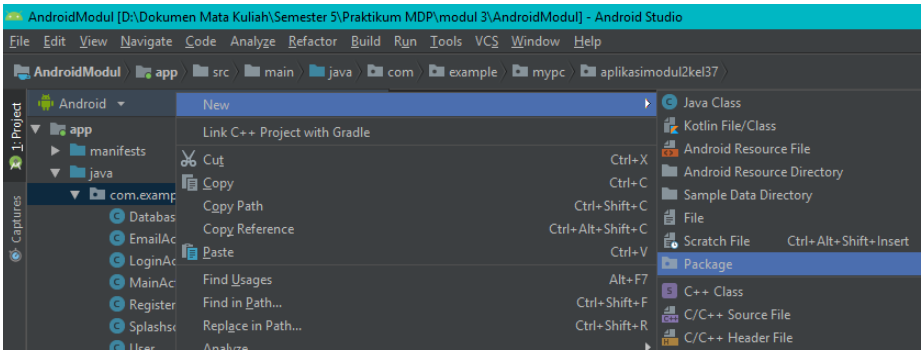
Gambar 4.10. File - Open

3. Tunggu hingga **build.gradle** dan **indexing** selesai seluruhnya
4. Pada tab **project**, pilih mode **Android**.



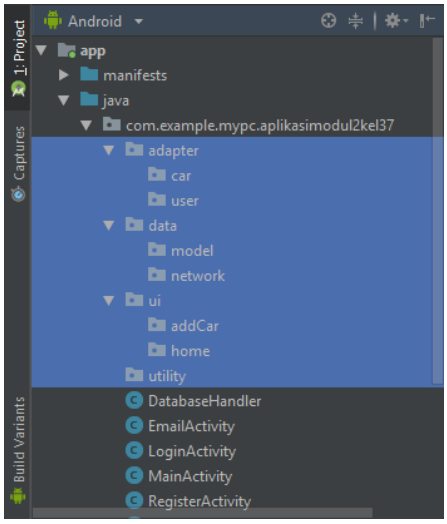
Gambar 4.11. Pilih Mode Android

5. Lalu buat package baru dengan cara klik kanan pada **main package java – new – package**



Gambar 4.12 New Package

6. Tambahkan beberapa package baru sehingga menjadi seperti berikut

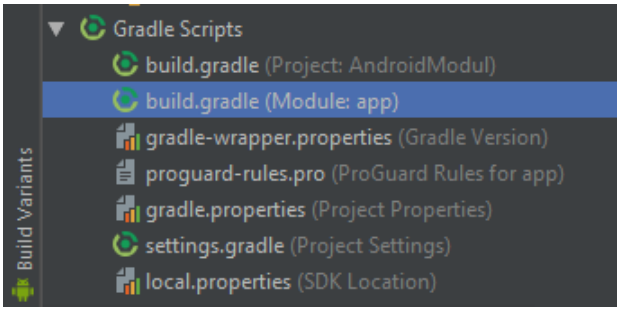


Gambar 4.13. Tambah Beberapa Package

Package – package ini akan digunakan untuk menampung class – class dalam metode MVP sesuai dengan kegunaanya, dengan ini dokumentasi projek akan lebih mudah dan terstruktur.

4.4.2 Persiapan Retrofit

1. Selanjutnya, tambahkan beberapa dependencies ke build.gradle. Dobel klik **build.gradle (Module:app)**.



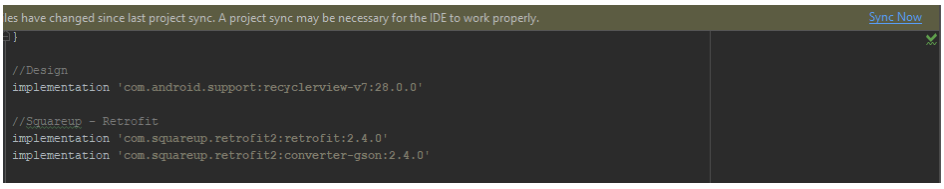
Gambar 4.14. Double click build.grade

2. Lalu tambahkan, code berikut

```
//Design
implementation 'com.android.support:recyclerview-v7:28.0.0'

//Squareup - Retrofit
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
```

Retrofit merupakan library yang akan digunakan untuk mengubah API yang diperoleh dari Refstull Server (HTTP) ke dalam bentuk data yang dapat diolah oleh Java

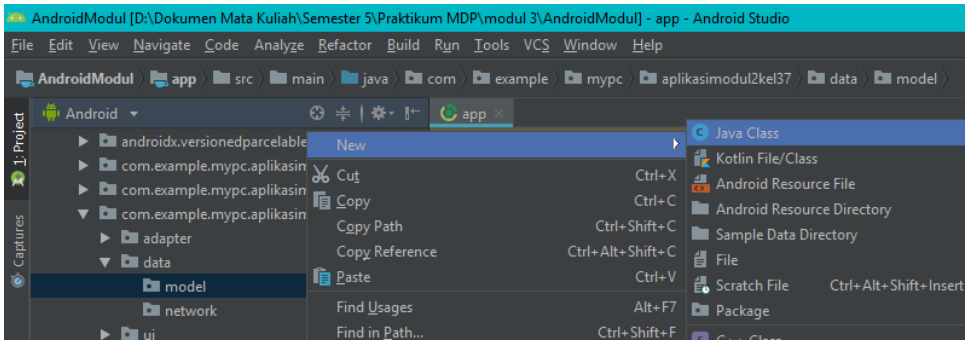
3. Setelahnya, klik tombol **Sync Now** dibagian kanan atas



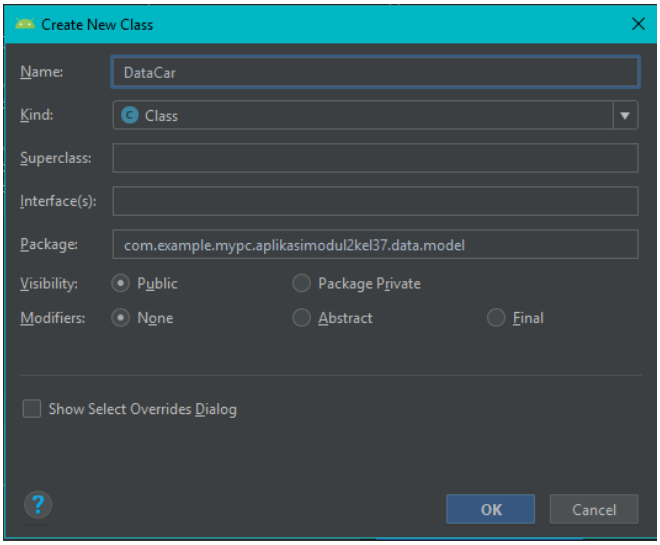
Gambar 4.15. Klik Sync Now

Tunggu hingga **download** dan **build.gradle** selesai

4. Setelah package dan dependencies, selanjutnya mempersiapkan Retrofit client. Buat class model pada package **data - model** dengan nama **DataCar**



Gambar 4.16. Membuat Java Class



Gambar 4.17. Membuat New Class DataCar

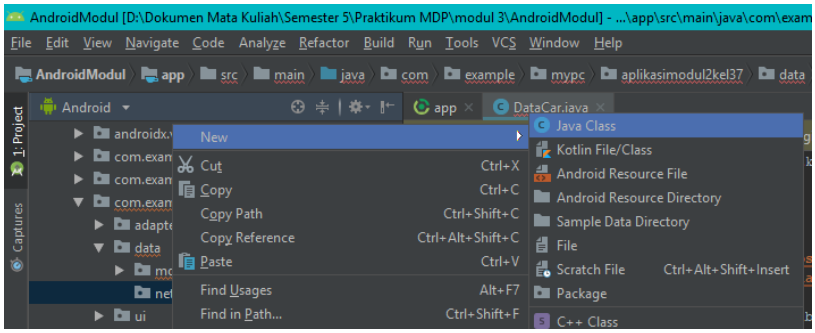
Pada model ini, akan menggunakan POJO (Plain Old Java Object) yaitu class yang berdiri sendiri dan tidak ada ketergantungan dengan class lain. Pada class DataCar tambahkan implements Parcalable untuk membungkus data sehingga dapat dikirimkan/dipindahkan antar activity. Kemudian tambahkan, code berikut,

```
import android.os.Parcel;
import android.os.Parcelable;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

public class DataCar implements Parcelable {

    @SerializedName("id")
    @Expose
    private Integer id;
    @SerializedName("name")
    @Expose
```

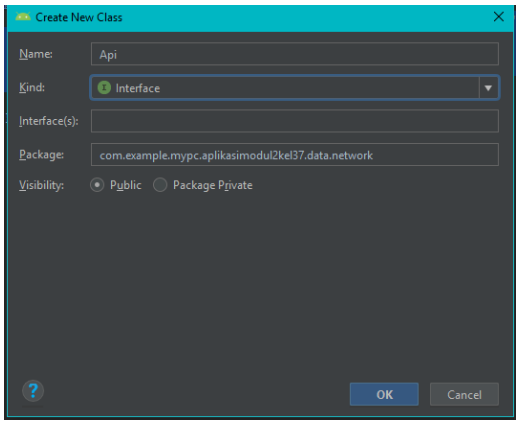
```
        private String name;
        @SerializedName("merk")
        @Expose
        private String merk;
        @SerializedName("model")
        @Expose
        private String model;
        @SerializedName("year")
        @Expose
        private String year;

        protected DataCar(Parcel in) {
            if (in.readByte() == 0) {
                id = null;
            } else {
                id = in.readInt();
            }
            id = in.readInt();
            name = in.readString();
            merk = in.readString();
            model = in.readString();
            year = in.readString();
        }

        public static final Creator<DataCar> CREATOR = new
Creator<DataCar>() {
            @Override
            public DataCar createFromParcel(Parcel in) {
                return new DataCar(in);
            }

            @Override
            public DataCar[] newArray(int size) {
                return new DataCar[size];
            }
        };

        public Integer getId() {
            return id;
        }

        public void setId(int id) {
            this.id = id;
        }

        public String getName() {
            return name;
        }

        public void setName(String name) {
            this.name = name;
        }

        public String getMerk() {
            return merk;
        }
```

```
    public void setMerk(String merk) {
        this.merk = merk;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getYear() {
        return year;
    }

    public void setYear(String year) {
        this.year = year;
    }

    @Override
    public String toString() {
        return "DataUser{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", merk='" + merk + '\'' +
                ", model='" + model + '\'' +
                ", year='" + year + '\'' +
                '}';
    }

    @Override
    public int describeContents() {
        return 0;
    }

    @Override
    public void writeToParcel(Parcel dest, int flags) {
        if (id == null) {
            dest.writeByte((byte) 0);
        } else {
            dest.writeByte((byte) 1);
            dest.writeInt(id);
        }
        dest.writeInt(id);
        dest.writeString(name);
        dest.writeString(merk);
        dest.writeString(model);
        dest.writeString(year);
    }
}
```

5. Buat **interface** pada package **data - network** dengan nama **Api** sama seperti langkah sebelumnya, namun pada window box, pilih **interface** lalu beri nama **Api**



Gambar 4.18. Membuat Java Class



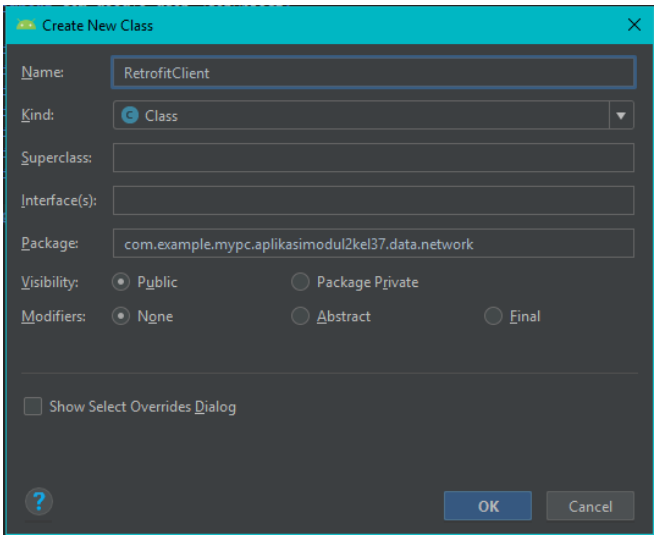
Gambar 4.19. Membuat New Class Interface Api

Interface ini digunakan untuk menginisialisasi key pada saat POST, GET, DELETE dan lain-lain ke Endpoint Restfull API. Pada praktikum ini akan menggunakan POST dan GET (all). Kemudian tambahkan, code berikut,

```
import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.Headers;
import retrofit2.http.POST;
import retrofit2.http.Query;

public interface Api {

    @Headers("Content-Type: application/json")

    @GET("getAllCar.php")
    Call<JsonObject> getAllCar();

    @GET("getCarById.php")
```

```
        Call<JsonObject> GetCarById(
                @Query("id") int id
        );

        @FormUrlEncoded
        @POST("addCar.php")
        Call<JsonObject> addCar(
                @Field("name") String name,
                @Field("merk") String merk,
                @Field("model") String model,
                @Field("year") String year
        );

    }
```

6. Buat class baru pada package **data – network** dengan nama **RetrofitClient**, yang berfungsi untuk inisialisasi library retrofit.



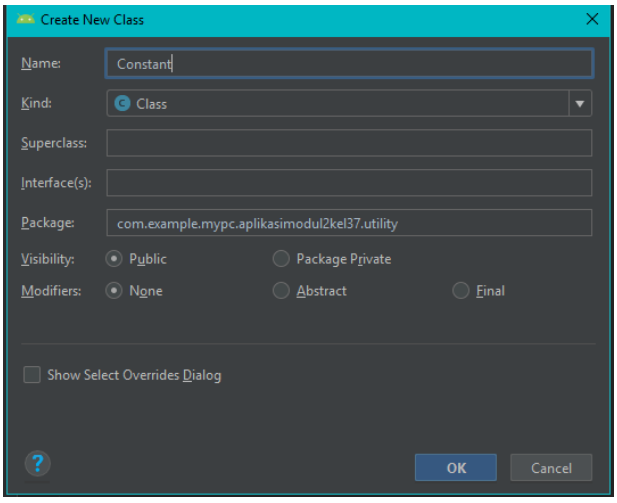
Gambar 4.20. Membuat New Class RetrofitClient

Kemudian tambahkan, code berikut,

```
import java.util.concurrent.TimeUnit;

import okhttp3.OkHttpClient;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {
    private static RetrofitClient retrofitClient;

    private RetrofitClient(){}

    public static RetrofitClient getInstance(){
        if(retrofitClient == null){
            retrofitClient = new RetrofitClient();
        }
```

```
            return retrofitClient;
        }

        public Api getApi(){
            return getRetrofit().create(Api.class);
        }

        public Retrofit getRetrofit(){

            final     OkHttpClient     okHttpClient     =     new
OkHttpClient.Builder()
                    .readTimeout(15, TimeUnit.SECONDS)
                    .connectTimeout(15, TimeUnit.SECONDS)
                    .build();

            return new Retrofit.Builder()
                    .baseUrl(Constant.BASE_URL)

.addConverterFactory(GsonConverterFactory.create())
                    .client(okHttpClient)
                    .build();
        }
    }
```

7. Buat class pada package **utility** dengan nama **Constant**, yang akan berisikan variable - variable konstan atau tetap. **Tambahkan** (bukan replace) code berikut kedalam class **Constant**



Gambar 4.21. Membuat New Class Constant

Kemudian tambahkan, code berikut,

```
   public      static      final      String      BASE_URL      =
"https://vizyan.000webhostapp.com/";

   public final class Extra {
   public static final String DATA = "EXTRA_DATA";
   }
```

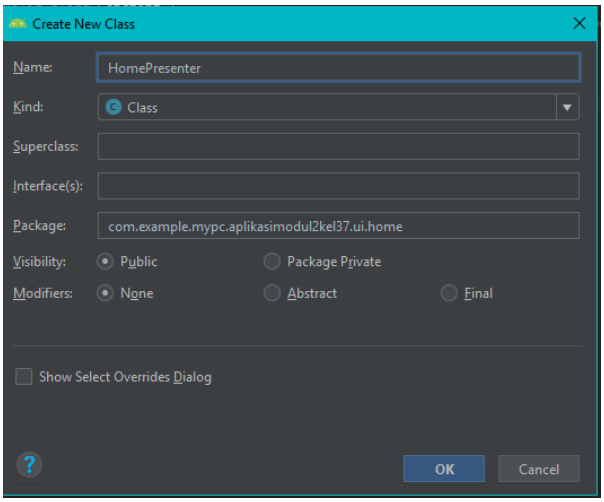
6.4.3. Pembuatan Activity

Selanjutnya, membuat activity – activity yang diperlukan pada aplikasi. Dengan konsep MVP, setiap activity nantinya terdiri dari 3 class. 1 class activity, 1 class presenter dan 1 class interface (view).

- Activity berfungsi untuk menghubungkan System Aplikasi dengan User
- Presenter berfungsi untuk menghubungkan Model dengan Retrofil Client
- Interface (View) berfungsi untuk menghubungkan Activity dengan Presenter

1. Membuat home

a. Pertama, buatlah presenter pada package **ui - home** dengan cara yang sama seperti membuat class biasa. Beri nama **HomePresenter.**

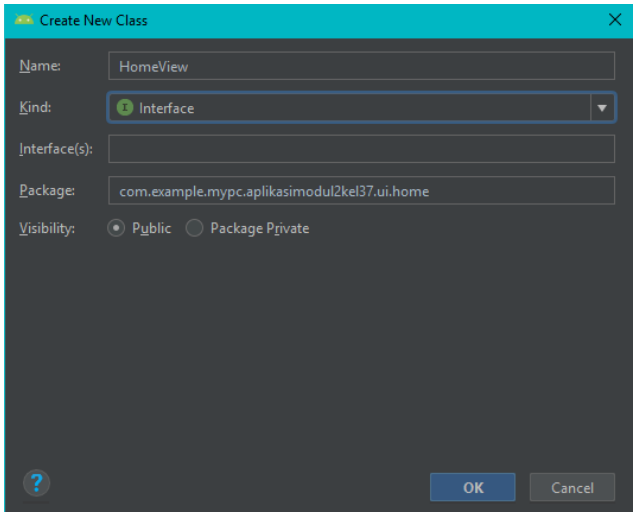


Gambar 4.22. Membuat Class HomePresenter

```
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class HomePresenter {
```

```
        private HomeView homeView;

        public HomePresenter (HomeView homeView){
            this.homeView = homeView;
        }

        public void getAllCar(){
            final String tag = "Home-getAllCar";
            RetrofitClient.getInstance()
                    .getApi()
                    .getAllCar()
                    .enqueue(new Callback<JsonObject>() {
                        @Override
                        public                                  void
onResponse(Call<JsonObject>     call,     Response<JsonObject>
response) {
                            if (response.isSuccessful()){
                                JsonObject        body        =
response.body();
                                JsonArray        array        =
body.get("result").getAsJsonArray();
                                Type      type      =      new
TypeToken<List<DataCar>>(){}.getType();
                                List<DataCar>  dataCars  =  new
Gson().fromJson(array, type);

homeView.successShowCar(dataCars);
                                Log.e(tag,
response.body().toString());
                            } else {
                                homeView.failedShowCar("Maaf
terjadi kesalahan");
                                Log.e(tag,
response.errorBody().toString());
                            }
                        }

                        @Override
                        public void onFailure(Call<JsonObject>
call, Throwable t) {
                            homeView.failedShowCar("Maaf
terjadi kesalahan");
                            Log.d(tag,
t.getMessage().toString());
                        }
                    });
        }
    }
```

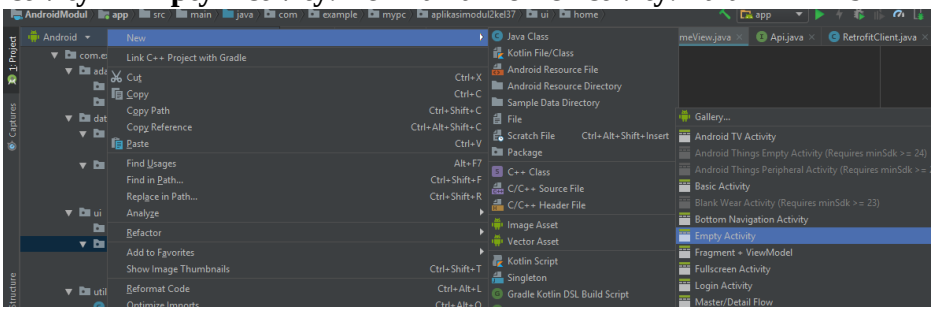
b. Setelah itu buat **interface** pada package **ui - home** dengan nama **HomeView**. Lalu **tambahkan** (bukan replace) code berikut didalam classnya



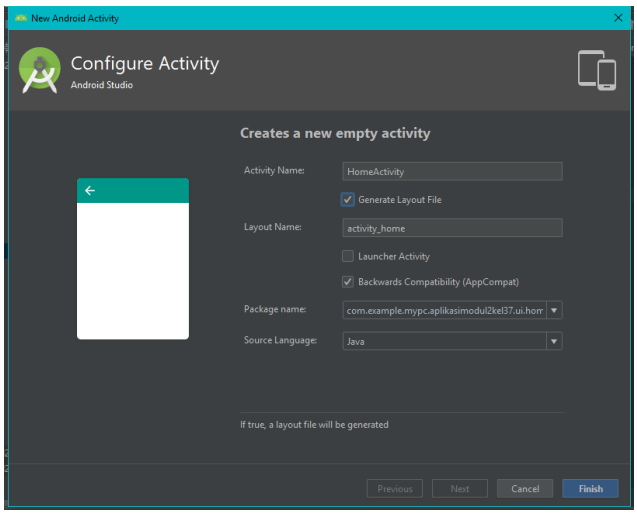
Gambar 4.23. Membuat Interface HomeView

```
    void successShowCar(List<DataCar> dataCars);
    void failedShowCar(String message);
```

c. Selanjutnya, buat activity dengan cara klik kanan package **home – New – Activity – Empty Activity**. Beri nama **HomeActivity**. Lalu klik finish

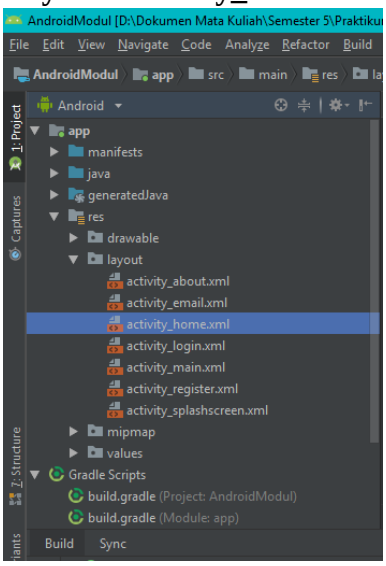


Gambar 4.24. Membuat activity baru



Gambar 4.25. Membuat HomeActivity

d. Sebelum ke code activity, design terlebih dahulu tampilan pada layout activity nya. Klik package **res – layout – activity_home.xml**



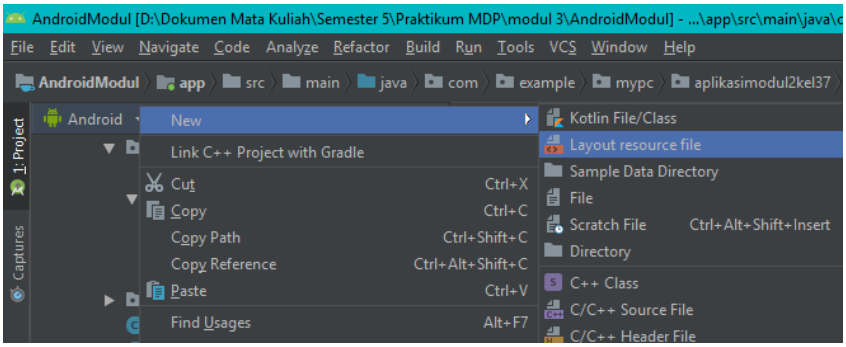
Gambar 4.26. Memilih activity_home.xml

e. Pilih mode text, lalu tambahkan code berikut kedalamnya

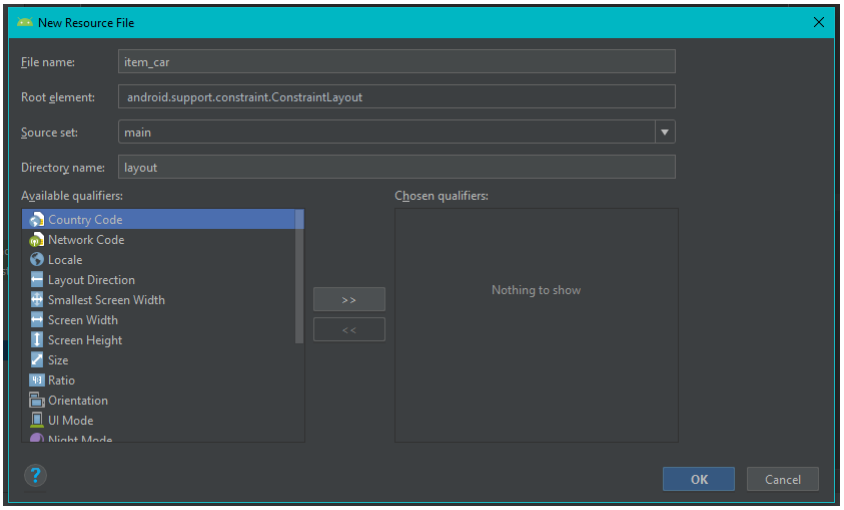
```
   <?xml version="1.0" encoding="utf-8"?>
   <android.support.v4.widget.SwipeRefreshLayout
xmlns:android="http://schemas.android.com/apk/res/android"
       xmlns:app="http://schemas.android.com/apk/res-auto"
       xmlns:tools="http://schemas.android.com/tools"
       android:id="@+id/srlHome"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       tools:context=".ui.home.HomeActivity">

       <android.support.constraint.ConstraintLayout
           android:layout_width="match_parent"
           android:layout_height="match_parent">


<android.support.design.widget.FloatingActionButton
               android:id="@+id/fabHome"
               android:layout_width="wrap_content"
               android:layout_height="wrap_content"
               android:layout_marginEnd="15dp"
               android:layout_marginBottom="15dp"
               android:clickable="true"

app:backgroundTint="@color/design_default_color_primary"
               app:fabSize="normal"

app:layout_constraintBottom_toBottomOf="parent"
               app:layout_constraintEnd_toEndOf="parent"
               app:srcCompat="@android:drawable/ic_menu_add"
/>
```

```
            <android.support.v7.widget.RecyclerView
                android:id="@+id/rvHome"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_marginEnd="5dp"
                android:layout_marginStart="5dp"

app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

        </android.support.constraint.ConstraintLayout>
    </android.support.v4.widget.SwipeRefreshLayout>
```

f. Buat layout item, dengan cara klik kanan pada **res – layout – Layout Resource File**. Beri nama **item_car**



Gambar 4.27. Membuat new Layout resorce file



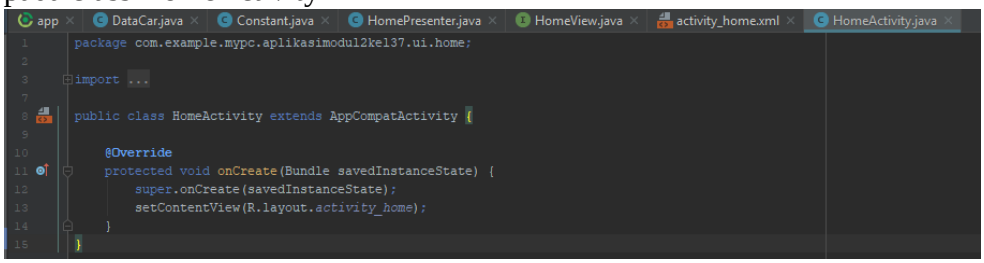
Gambar 4.28. Membuat layout resource file item_car

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
```

```
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:minHeight="50dp"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_marginTop="5dp"
        android:layout_marginStart="10dp"
        android:layout_marginEnd="10dp"
        app:cardCornerRadius="15dp"
        android:clickable="true">

        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:minHeight="50dp"
            android:clickable="true">

            <LinearLayout
                android:id="@+id/llItemC"
                android:layout_width="match_parent"
                android:layout_height="60dp"
                android:clickable="true"
                android:orientation="horizontal">

                <ImageView
                    android:id="@+id/ivItemC"
                    android:layout_width="35dp"
                    android:layout_height="35dp"
                    android:layout_gravity="center"
                    android:layout_marginStart="20dp"
                    android:src="@mipmap/ic_launcher" />

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:gravity="center">

                    <TextView
                        android:id="@+id/tvItemCName"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_gravity="center"
                        android:layout_marginStart="15dp"
                        android:text="Name"
                        android:textSize="18sp"
                        android:textStyle="bold"
                        tools:text="" />

                    <TextView
                        android:id="@+id/tvItemCMerk"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_gravity="center"
```

```
                        android:layout_marginStart="15dp"
                        android:text="Name"
                        android:textSize="18sp"
                        tools:text="" />
                </LinearLayout>

            </LinearLayout>
        </RelativeLayout>

    </android.support.v7.widget.CardView>
```

g. Selanjutnya tambahkan kode pada HomeActivity. Dengan cara dobel click pada class HomeActivity



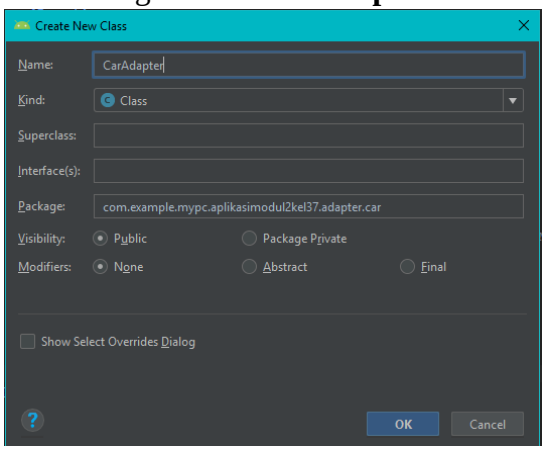
Gambar 4.29. Menambahkan kode pada HomeActivity

```
    import android.content.Intent;
    import android.support.design.widget.FloatingActionButton;
    import android.support.v4.widget.SwipeRefreshLayout;
    import android.support.v7.app.AppCompatActivity;
    import android.os.Bundle;
    import android.support.v7.widget.LinearLayoutManager;
    import android.support.v7.widget.RecyclerView;
    import android.widget.Toast;

    import java.util.List;

    public class HomeActivity extends AppCompatActivity
implements HomeView, CarListerner{

        private HomePresenter homePresenter;
        private CarAdapter carAdapter;
        FloatingActionButton fabHome;
        RecyclerView rvHome;
        SwipeRefreshLayout srlHome;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_home);
            initPresenter();
            initView();
            initDataPresenter();

            addCar();
```

```
            refresh();
        }

        private void initView() {
            fabHome = findViewById(R.id.fabHome);
            rvHome = findViewById(R.id.rvHome);
            srlHome = findViewById(R.id.srlHome);
        }

        private void initPresenter() {
            homePresenter = new HomePresenter(this);
        }

        private void initDataPresenter() {
            homePresenter.getAllCar();
        }

        private void addCar() {
            fabHome.setOnClickListener(v -> {
                Intent addCar = new Intent(HomeActivity.this,
AddActivity.class);
                startActivity(addCar);
                finish();
            });
        }

        private void refresh(){
            srlHome.setOnRefreshListener(()                    ->
initDataPresenter());
        }

        @Override
        public void successShowCar(List<DataCar> dataCars) {
            if (srlHome.isRefreshing()){
                srlHome.setRefreshing(false);
            }
            carAdapter = new CarAdapter(dataCars);
            carAdapter.setAdapterListener(this);
            rvHome.setLayoutManager(new
LinearLayoutManager(this));
            rvHome.setAdapter(carAdapter);
        }

        @Override
        public void failedShowCar(String message) {
            Toast.makeText(this,  "Maaf  terjadi  kesalahan",
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onCarClick(DataCar dataCar) {
            //Intent     intent     =     new     Intent(this,
DetailActivity.class);
            //intent.putExtra(Constant.Extra.DATA, dataCar);
            //startActivity(intent);
        }
```

```
    }
```

h. Setelahnya, buat adapter untuk menampung list/daftar data API yang didapatkan, dalam praktikum ini yaitu daftar API Car. Buat class baru pada package **adapter – car** dengan nama **CarAdapter.**

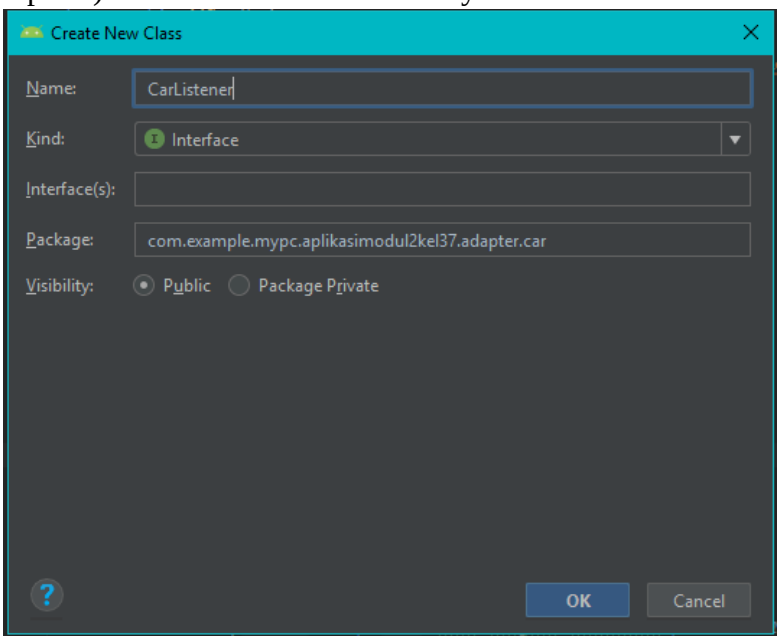


Gambar 4.30. Membuat class CarAdapter

```
   import android.support.annotation.NonNull;
   import android.support.v7.widget.RecyclerView;
   import android.view.LayoutInflater;
   import android.view.View;
   import android.view.ViewGroup;

   import java.util.List;

   public          class          CarAdapter          extends
RecyclerView.Adapter<CarViewHolder> {

       private List<DataCar> dataCars;
       private CarListerner carListerner;

       public CarAdapter(List<DataCar> dataCars){
           this.dataCars = dataCars;
       }

       public      void      setAdapterListener(CarListerner
carListerner){
           this.carListerner = carListerner;
       }

       @NonNull
       @Override
       public    CarViewHolder    onCreateViewHolder(@NonNull
ViewGroup parent, int viewType) {
           View view = LayoutInflater
                   .from(parent.getContext())
                   .inflate(R.layout.item_car,        parent,
false);
           return new CarViewHolder(view);
```

```
        }

        @Override
        public  void  onBindViewHolder(@NonNull  CarViewHolder
holder,  int position) {
            DataCar dataUser = get(position);
            holder.bind(dataUser, carListerner);
        }

        private DataCar get(int position) {
            return dataCars.get(position);
        }

        @Override
        public int getItemCount() {
            if (dataCars == null) return 0;
            return dataCars.size();
        }
    }
```

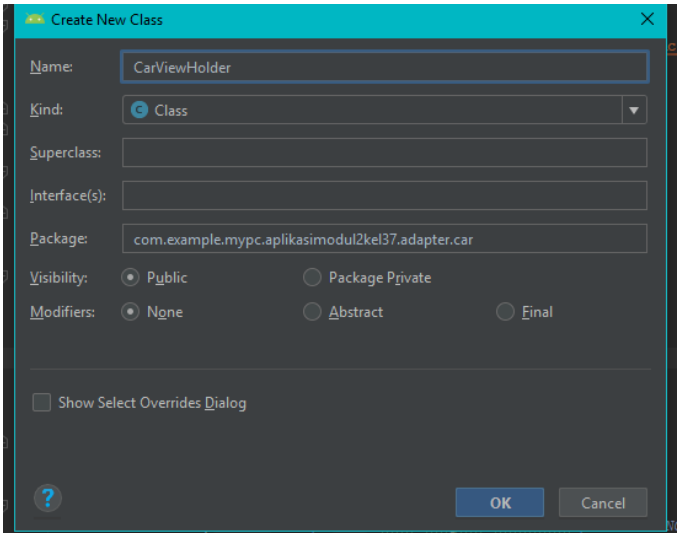
i. Buat **interface** di package yang sama dengan nama CarListener. **Tambahkan** (bukan replace) code berikut didalam classnya



Gambar 4.31. Membuat class CarListerner

```
void onCarClick(DataCar dataCar);
```

j. Buat class di package yang sama dengan nama CarViewHolder



Gambar 4.32. Membuat class CarViewHolder

```
   import android.support.v7.widget.RecyclerView;
   import android.view.View;
   import android.widget.LinearLayout;
   import android.widget.TextView;

   public class CarViewHolder extends RecyclerView.ViewHolder
{

       LinearLayout linearLayout;
       TextView tvHomeName;
       TextView tvHomeMerk;

       public CarViewHolder(View itemView) {
           super(itemView);
           initView(itemView);
       }

       private void initView(View itemView) {
           linearLayout                                    =
itemView.findViewById(R.id.llItemC);
           tvHomeName                                      =
itemView.findViewById(R.id.tvItemCName);
           tvHomeMerk                                      =
itemView.findViewById(R.id.tvItemCMerk);
       }

       public   void   bind(final   DataCar   dataCar,   final
CarListerner carListerner) {
           tvHomeName.setText(dataCar.getName());
           tvHomeMerk.setText(dataCar.getMerk());

           linearLayout.setOnClickListener(v               ->
carListerner.onCarClick(dataCar));
       }
   }
```

2. Membuat addCar

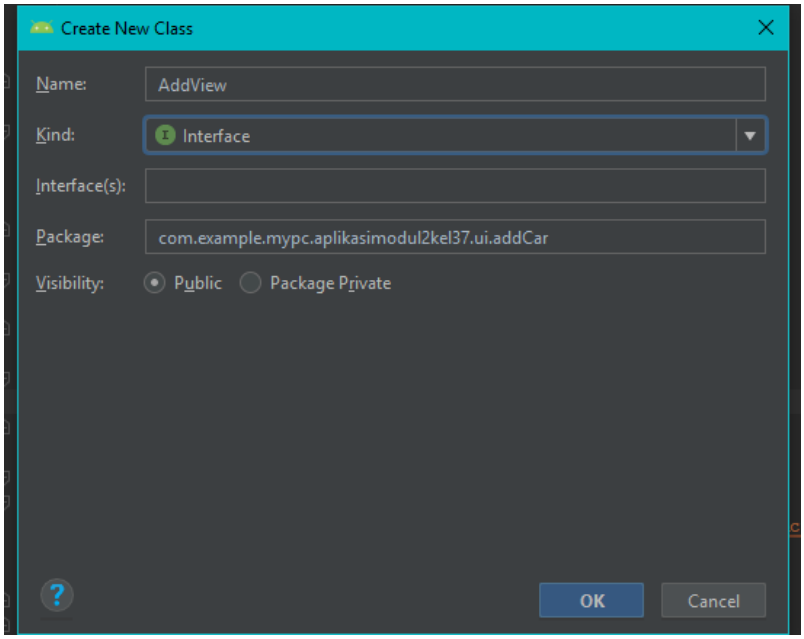
Sama seperti pada home, tambahkan presenter, interface (view) dan activity dengan nama masing – masing. AddPresenter, AddView dan AddActivity. Tambahkan didalam **package addCar** lalu masing – masing replace code yang ada didalamnya dengan code yang ada pada file.txt dibawah

a. AddPresenter

```
import android.util.Log;

import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class AddPresenter {

    private AddView addView;

    public AddaPresenter(AddView addView){
        this.addView = addView;
    }

    public void addCar(){
        final String tag = "Add-addCar";
        String name = addView.getName();
        String merk = addView.getMerk();
        String model = addView.getModel();
        String year = addView.getYear();

        RetrofitClient.getInstance()
                .getApi()
                .addCar(name, merk, model, year)
                .enqueue(new Callback<JsonObject>() {
                    @Override
                    public                              void
onResponse(Call<JsonObject>     call,     Response<JsonObject>
response) {
                        if(response.isSuccessful()){
                            addView.successAddCar();
                            Log.e(tag,
response.body().toString());
                        } else {
                            addView.failedAddCar();
                            Log.e(tag,
response.body().toString());
                        }
                    }

                    @Override
                    public void onFailure(Call<JsonObject>
call, Throwable t) {
```

```
                                addView.failedAddCar();
                                Log.e(tag,
t.getMessage().toString());
                            }
                        });
            }
    }
```

b. AddView (interface)



Gambar 4.33. Membuat Interface AddView

```
    public interface AddView {

        String getName();

        String getMerk();

        String getModel();

        String getYear();

        void successAddCar();

        void failedAddCar();
    }
```

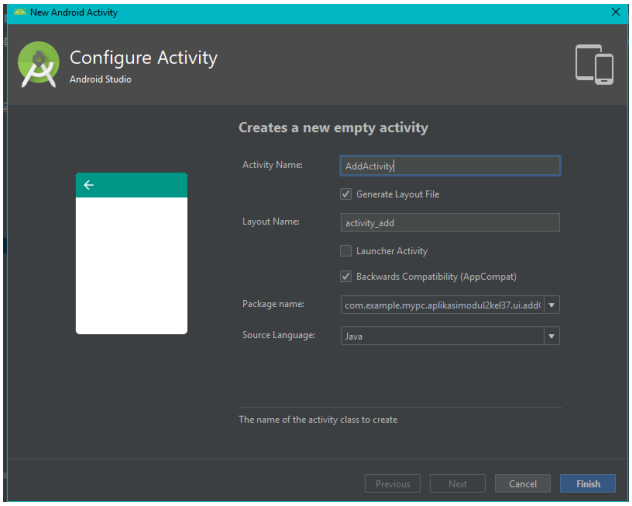
c. activity_add.xml

```
    <?xml version="1.0" encoding="utf-8"?>
    <android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        tools:context=".ui.addCar.AddActivity">

        <EditText
            android:id="@+id/etAddName"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:layout_marginStart="10dp"
            android:layout_marginTop="10dp"
            android:ems="10"
            android:hint="Nama"
            android:inputType="textPersonName"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <EditText
            android:id="@+id/etAddMerk"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:layout_marginStart="10dp"
            android:layout_marginTop="5dp"
            android:ems="10"
            android:hint="Merk"
            android:inputType="textPersonName"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/etAddName" />

        <EditText
            android:id="@+id/etAddModel"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:layout_marginStart="10dp"
            android:layout_marginTop="5dp"
            android:ems="10"
            android:hint="Model"
            android:inputType="textPersonName"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/etAddMerk" />

        <EditText
            android:id="@+id/etAddYear"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:layout_marginStart="10dp"
            android:layout_marginTop="5dp"
            android:ems="10"
            android:hint="Tahun"
```

```
            android:inputType="number"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/etAddModel" />

        <Button
            android:id="@+id/btnAdd"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:layout_marginTop="10dp"
            android:text="Tambah"
            app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/etAddYear" />
    </android.support.constraint.ConstraintLayout>
```

d. AddActivity



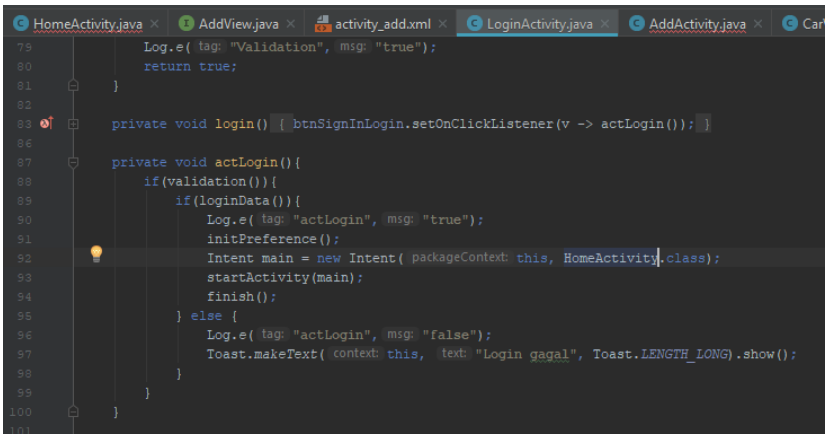
Gambar 4.34. Membuat activity AddActivity

```
    import android.content.Intent;
    import android.support.v7.app.AppCompatActivity;
    import android.os.Bundle;
    import android.widget.Button;
    import android.widget.EditText;
    import android.widget.Toast

    public class AddActivity extends AppCompatActivity
implements AddView{

        private AddPresenter addPresenter;
        EditText etAddName, etAddMerk, etAddModel, etAddYear;
        Button btnAdd;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_add);
            initPresenter();
```

```
            initView();

            addCar();
        }

        private void initView() {
            etAddName = findViewById(R.id.etAddName);
            etAddMerk = findViewById(R.id.etAddMerk);
            etAddModel = findViewById(R.id.etAddModel);
            etAddYear = findViewById(R.id.etAddYear);
            btnAdd = findViewById(R.id.btnAdd);
        }

        private void initPresenter() {
            addPresenter = new AddPresenter(this);
        }

        private void addCar() {
            btnAdd.setOnClickListener(v -> {
                addPresenter.addCar();
            });
        }

        @Override
        public String getName() {
            return etAddName.getText().toString();
        }

        @Override
        public String getMerk() {
            return etAddMerk.getText().toString();
        }

        @Override
        public String getModel() {
            return etAddModel.getText().toString();
        }

        @Override
        public String getYear() {
            return etAddYear.getText().toString();
        }

        @Override
        public void successAddCar() {
            Toast.makeText(this, "Berhasil Menambahkan Mobil",
Toast.LENGTH_SHORT).show();
            Intent   home   =   new   Intent(AddActivity.this,
HomeActivity.class);
            startActivity(home);
            finish();
        }

        @Override
        public void failedAddCar() {
```

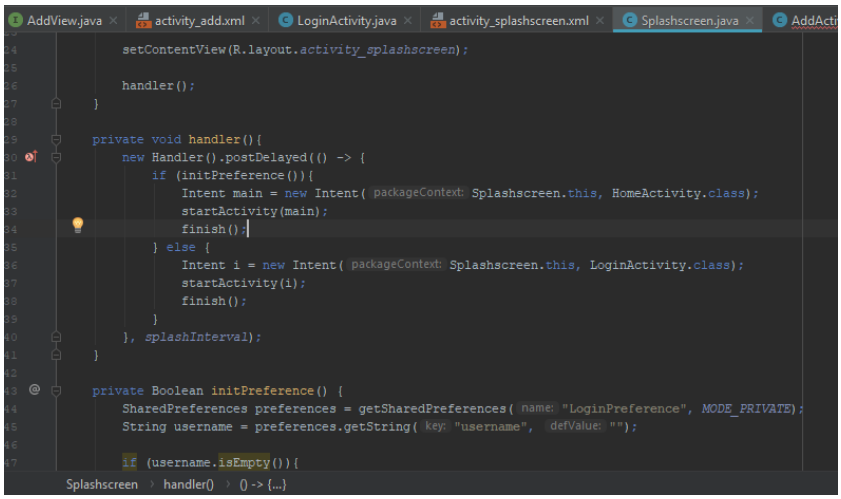
```
            Toast.makeText(this,    "Gagal    Menambah    Mobil",
Toast.LENGTH_SHORT).show();
        }
    }
```

3. Pada LoginActivity, ganti bagian intent yang mengarah ke MainActivity menjadi HomeActivity. Seperti pada gambar



Gambar 4.35. Mengubah MainActivity menjadi HomeActivity

4. Ganti pula code pada activity Splashscreen, seperti gambar dibawah

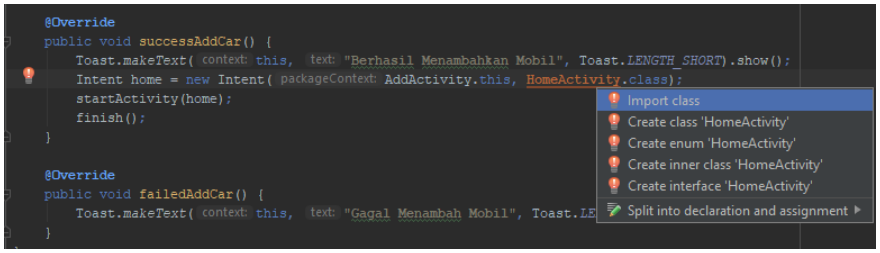


Gambar 4.36. Mengganti pada splashscreen

5. Terakhir tambahkan permission pada manifest untuk akses internet dengan cara klik file **Manifest** seperti pada gambar berikut, lalu tambahkan

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

6. Cek semua class yang ada, apabila terdapat code berwarna merah. Arahkan kursor ke code tersebut lalu tekan tombol kombinasi pada keyboard **alt + enter**. Lalu pilih **import class**



Gambar 4.37. Import class

7. Deploy aplikasi ke emulator android atau ke smartphone. Jangan lupa, aktifkan terlebih dahulu **debugging mode** pada smartphone masing – masing melalui developer mode
8. Teliti lagi langkah kerja bila masih gagal, ingat **case-sensitive**

4.5 Analisa Hasil Percobaan

1. DataCar

```
package com.example.mypc.aplikasimodul2kel37.data.model;

import android.os.Parcel;
import android.os.Parcelable;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

public class DataCar implements Parcelable { //

    @SerializedName("id")
    @Expose
    private Integer id;
    @SerializedName("name")
    @Expose
    private String name;
    @SerializedName("merk")
    @Expose
    private String merk;
    @SerializedName("model")
    @Expose
    private String model;
    @SerializedName("year")
    @Expose
    private String year;

    protected DataCar(Parcel in) {
        if (in.readByte() == 0) {
            id = null;
        } else {
            id = in.readInt();
        }
        id = in.readInt();
        name = in.readString();
        merk = in.readString();
        model = in.readString();
        year = in.readString();
    }

    public static final Creator<DataCar> CREATOR = new
Creator<DataCar>() {
        @Override
        public DataCar createFromParcel(Parcel in) {
            return new DataCar(in);
        }

        @Override
        public DataCar[] newArray(int size) {
            return new DataCar[size];
        }
    };

    public Integer getId() {
```

```
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getMerk() {
    return merk;
}

public void setMerk(String merk) {
    this.merk = merk;
}

public String getModel() {
    return model;
}

public void setModel(String model) {
    this.model = model;
}

public String getYear() {
    return year;
}

public void setYear(String year) {
    this.year = year;
}

@Override
public String toString() {
    return "DataUser{" +
            "id=" + id +
            ", name='" + name + '\'' +
            ", merk='" + merk + '\'' +
            ", model='" + model + '\'' +
            ", year='" + year + '\'' +
            '}';
}

@Override
public int describeContents() {
    return 0;
}

@Override
```

```
    public void writeToParcel(Parcel dest, int flags) {
        if (id == null) {
            dest.writeByte((byte) 0);
        } else {
            dest.writeByte((byte) 1);
            dest.writeInt(id);
        }
        dest.writeInt(id);
        dest.writeString(name);
        dest.writeString(merk);
        dest.writeString(model);
        dest.writeString(year);
    }
}
```

Pada file DataCar ini berisikan objek-objek yang akan digunakan beserta tipe datanya. Objek-objek yang digunakan yaitu id, name, merk, model, dan year yang mana tipe datanya adalah integer untuk id string untuk name, merk, model, dan year. Terdapat juga setter getter yang berguna untuk menambahkan data maupun menampilkan data. `public String toString()` digunakan untuk memasukkan data mobil ke dalam database.

2. Api

```
package com.example.mypc.aplikasimodul2kel37.data.network;

import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.Headers;
import retrofit2.http.POST;
import retrofit2.http.Query;

public interface Api {

    @Headers("Content-Type: application/json")

    @GET("getAllCar.php")
    Call<JsonObject> getAllCar();

    @GET("getCarById.php")
    Call<JsonObject> GetCarById(
            @Query("id") int id
    );

    @FormUrlEncoded
    @POST("addCar.php")
    Call<JsonObject> addCar(
```

```
            @Field("name") String name,
            @Field("merk") String merk,
            @Field("model") String model,
            @Field("year") String year
    );

}
```

File API berisikan tentang pengambilan data end point yang terdapat pada link yang telah diberikan, seperti getAllCar yang digunakan untuk menampilkan semua data mobil, getCarById yang digunakan untuk menampilkan data mobil sesuai Id nya, dan addCar untuk menambahkan data mobil. Ditentukan juga tipe data yang akan dipakai.

3. RetrofitClient

```
package com.example.mypc.aplikasimodul2kel37.data.network;

import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.concurrent.TimeUnit;

import okhttp3.OkHttpClient;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {
    private static RetrofitClient retrofitClient;

    private RetrofitClient(){}

    public static RetrofitClient getInstance(){
        if(retrofitClient == null){
            retrofitClient = new RetrofitClient();
        }

        return retrofitClient;
    }

    public Api getApi(){
        return getRetrofit().create(Api.class);
    }

    public Retrofit getRetrofit(){

        final OkHttpClient okHttpClient = new
OkHttpClient.Builder() //pakai http client untuk waktu maksimal
koneksi
                .readTimeout(15, TimeUnit.SECONDS)
                .connectTimeout(15, TimeUnit.SECONDS)
                .build();
```

```
        return new Retrofit.Builder()
                .baseUrl(Constant.BASE_URL)

.addConverterFactory(GsonConverterFactory.create())
                .client(okHttpClient)
                .build();
    }
}
```

Retrofit adalah library Android yang dibuat oleh Squre yang digunakan sebagai REST Client pada Android. RetrofitClient berisi tentang retrofit client yang akan digunakan pada aplikasi ini. Pengaturannya yaitu seperti retrofitClient, getApi, dan juga getRetrofit.

4. HomePresenter

```
package com.example.mypc.aplikasimodul2kel37.ui.home;

import android.util.Log;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import
com.example.mypc.aplikasimodul2kel37.data.network.RetrofitClient;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class HomePresenter {

    private HomeView homeView;

    public HomePresenter (HomeView homeView){
        this.homeView = homeView;
    }

    public void getAllCar(){
        final String tag = "Home-getAllCar";
        RetrofitClient.getInstance()
                .getApi()
                .getAllCar()
                .enqueue(new Callback<JsonObject>() {
                    @Override
                    public void onResponse(Call<JsonObject> call,
Response<JsonObject> response) { //kalau kita terhubung ke server
                        if (response.isSuccessful()){ //jika data
benar
```

```
                            JsonObject body = response.body();
                            JsonArray array =
body.get("result").getAsJsonArray();
                            Type type = new
TypeToken<List<DataCar>>(){}.getType();
                            List<DataCar> dataCars = new
Gson().fromJson(array, type);
                            homeView.successShowCar(dataCars);
                            Log.e(tag,
response.body().toString());
                        } else { //jika data salah
                            homeView.failedShowCar("Maaf terjadi
kesalahan");
                            Log.e(tag,
response.errorBody().toString());
                        }
                    }

                    @Override
                    public void onFailure(Call<JsonObject> call,
Throwable t) { //kalau responnya gagal
                        homeView.failedShowCar("Maaf terjadi
kesalahan");
                        Log.d(tag, t.getMessage().toString());
                    }
                });
    }
}
```

Home presenter berisikan tentang pengambilan data menggunakan retrofit dari Api pada getAllCar.php. Didalam terdapat sourcecode tentang respon pengambilan data tersebut. Jika berhasil akan menampilkan datanya namun jika gagal akan menampilkan kalimat “Maaf terjadi kesalahan”.

5. HomeView

```
package com.example.mypc.aplikasimodul2kel37.ui.home;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

import java.util.List;

public interface HomeView {
    void successShowCar(List<DataCar> dataCars);
    void failedShowCar(String message);
}
```

HomeView disini berfungsi untuk menghubungkan antara HomeActivity dengan HomePresenter. Di dalamnya berisi void untuk successShowCar dan juga void failedShowCar. Jika successShowCar maka akan ditampilkan list yang berisi

daftar mobil, sedangkan jika failedShowCar maka akan ditampilkan message atau pesannya.

6. HomeActivity

```
package com.example.mypc.aplikasimodul2kel37.ui.home;

import android.content.Intent;
import android.support.design.widget.FloatingActionButton;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import
com.example.mypc.aplikasimodul2kel37.adapter.car.CarAdapter;
import
com.example.mypc.aplikasimodul2kel37.adapter.car.CarListener;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel37.ui.addCar.AddActivity;
import
com.example.mypc.aplikasimodul2kel37.ui.detail.DetailActivity;
import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.List;

public class HomeActivity extends AppCompatActivity implements
HomeView, CarListener {

    private HomePresenter homePresenter;
    private CarAdapter carAdapter;
    FloatingActionButton fabHome;
    RecyclerView rvHome;
    SwipeRefreshLayout srlHome;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        initPresenter();
        initView();
        initDataPresenter();

        addCar();
        refresh();
    }

    private void initView() {
        fabHome = findViewById(R.id.fabHome);
        rvHome = findViewById(R.id.rvHome);
        srlHome = findViewById(R.id.srlHome);
    }
```

```
    private void initPresenter() {
        homePresenter = new HomePresenter(this);
    }

    private void initDataPresenter() {
        homePresenter.getAllCar();
    }

    private void addCar() {
        fabHome.setOnClickListener(v -> {
            Intent addCar = new Intent(HomeActivity.this,
AddActivity.class);
            startActivity(addCar);
            finish();
        });
    }

    private void refresh(){
        srlHome.setOnRefreshListener(() -> initDataPresenter());
    }

    @Override
    public void successShowCar(List<DataCar> dataCars) {
        if (srlHome.isRefreshing()){
            srlHome.setRefreshing(false);
        }
        carAdapter = new CarAdapter(dataCars);
        carAdapter.setAdapterListener(this);
        rvHome.setLayoutManager(new LinearLayoutManager(this));
        rvHome.setAdapter(carAdapter);
    }

    @Override
    public void failedShowCar(String message) {
        Toast.makeText(this, "Maaf terjadi kesalahan",
Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onCarClick(DataCar dataCar) {
        Intent intent = new Intent(this, DetailActivity.class);
        intent.putExtra(Constant.Extra.DATA, dataCar);
        startActivity(intent);
    }
}
```

Terdapat inisialisasi homePresenter, carAdapter, FloatingActionButton, RecyclerView, dan SwipeRefreshLayout. Kemudian terdapat beberapa method yang akan dipakai seperti initView, initPresenter, initDataPresenter, addCar, dan juga refresh.

7. Activity_home.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.SwipeRefreshLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/srlHome"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.home.HomeActivity">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <android.support.design.widget.FloatingActionButton
            android:id="@+id/fabHome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="15dp"
            android:layout_marginBottom="15dp"
            android:clickable="true"

app:backgroundTint="@color/design_default_color_primary"
            app:fabSize="normal"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:srcCompat="@android:drawable/ic_menu_add" />

        <android.support.v7.widget.RecyclerView
            android:id="@+id/rvHome"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginEnd="5dp"
            android:layout_marginStart="5dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

    </android.support.constraint.ConstraintLayout>
</android.support.v4.widget.SwipeRefreshLayout>
```

Activity_home.xml otomatis terbuat saat pembuatan HomeActivity. File ini berfungsi untuk membuat tampilan halaman utama saat program dijalankan setelah login. Dalam pengaturannya dibuat menjadi constraint layout. Terdapat FloatingActionButton yang berfungsi untuk menambah data mobil pada halaman home. Terdapat pula RecyclerView yang berfungsi untuk menampilkan data mobil yang sudah dimasukkan.

8. Item_car.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<android.support.v7.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="50dp"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_marginTop="5dp"
    android:layout_marginStart="10dp"
    android:layout_marginEnd="10dp"
    app:cardCornerRadius="15dp"
    android:clickable="true">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:minHeight="50dp"
        android:clickable="true">

        <LinearLayout
            android:id="@+id/llItemC"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:clickable="true"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/ivItemC"
                android:layout_width="35dp"
                android:layout_height="35dp"
                android:layout_gravity="center"
                android:layout_marginStart="20dp"
                android:src="@mipmap/ic_launcher" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical"
                android:gravity="center">

                <TextView
                    android:id="@+id/tvItemCName"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
                    android:layout_marginStart="15dp"
                    android:text="Name"
                    android:textSize="18sp"
                    android:textStyle="bold"
                    tools:text="" />

                <TextView
                    android:id="@+id/tvItemCMerk"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
```

```
                    android:layout_marginStart="15dp"
                    android:text="Name"
                    android:textSize="18sp"
                    tools:text="" />
            </LinearLayout>

        </LinearLayout>
    </RelativeLayout>

</android.support.v7.widget.CardView>
```

Item_car.xml berfungsi untuk mengatur tampilan daftar atau list mobil di halaman home. Pada file ini menggunakan relative layout. Terdapat image view yang digunakan untuk menampilkan logo ic_launcher dan juga terdapat text view yang digunakan untuk menampilkan nama dan merk data mobil.

9. CarAdapter

```
package com.example.mypc.aplikasimodul2kel37.adapter.car;

import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

import java.util.List;

public class CarAdapter extends
RecyclerView.Adapter<CarViewHolder> {

    private List<DataCar> dataCars;
    private CarListener carListerner;

    public CarAdapter(List<DataCar> dataCars){
        this.dataCars = dataCars;
    }

    public void setAdapterListener(CarListener carListerner){
        this.carListerner = carListerner;
    }

    @NonNull
    @Override
    public CarViewHolder onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
        View view = LayoutInflater
                .from(parent.getContext())
                .inflate(R.layout.item_car, parent, false);
```

```
        return new CarViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull CarViewHolder holder,
int position) {
        DataCar dataUser = get(position);
        holder.bind(dataUser, carListerner);
    }

    private DataCar get(int position) {
        return dataCars.get(position);
    }

    @Override
    public int getItemCount() {
        if (dataCars == null) return 0;
        return dataCars.size();
    }
}
```

CarAdapter berfungsi untuk menampung data API yang mana API sendiri berfungsi untuk mengintegrasikan aplikasi yang berbeda secara bersamaan.

10. CarListener

```
package com.example.mypc.aplikasimodul2kel37.adapter.car;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

public interface CarListener {
    void onCarClick(DataCar dataCar);
}
```

CarListener berisi method pengambilan data car dari dataCar.

11. CarViewHolder

```
package com.example.mypc.aplikasimodul2kel37.adapter.car;

import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.TextView;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

public class CarViewHolder extends RecyclerView.ViewHolder {

    LinearLayout linearLayout;
    TextView tvHomeName;
    TextView tvHomeMerk;

    public CarViewHolder(View itemView) {
        super(itemView);
```

```
        initView(itemView);
    }

    private void initView(View itemView) {
        linearLayout = itemView.findViewById(R.id.llItemC);
        tvHomeName = itemView.findViewById(R.id.tvItemCName);
        tvHomeMerk = itemView.findViewById(R.id.tvItemCMerk);
    }

    public void bind(final DataCar dataCar, final CarListener
carListerner) {
        tvHomeName.setText(dataCar.getName());
        tvHomeMerk.setText(dataCar.getMerk());

        linearLayout.setOnClickListener(v ->
carListerner.onCarClick(dataCar));
    }
}
```

CarViewholder digunakan untuk pengaturan tampilan list pada home activity. Terdapat inisialisasi LinearLayout, tvHomeName, dan juga tvHomeMerk. Lalu pada initView menyambungkan objek dengan tampilan yang berada pada item_car.xml. Untuk pengisian TextViewnya diambil dari DataCar lalu di setText ke tvHomeName dan tvHomeMerk.

12. AddPresenter

```
package com.example.mypc.aplikasimodul2kel37.ui.addCar;

import android.util.Log;

import
com.example.mypc.aplikasimodul2kel37.data.network.RetrofitClient;
import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class AddPresenter {

    private AddView addView;

    public AddPresenter(AddView addView){
        this.addView = addView;
    }

    public void addCar(){
        final String tag = "Add-addCar";
        String name = addView.getName();
        String merk = addView.getMerk();
```

```
        String model = addView.getModel();
        String year = addView.getYear();

        RetrofitClient.getInstance()
                .getApi()
                .addCar(name, merk, model, year)
                .enqueue(new Callback<JsonObject>() {
                    @Override
                    public void onResponse(Call<JsonObject> call,
Response<JsonObject> response) {
                        if(response.isSuccessful()){
                            addView.successAddCar();
                            Log.e(tag,
response.body().toString());
                        } else {
                            addView.failedAddCar();
                            Log.e(tag,
response.body().toString());
                        }
                    }

                    @Override
                    public void onFailure(Call<JsonObject> call,
Throwable t) {
                        addView.failedAddCar();
                        Log.e(tag, t.getMessage().toString());
                    }
                });
    }
}
```

AddPresenter berisi method addcar yng berfungsi utuk menambahkan data mobil ke dalam database API yang digunakan. Terdapat inisialisasi objek dan tipe data yang digunakan yaitu name, merk, model, dan year.

13. AddView

```
package com.example.mypc.aplikasimodul2kel37.ui.addCar;

public interface AddView {

    String getName();

    String getMerk();

    String getModel();

    String getYear();

    void successAddCar();

    void failedAddCar();
}
```

File AddView berisi method-method yang mana berfungsi untuk menguhubngkan anatara AddActivity dan AddPresenter.

14. AddActivity

```
package com.example.mypc.aplikasimodul2kel37.ui.addCar;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.ui.home.HomeActivity;


public class AddActivity extends AppCompatActivity implements
AddView{

    private AddPresenter addPresenter;
    EditText etAddName, etAddMerk, etAddModel, etAddYear;
    Button btnAdd;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add);
        initPresenter();
        initView();

        addCar();
    }

    private void initView() {
        etAddName = findViewById(R.id.etAddName);
        etAddMerk = findViewById(R.id.etAddMerk);
        etAddModel = findViewById(R.id.etAddModel);
        etAddYear = findViewById(R.id.etAddYear);
        btnAdd = findViewById(R.id.btnAdd);
    }

    private void initPresenter() {
        addPresenter = new AddPresenter(this);
    }

    private void addCar() {
        btnAdd.setOnClickListener(v -> {
            addPresenter.addCar();
        });
    }

    @Override
    public String getName() {
        return etAddName.getText().toString();
```

```
    }

    @Override
    public String getMerk() {
        return etAddMerk.getText().toString();
    }

    @Override
    public String getModel() {
        return etAddModel.getText().toString();
    }

    @Override
    public String getYear() {
        return etAddYear.getText().toString();
    }

    @Override
    public void successAddCar() {
        Toast.makeText(this, "Berhasil Menambahkan Mobil",
Toast.LENGTH_SHORT).show();
        Intent home = new Intent(AddActivity.this,
HomeActivity.class);
        startActivity(home);
        finish();
    }

    @Override
    public void failedAddCar() {
        Toast.makeText(this, "Gagal Menambah Mobil",
Toast.LENGTH_SHORT).show();
    }
}
```

Pada AddActivity terdapat method utama pada halaman add yang dibuat pada activity_add.xml. Pertama kita inisialisasikan objek-objek yang kita gunakan seperti AddPresenter, EditText, dan button. Lalu inisialisasi method yang akan digunakan pada AddActivity seperti initPresenter, initView, dan addCar. Pada initPresenter kita menginisialisasi presenter yang digunakan yaitu HomePresenter. Lalu pada initView kita menyambungkan objek pada AddActivity dengan yang berada pada activity_add.xml. Lalu pada initPresenter kita menginisialisasi presenter yang kita gunakan yaitu AddPresenter. Pada addCar jika kita tekan tombol add maka akan berpindah ke AddPresenter. Lalu terdapat method getName, getMerk, getModel, dan getYear yang semuanya diambil dari text yang ada pada EditText pada activity_add.xml. Lalu terdapat method jika berhasil menambahkan maka akan menampilkan Toast “Berhasil Menambahkan Mobil” dan terdapat method jika gagal maka akan menampilkan Toast “Gagal Menambah Mobil”.

15. Activity_add.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.addCar.AddActivity">

    <EditText
        android:id="@+id/etAddName"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"
        android:ems="10"
        android:hint="Nama"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/etAddMerk"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Merk"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddName" />

    <EditText
        android:id="@+id/etAddModel"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Model"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddMerk" />

    <EditText
        android:id="@+id/etAddYear"
        android:layout_width="0dp"
```

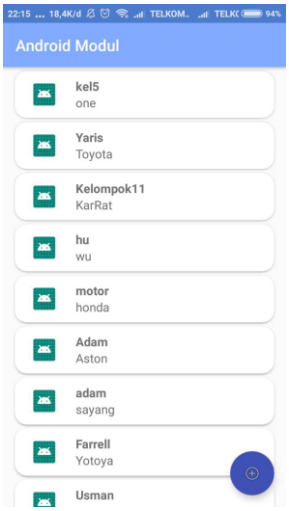
```
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Tahun"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddModel" />

    <Button
        android:id="@+id/btnAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginTop="10dp"
        android:text="Tambah"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddYear" />
</android.support.constraint.ConstraintLayout>
```

Activity_add.xml berfungsi untuk mengatur tampilan atau layout pada halaman menambah data mobil. Terdapat 4 editText yaitu untuk memasukkan Nama, Merk, Model, dan Tahun. Terdapat pula satu tombol atau button yang bertuliskan “Tambah” yang berfungsi untuk menambah data mobil yang dituliskan di editText untuk disimpan di database API.

Berikut adalah hasil tangkapan layar tampilan-tampilan pada praktikum ini,

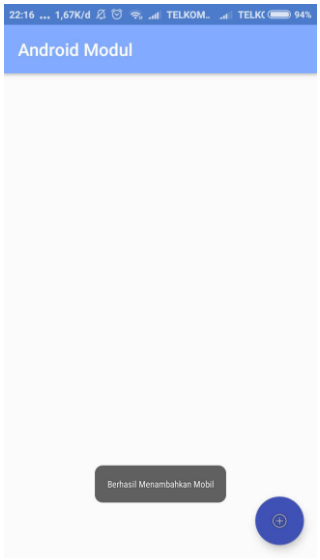
Gambar 4.38. Tampilan splashscreen

Gambar 4.39. Tampilan login



Gambar 4.40. Tampilan Halaman utama

Gambar 4.41. Tampilan Tambah data



Gambar 4.42. Tampilan tambah data sukses

4.6 Tugas

Pada tugas ini, terdapat dua level dimana untuk level 1 praktikan diminta untuk membuat sebuah activity baru bernama DetailCar yang mana apabila ketika user melakukan klik pada data mobil yang ada pada halaman utama maka akan ditampilkan data detail dari mobil tersebut. Berikut adalah langkah-langkah dalam pembuatan activity baru ini,

1. Detail Activity

```
package com.example.mypc.aplikasimodul2kel37.ui.detail;

import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.List;

public class DetailActivity extends AppCompatActivity implements
DetailView {

    private DataCar dataCar;
    private DetailPresenter detailPresenter;
    private TextView tvNama;
    private TextView tvMerk;
    private TextView tvModel;
    private TextView tvTahun;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        initView();
        initIntentData();
        initPresenter();
        initData();
    }

    private void initView() {
        tvNama = findViewById(R.id.tvTampil_nama);
        tvMerk = findViewById(R.id.tvTampil_Merek);
        tvModel = findViewById(R.id.tvTampil_Model);
        tvTahun = findViewById(R.id.tvTampil_Tahun);
    }

    private void initData() {
        detailPresenter.getCarById(dataCar);
```

```
    }

    private void initPresenter() {
        detailPresenter = new DetailPresenter(this);
    }

    private void initIntentData() {
        dataCar =
getIntent().getParcelableExtra(Constant.Extra.DATA);
        if (dataCar == null) finish();
    }

    @Override
    public void showErrorCarById(String message) {
        Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
    }

    @Override
    public void showSuccessCarById(List<DataCar> car) {
        tvNama.setText(car.get(0).getName());
        tvMerk.setText(car.get(0).getMerk());
        tvModel.setText(car.get(0).getModel());
        tvTahun.setText(car.get(0).getYear());
    }
}
```

DetailActivity ini merupakan sebuah activity yang mana akan menampilkan data mobil berupa Nama, Merk, Model, dan Tahun jika showSuccesscarById. Untuk menampilkan data detail dari sebuah mobil menggunakan id. Terdapat inisialisas objek-objek yang akan digunakan seperti DetailPresenter, DataCar, dan EditText. Kemudian diinisialisasikan method yang akan digunakan pada home activity seperti initPresenter, intiView, initIntentData, dan juga initData. Pada initPresenter diinisialisasi presenter yang digunakan yaitu DetailPresenter. Lalu pada initView kita menyambungkan objek pada DetailActivity dengan yang berada pada activity_Detail.xml. Lalu pada initData kita menginisialisasi data yang kita gunakan. Yaitu getCarById diambil dari DetailPresenter. Lalu terdapat method getName, getMerk, getModel, getYear yang berfungsi untuk mengambil data dari database API lalu dipasangkan pada EditText pada activity_detail.xml.

2. Activity_detail.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    tools:context=".ui.detail.DetailActivity">


    <TextView
        android:id="@+id/tvTampil_Tahun"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_Model"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_Model"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_Model"
/>

    <TextView
        android:id="@+id/tvTampil_Model"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_Merek"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_Merek"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_Merek"
/>

    <TextView
        android:id="@+id/tvTampil_Merek"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_nama"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_nama"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_nama"
/>

    <TextView
        android:id="@+id/tvTampil_nama"
        android:layout_width="247dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="48dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"

app:layout_constraintStart_toEndOf="@+id/textView_tampilNama"

app:layout_constraintTop_toBottomOf="@+id/textView5_TambilDetail"
        app:layout_constraintVertical_bias="0.0" />
```

```
    <TextView
        android:id="@+id/textView_tampilNama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:text="@string/namacar"
        android:textSize="24sp"

app:layout_constraintBottom_toBottomOf="@+id/tvTampil_nama"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_nama"
        app:layout_constraintVertical_bias="1.0" />

    <TextView
        android:id="@+id/textView2_TampilMerek"
        android:layout_width="75dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:text="@string/merkcar"
        android:textSize="24sp"

app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Merek"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Merek"
        app:layout_constraintHorizontal_bias="1.0"

app:layout_constraintStart_toStartOf="@+id/textView_tampilNama"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_Merek"
        app:layout_constraintVertical_bias="0.0" />

    <TextView
        android:id="@+id/textView3_TampilModel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:text="@string/modelcar"
        android:textSize="24sp"

app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Model"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Model"

app:layout_constraintStart_toStartOf="@+id/textView2_TampilMerek"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_Model"
        app:layout_constraintVertical_bias="0.0" />

    <TextView
        android:id="@+id/textView4_TampilTahun"
        android:layout_width="wrap_content"
        android:layout_height="30dp"
        android:layout_marginEnd="8dp"
        android:text="@string/tahuncar"
        android:textSize="24sp"

app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Tahun"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Tahun"
```

```
app:layout_constraintStart_toStartOf="@+id/textView3_TampilModel"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_Tahun" />

    <TextView
        android:id="@+id/textView5_TambilDetail"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="@string/tittlecar"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

Inti dari file ini yaitu untuk mengatur tampilan pada saat ditampilkan detail data dari suatu mobil. Pada file xml ini digunakan Constraint Layout. Terdapat text view yang digunakan untuk menampilkan nameCar, merkCar, modelCar, dan juga yearCar.

3. DetailPresenter

```
package com.example.mypc.aplikasimodul2kel37.ui.detail;

import android.util.Log;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import
com.example.mypc.aplikasimodul2kel37.data.network.RetrofitClient;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class DetailPresenter {
    private final DetailView mView ;
    public  DetailPresenter(DetailView detailView){ mView
=detailView;}

    public void getCarById(DataCar dataCar) {
        RetrofitClient.getInstance()
                .getApi()
                .GetCarById(dataCar.getId())
```

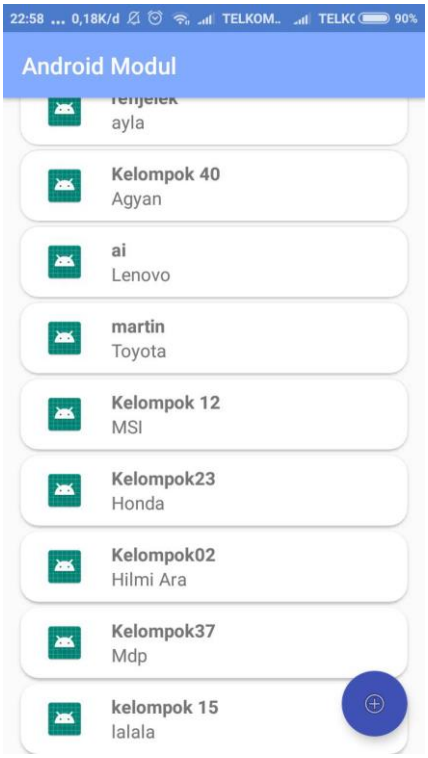
```
                .enqueue(new Callback<JsonObject>() {
                    @Override
                    public void onResponse(Call<JsonObject> call,
Response<JsonObject> response) {
                        if (response.isSuccessful()) {
                            JsonObject body = response.body();
                            JsonArray array =
body.get("result").getAsJsonArray();
                            Type type =new
TypeToken<List<DataCar>>(){}.getType();
                            List<DataCar> dataCar = new
Gson().fromJson(array, type);
                            mView.showSuccessCarById(dataCar);
                        } else {
                            mView.showErrorCarById("Terdapat
kesalahan, mohon coba kembali");
                        }
                    }
                    @Override
                    public void onFailure(Call<JsonObject> call,
Throwable t) {
                        Log.d("DATA", t.getMessage());
                        mView.showErrorCarById(t.getMessage());
                    }
                });
    }
}
```

DetailPresenter digunakan untuk mengambil data getCarById yang diambil dari DataCar. Data yang diambil merupakan data yang sesuai dengan id mobil tersebut yag dipilih. Jika berhasil maka akan ditampilkan data mobil yang dipilih, namun jika gagal maka akan muncul “Terdapat kesalahan, mohon coba kembali”.

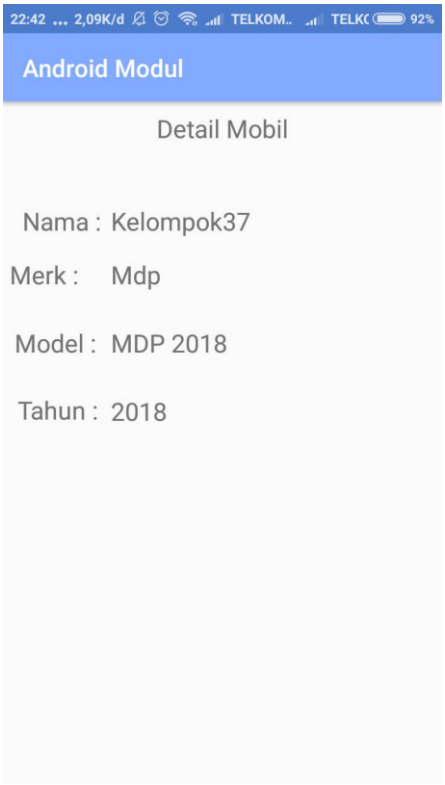
4. DetailView

```
package com.example.mypc.aplikasimodul2kel37.ui.detail;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import java.util.List;

public interface DetailView {
    void showErrorCarById(String message);

    void showSuccessCarById(List<DataCar> dataCar);
}
```

Pada DetailView terdapat method-method yang akan digunakan pada DetailActivity dan juga DetailPresenter. Berikut adalah hasil tangkapan layar tugas ini,



Gambar 4.43. Tampilan Utama



Gambar 4.44. Tampilan Detail Car

4.7 Kesimpulan

1. Ketika membuat suatu activity baru maka otomatis akan terbuat file .xml yang berfungsi untuk mengatur tampilan halaman tersebut.
2. View hanya berisi inisialisasi method yang digunakan pada activity dan presenter.
3. Dalam menjalankan aplikasi, perangkat android harus terhubung dengan koneksi internet karena aplikasi pada praktikum ini dibuat dengan terhubung dengan database online.
4. Presenter digunakan untuk mengambil data dari luar seperti retrofit, API, dan lain sebagainya.
5. Import class digunakan agar suatu class lain dapat digunakan pada class tersebut yang diinginkan.
6. Intent digunakan untuk pindah Activity.
7. Toast digunakan untuk memberikan pemberitahuan berupa Pop Up.
8. Untuk mempermudah pengaturan dan pengelompokan suatu activity dapat dibuat suatu package.