



UNIVERSITAS
BUDI LUHUR

ANDROID API (Application Programming Interface) CRUD MYSQL PHP

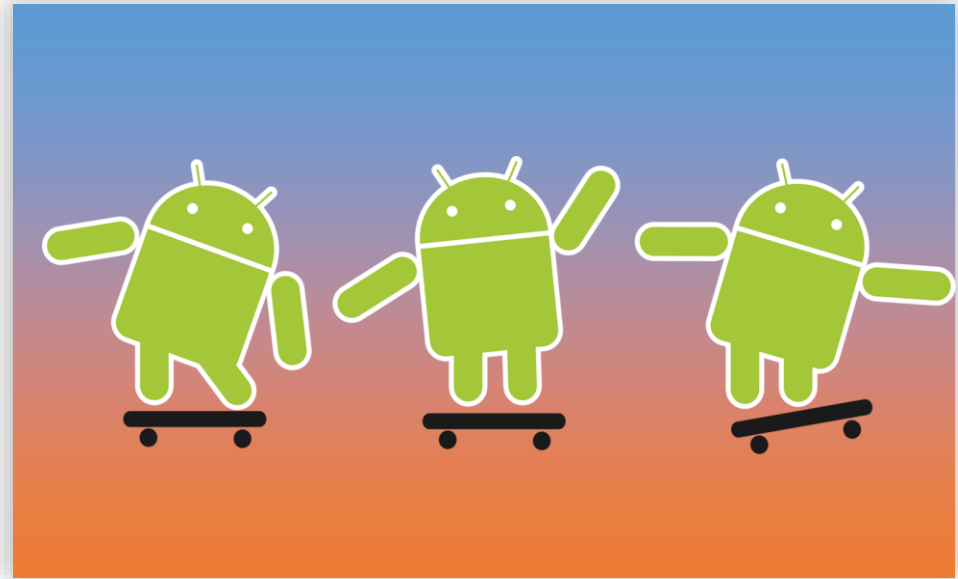
Presented By Putri Hayati, S.ST, M.Kom

[Read More](#)



Kategori

- 1. API (Application Programming Interface)
- 1. Praktikum dan Tugas





API (Application Programming Interface)

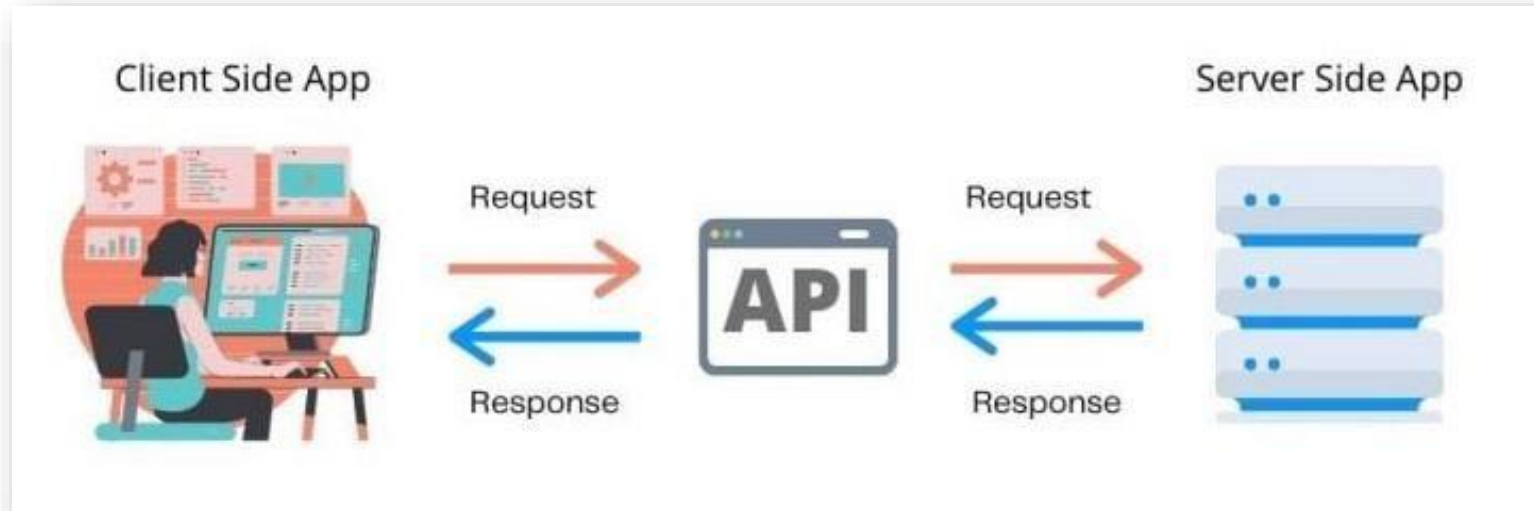


Application Programming Interface



Application Programming Interface atau **API** adalah sebuah antarmuka yang digunakan untuk menghubungkan antara satu aplikasi dengan aplikasi yang lain. Peran dari **API** adalah untuk sebagai **perantara** yang menghubungkan aplikasi berbeda, baik dari platform yang sama maupun lintas platform.

Cara Kerja - Application Programming Interface



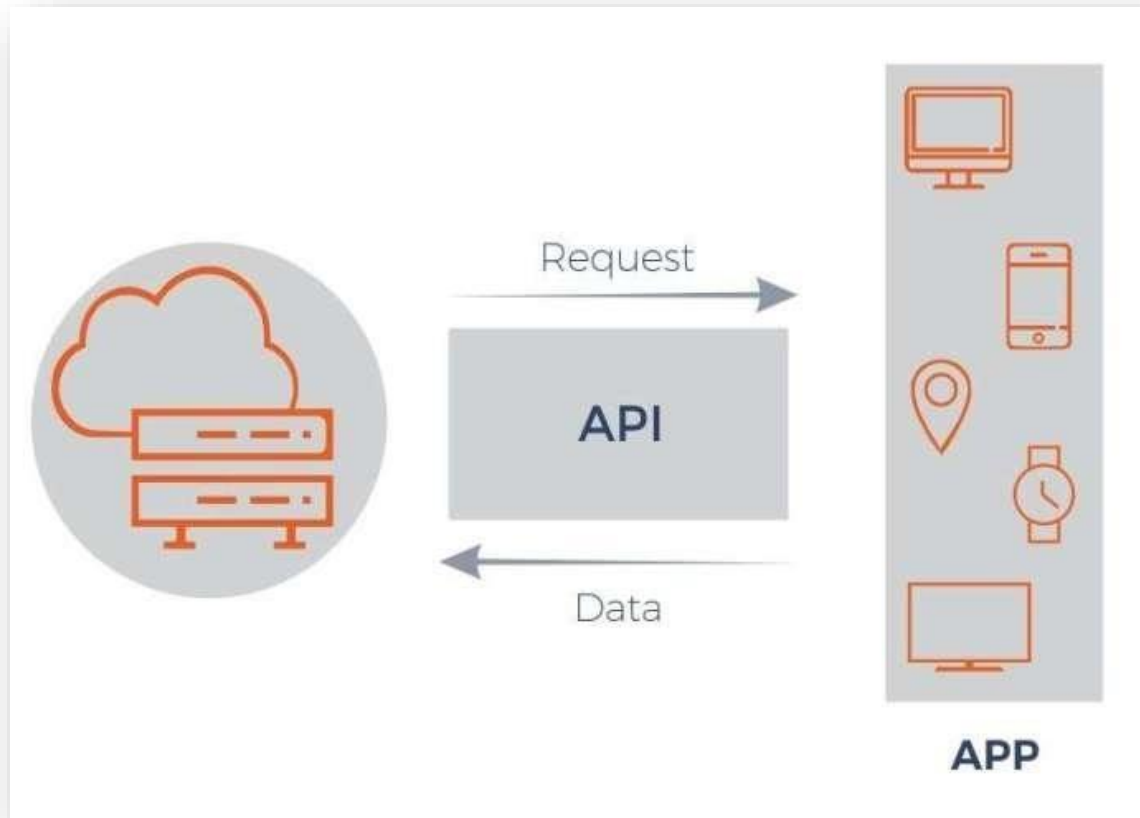
- **API** berkomunikasi melalui serangkaian aturan yang menentukan bagaimana komputer, aplikasi atau mesin dapat berbicara satu sama lain. **Web API** bertindak sebagai perantara antara dua mesin yang ingin terhubung satu sama lain untuk tugas tertentu.
- **Contoh** yang disederhanakan adalah ketika Anda masuk ke **Facebook** dari ponsel, Anda memberi tahu aplikasi Facebook bahwa Anda ingin mengakses akun Anda. Aplikasi seluler melakukan panggilan ke **Web API** untuk mengambil akun Facebook dan data Anda. Facebook kemudian akan mengakses informasi ini dari salah satu servernya dan mengembalikan data ke aplikasi seluler.

Cara Kerja - Application Programming Interface

Saat ini, kehadiran API telah berkembang menyesuaikan kebutuhan bisnis dan programmer. Eksistensi beberapa jenis **API** terdiri atas **public**, **privat**, **partner**, dan **composite** dengan penjelasan sebagai berikut :

- **Publik (public):** Sesuai dengan namanya, API yang bersifat publik ini bebas digunakan dan biasanya mudah diakses. Jenis Application Programming Interface yang sering disebut sebagai Open API ini tersedia untuk berbagai kebutuhan umum. Contohnya adalah Google Maps untuk pengelolaan lokasi serta Google Account untuk autentikasi pengguna.
- **Privat (private):** Berbeda dengan jenis publik, Application Programming Interface jenis privat tidak dapat diakses oleh siapa saja. Pengaturan aksesnya hanya berjalan secara tim atau organisasi internal. Adapun contoh penggunaannya adalah pengembangan tampilan (front-end) berupa daftar harga produk yang diambil dari data perusahaan (back-end).
- **Partner: Application Programming Interface** jenis ini berarti membatasi penggunaannya terhadap partner atau mitra yang sudah terikat dengan kerja sama tertentu.
- **Composite: Application Programming Interface** jenis ini berarti memuat atau memiliki “komposisi” berupa data terintegrasi antar server atau hosting yang berbeda.

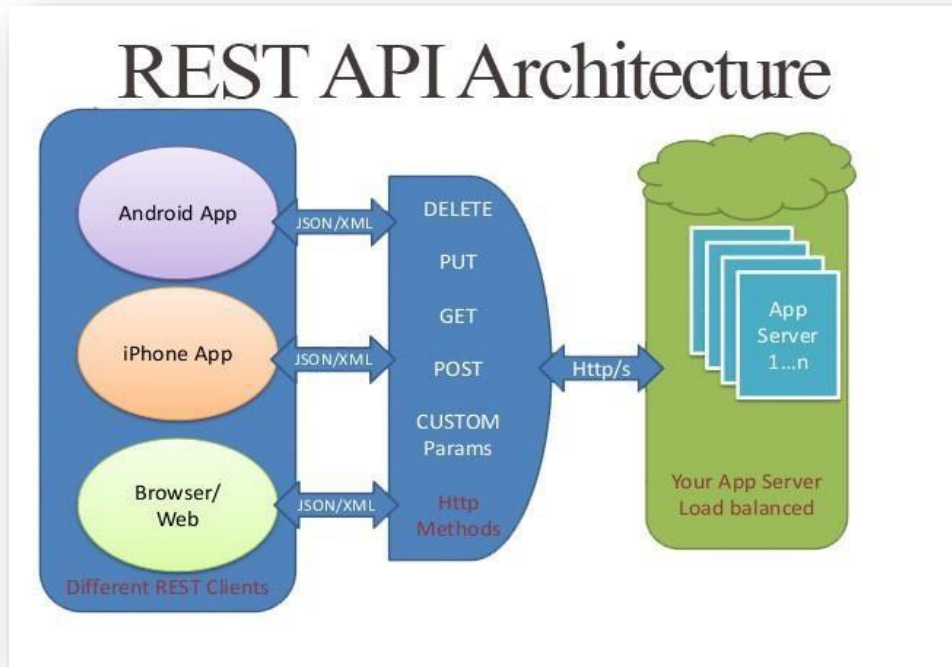
Implementasi - Application Programming Interface



- **Implementasi API** dapat berupa fungsi, **method**, atau **URL endpoint**. Kumpulan atau seperangkat antarmuka ini membentuk definisi dan protokol yang mudah untuk diintegrasikan.
- Hal ini memungkinkan Anda untuk tetap memanfaatkan layanan atau fitur bisnis produk lain tanpa mengetahui detail cara kerjanya.
- Bentuk pemanfaatan atau “kerja sama” ini tentu akan menyederhanakan proses administrasi, desain, bahkan merealisasikan peluang inovasi yang lebih besar

Jenis Web - Application Programming Interface

REST (RESTful) API

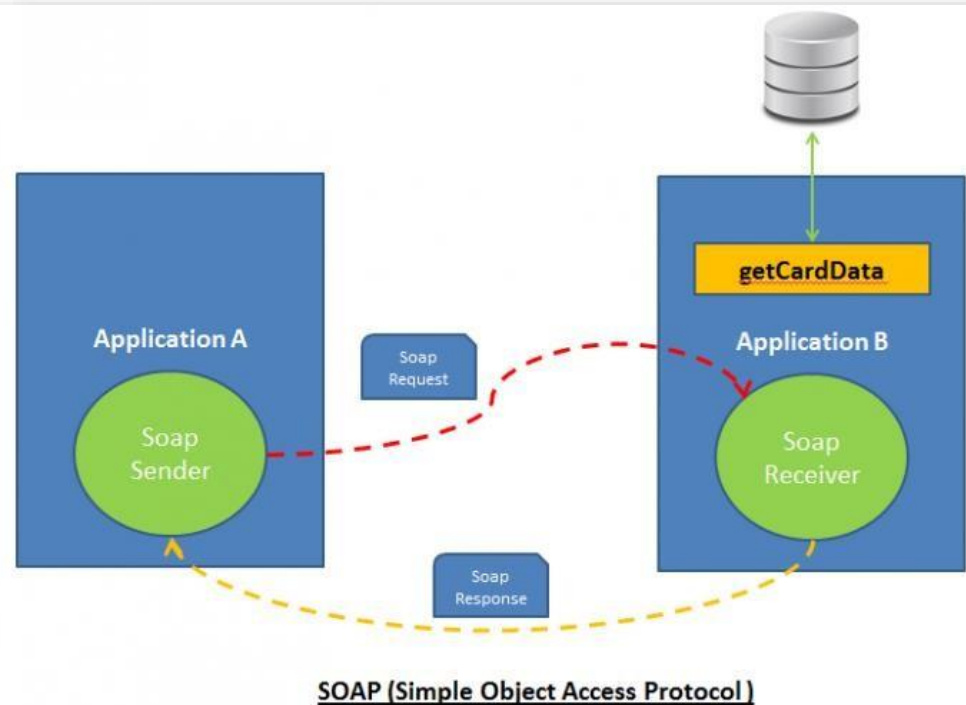


API ini memberikan akses ke data dengan menggunakan serangkaian operasi yang seragam dan standar. API REST didasarkan pada URL dan protokol HTTP dan didasarkan pada 6 kendala arsitektur ini.

- **Client-server based (Berbasis klien-server)** - klien menangani proses ujung depan sementara server menangani backend dan keduanya dapat diganti secara independen satu sama lain.
- **Uniform interface** (Seragam antarmuka) - mendefinisikan antarmuka antara klien dan server dan menyederhanakan arsitektur untuk memungkinkan setiap bagian berkembang secara terpisah
- **Stateless** – setiap permintaan dari klien ke server harus independen dan berisi semua informasi yang diperlukan sehingga server dapat memahami dan memprosesnya sesuai kebutuhan.
- **Cacheable** - memelihara tanggapan dalam cache antara klien dan server menghindari pemrosesan tambahan
- **Layered system (Sistem berlapis)** - lapisan disusun secara hierarkis sehingga masing-masing hanya dapat 'melihat' lapisan yang sesuai dengan yang berinteraksi dengannya.
- **Code-on-demand** - memungkinkan fungsionalitas klien diperluas dengan mengunduh dan mengeksekusi kode dalam bentuk applet dan skrip.

Jenis Web - Application Programming Interface

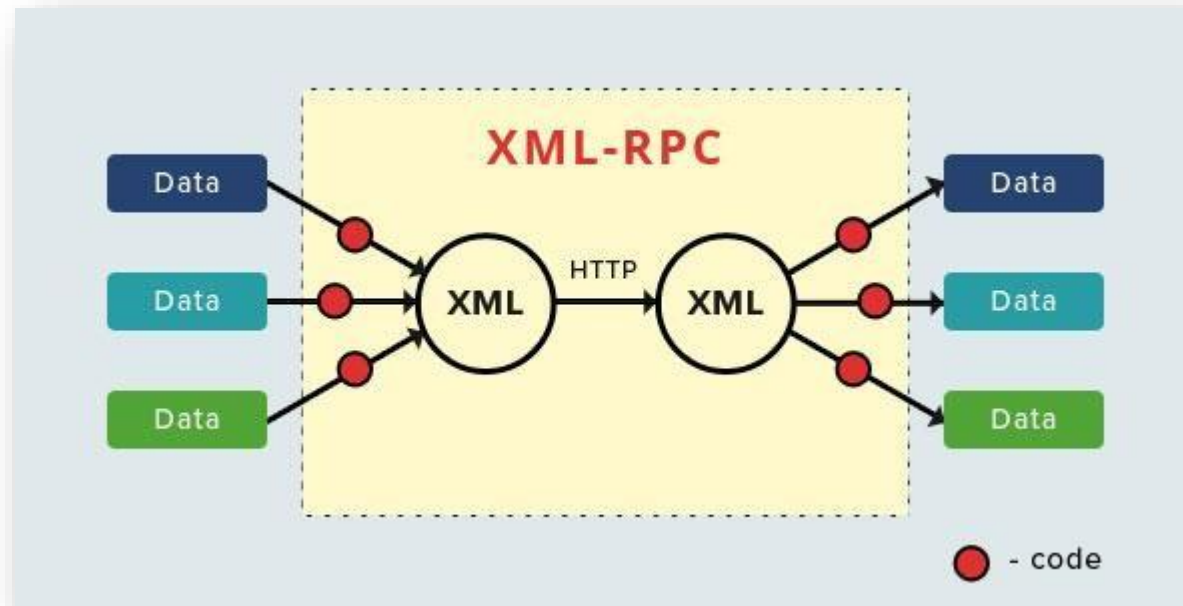
SOAP (Simple Object Access Protocol)



- **Simple Object Access Protocol** sedikit lebih rumit daripada **REST** karena memerlukan lebih banyak informasi tentang cara mengirim pesannya. API ini telah ada sejak akhir 1990-an dan menggunakan XML untuk mentransfer data. Ini membutuhkan aturan ketat dan keamanan tingkat lanjut yang membutuhkan lebih banyak bandwidth.
- Protokol ini tidak memiliki kemampuan untuk melakukan cache, memiliki komunikasi yang ketat dan membutuhkan setiap informasi tentang interaksi sebelum panggilan apa pun dianggap diproses.

Jenis Web - Application Programming Interface

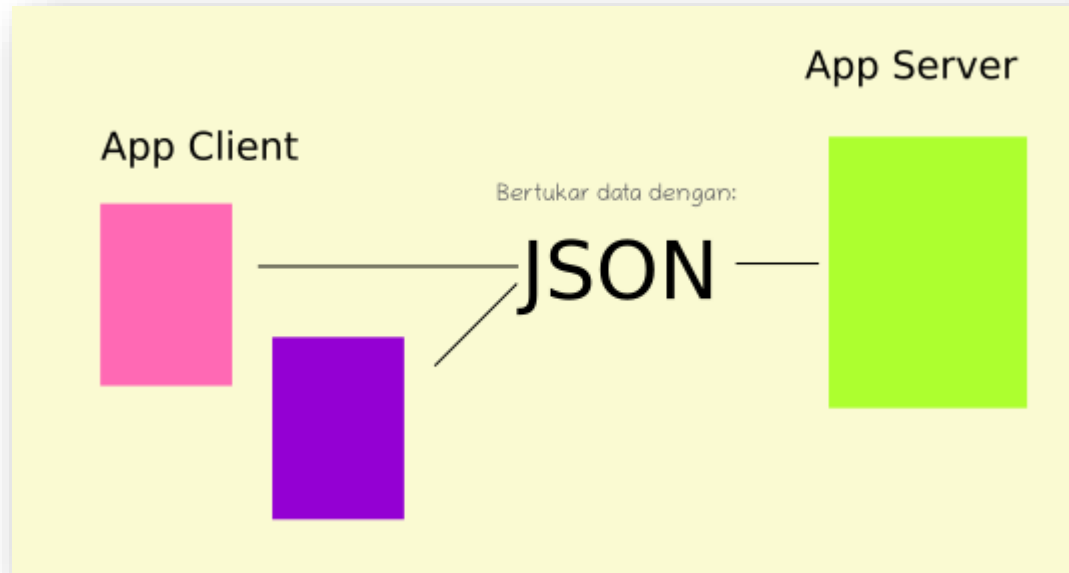
XML-RPC (eXtensible Markup Language -Remote Procedure Call)



- Dikenal sebagai bahasa markup yang dapat dikembangkan - Panggilan Prosedur Jarak Jauh. Protokol ini menggunakan format XML tertentu untuk mentransfer data dan lebih tua dan lebih sederhana dari SOAP. Klien melakukan RPC dengan mengirim permintaan HTTP ke server yang mengimplementasikan XML-RPC dan menerima respons HTTP.

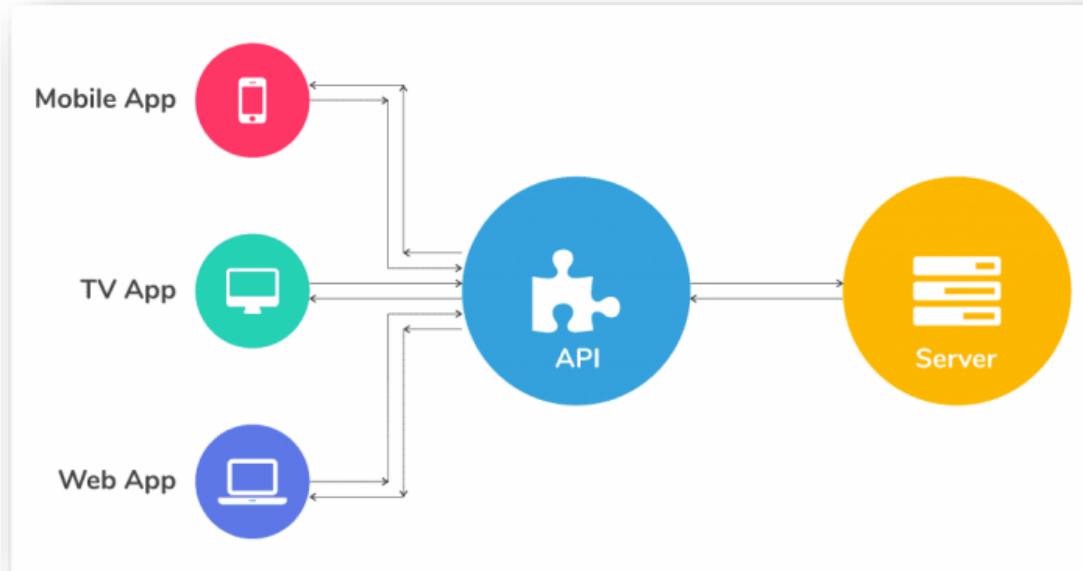
Jenis Web - Application Programming Interface

JSON-RPC



- Mirip dengan XML-RPC karena keduanya bekerja dengan cara yang sama kecuali protokol ini menggunakan JSON dan bukan format XML. Klien biasanya perangkat lunak yang memanggil metode tunggal dari sistem jarak jauh.

Contoh dan Komponen - Application Programming Interface

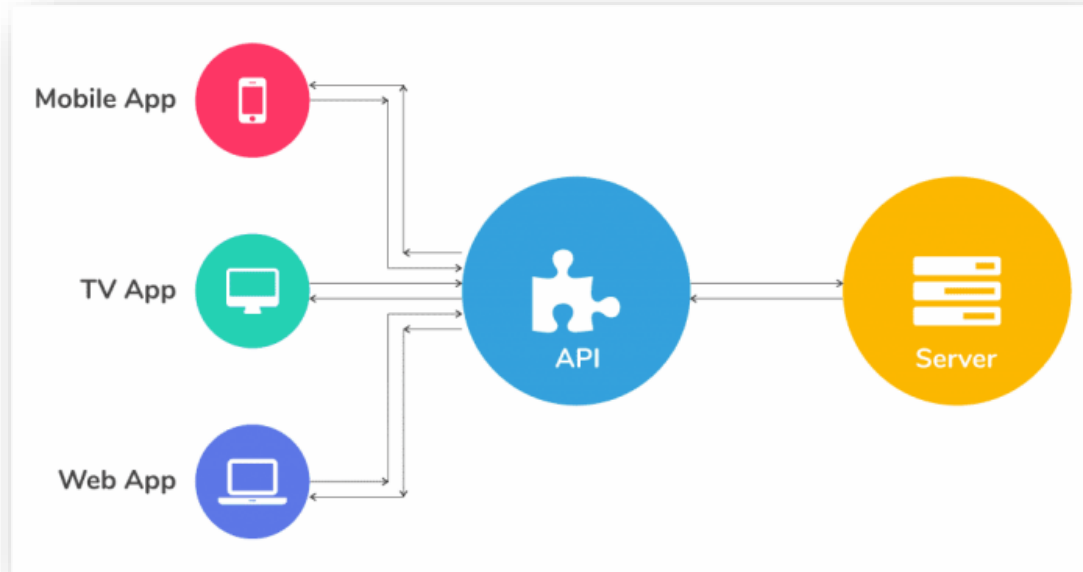


Tipe Method

Pengelolaan Application Programming Interface terkait erat dengan berbagai tipe method dalam request sebagai berikut. :

- **GET: Method** ini digunakan untuk menerima informasi saja, tidak termasuk melakukan perubahan apapun. Biasanya, method ini menandai operasi display atau menampilkan data.
- **POST: Method** ini digunakan untuk menambah suatu informasi. Hal ini terkait erat dengan operasi Add atau Create untuk suatu entitas.
- **PUT: Method** ini digunakan untuk mengubah suatu informasi yang biasa terkait erat dengan operasi Edit.
- **DELETE: Method** ini digunakan untuk menghapus suatu data tertentu yang biasa terkait erat dengan operasi Delete.

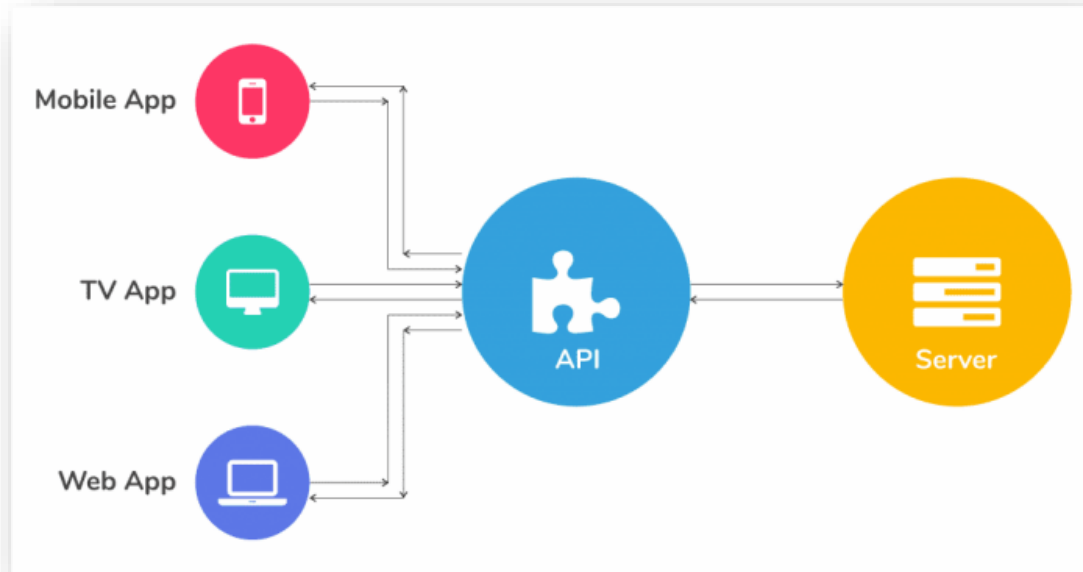
Contoh dan Komponen - Application Programming Interface



Endpoint

- **Endpoint** adalah **routing** atau tautan terstruktur dalam implementasi **Application Programming Interface**. Komponen ini terdiri atas dua aspek yaitu URL dan path. Adapun URL ialah link yang dapat diakses atas suatu platform sementara path adalah detail akses terhadap suatu informasi. Contoh URL adalah `https://contohURL.com` sementara path terkait misalnya `https://contohURL.com/produk` yang menambahkan detail berupa `/produk` untuk mengakses informasi terkait produk.

Contoh dan Komponen - Application Programming Interface



Data dan Header

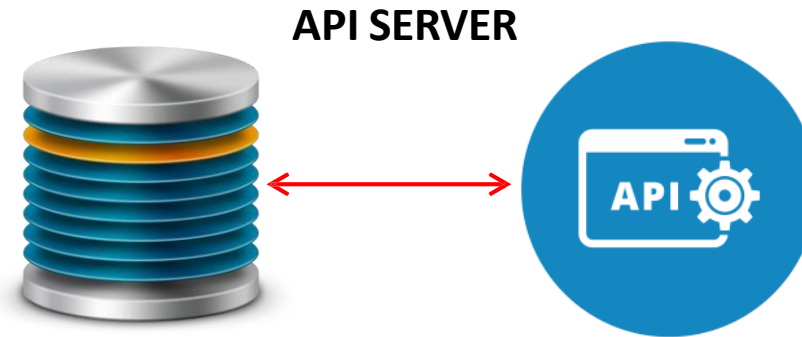
- Header merupakan komponen yang memuat informasi kepada klien dan server. Informasi khusus yang biasa dicantumkan pada header adalah “Content Type” atau jenis konten, misalnya token autentikasi. Sementara data atau body adalah informasi spesifik yang disampaikan. Saat ini, data ditampilkan dalam format JSON (JavaScript Object Notation).



Praktikum dan Tugas



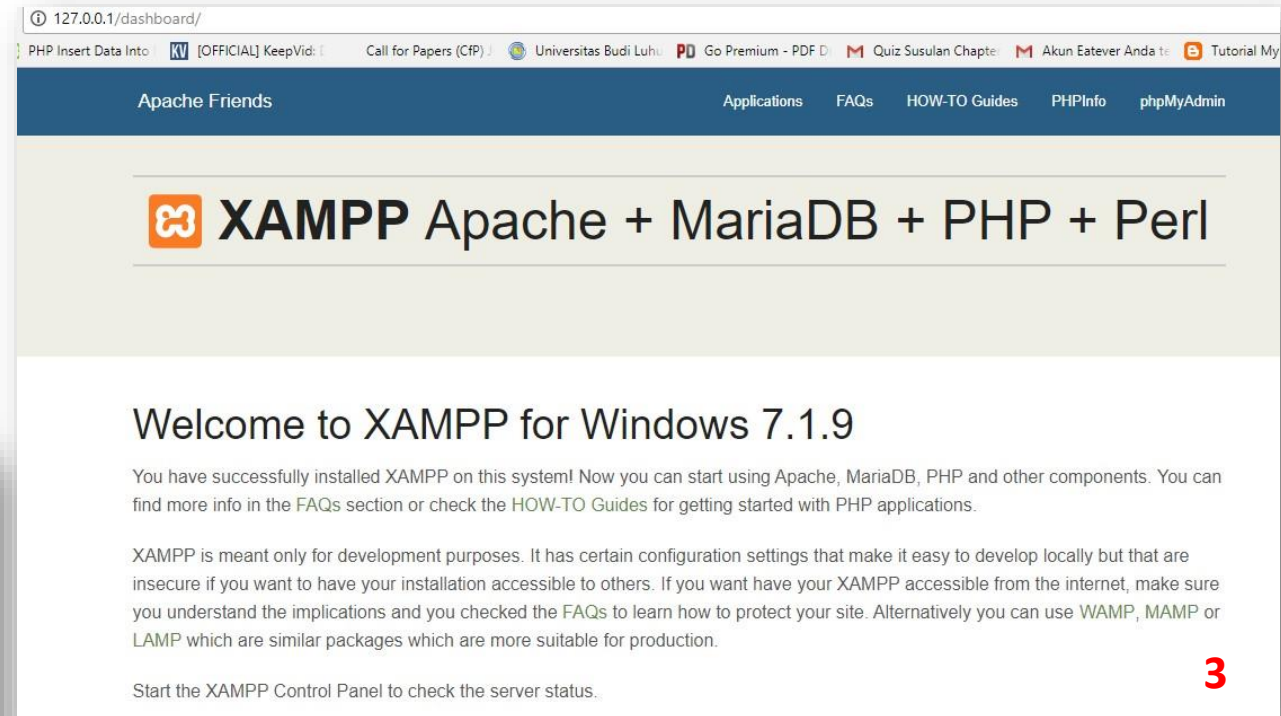
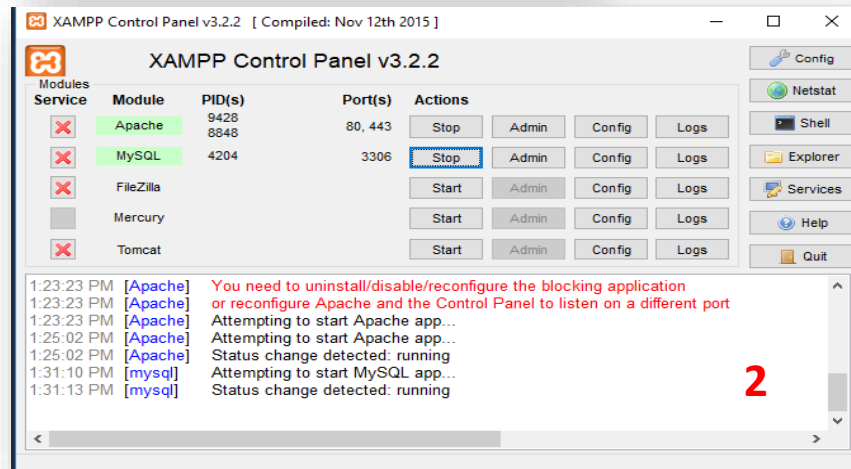
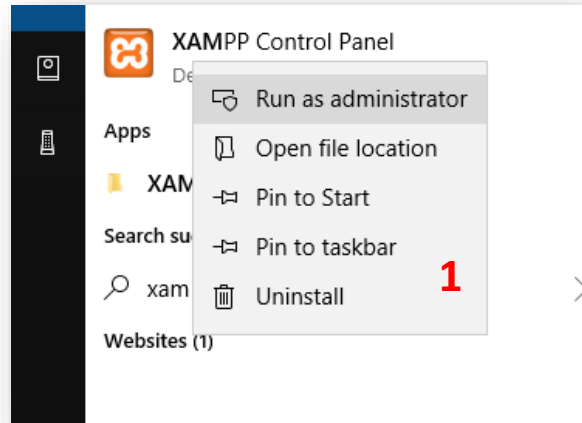
Tahap Awal :



Spesifikasi

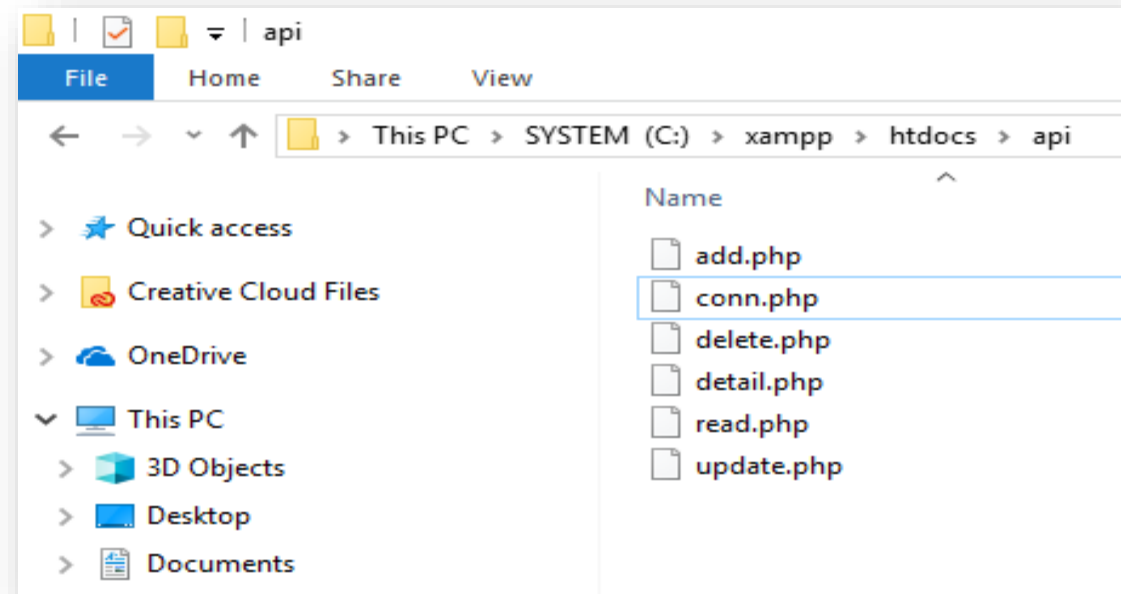
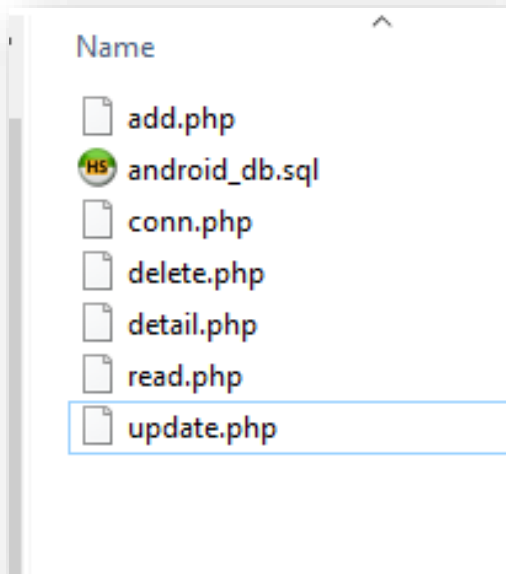


SETUP API SERVER MENGGUNAKAN XAMPP :



- **Copy Source Code ke Apache Web Directory : C:\xampp\htdocs -**

- Note : Biar rapi, buat directory khusus di Web Directory, dalam hal ini saya membuat folder “api” di Web Directory.



- Edit Database Connection di file conn.php

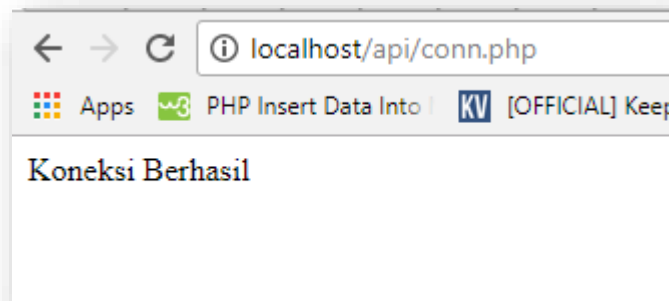
```
1  <?php
2
3  define('HOST', 'localhost');
4  define('USER', 'putri');
5  define('PASS', 'putri123');
6  define('DB', 'android_api');
7
8  $con = mysqli_connect(HOST, USER, PASS, DB);
9
10 if ($con->connect_error) {
11     die("Koneksi Gagal: " . $con->connect_error);
12 }
13 echo "Koneksi Berhasil";
14 ?>
15
```

- Struktur table di database “android_api” :

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(16)			No	None		AUTO_INCREMENT	Change Drop Primary Unique Index More
<input type="checkbox"/> 2	nama	varchar(20)	latin1_swedish_ci		No	None			Change Drop Primary Unique Index More
<input type="checkbox"/> 3	alamat	varchar(100)	latin1_swedish_ci		No	None			Change Drop Primary Unique Index More
<input type="checkbox"/> 4	jurusan	varchar(100)	latin1_swedish_ci		No	None			Change Drop Primary Unique Index More

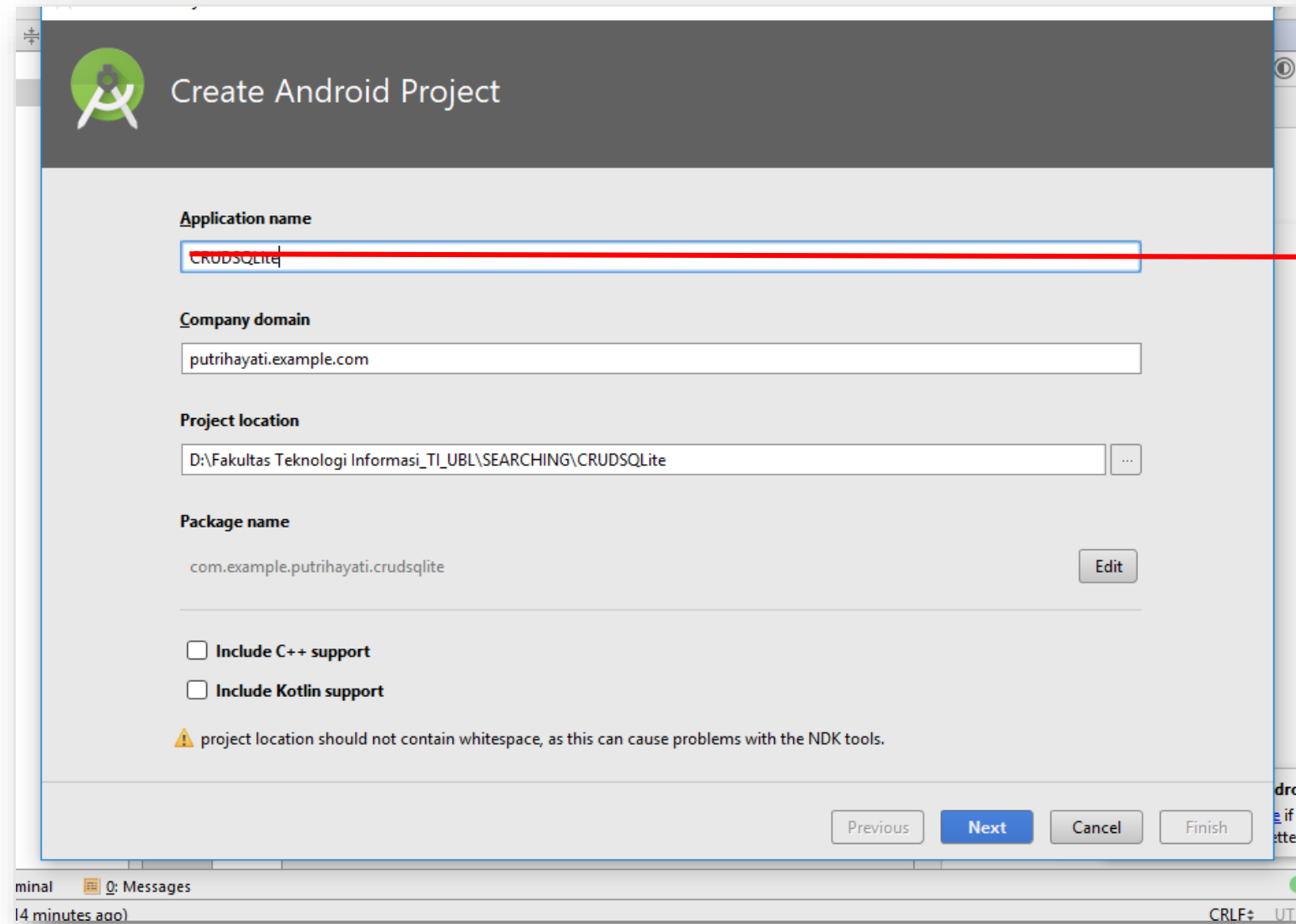
☐ Check all With selected: Browse Change Drop Primary Unique Index Add to central columns
 Remove from central columns

- Testing Koneksi ke Database :



APLIKASI API CRUD

1. Pertama - tama silahkan kalian buat sebuah project baru di **Android Studio** atau IDE
2. Project name : **Nim_Namadepan_APICRUD**
3. Pilih **'Empty Activity'**
4. Ikuti langkah pada slide berikutnya
5. Jika sudah berhasil lalu di screenrecord dikirim ke Group Telegram, dengan subject : **Nim_Nama_APICRUD_Kelompok**
6. Hasil untuk praktikum di kirim ke group telegram paling lambat tanggal **22 Desember 2022 –Pukul 23.00 WIB**



Create Android Project

Application name
CRUDSQLite

Company domain
putrihayati.example.com

Project location
D:\Fakultas Teknologi Informasi_TI_UBL\SEARCHING\CRUDSQLite

Package name
com.example.putrihayati.crudsqlite

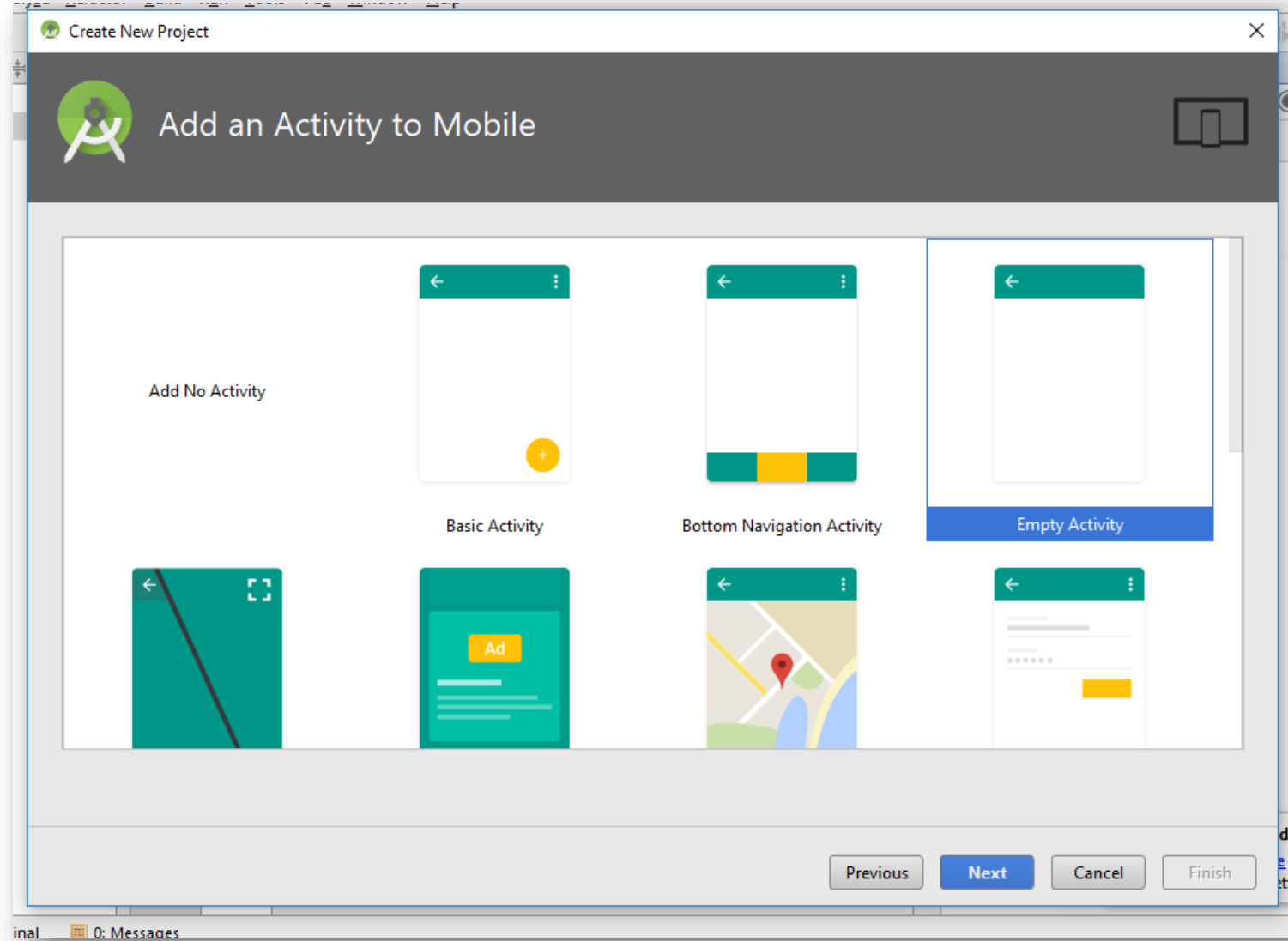
☐ Include C++ support

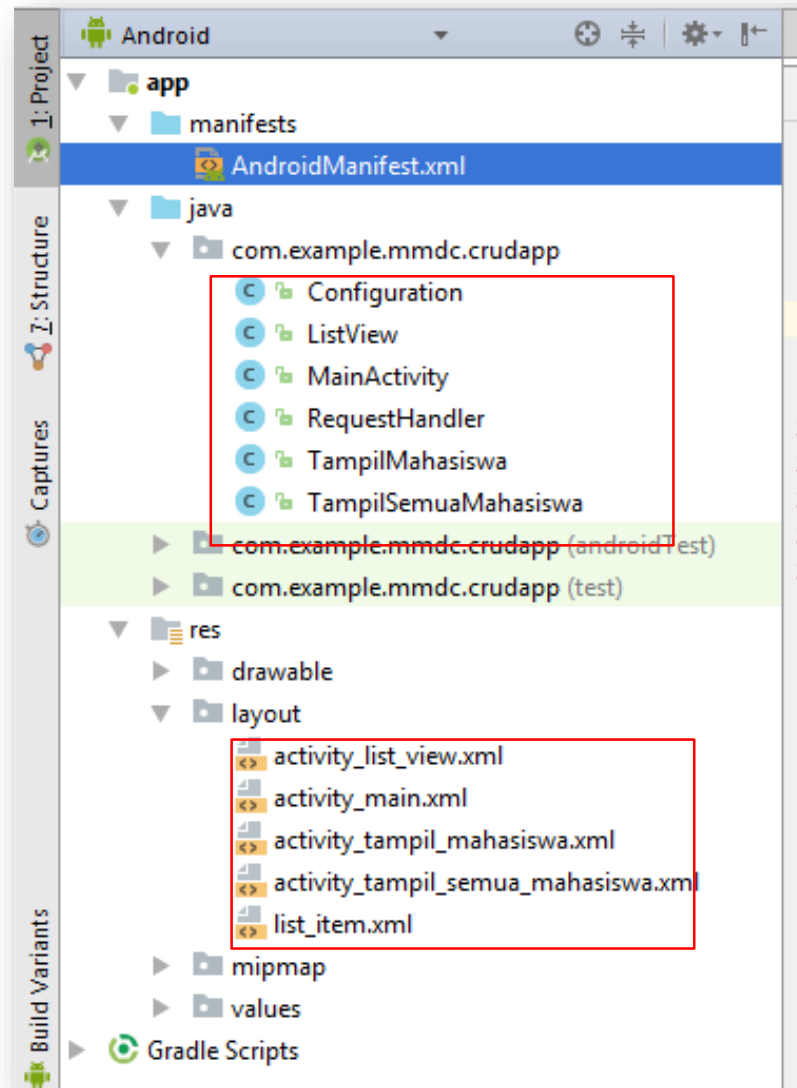
☐ Include Kotlin support

⚠ project location should not contain whitespace, as this can cause problems with the NDK tools.

Previous Next Cancel Finish

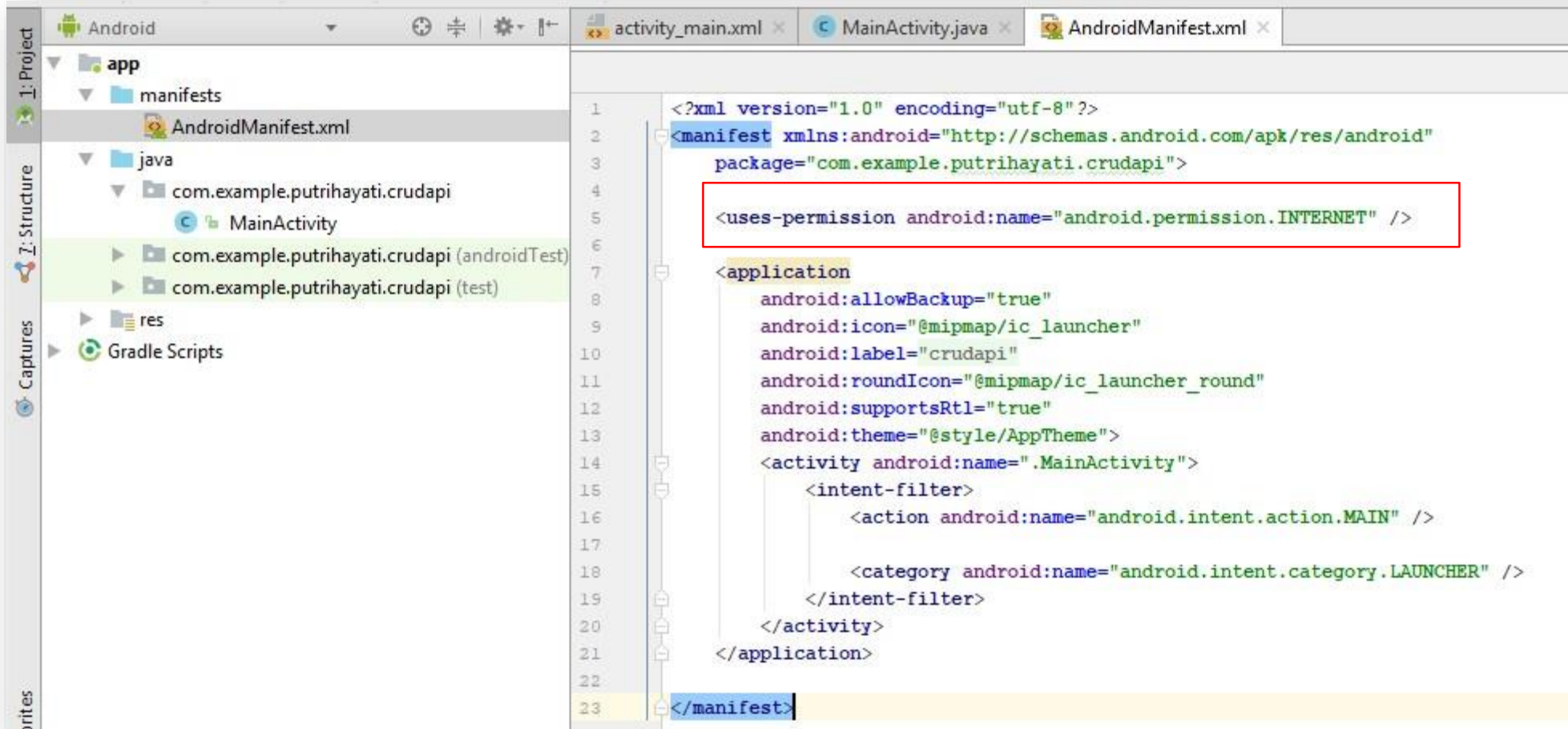
WARNING !!
Nim_Namadepan_
APICRUD



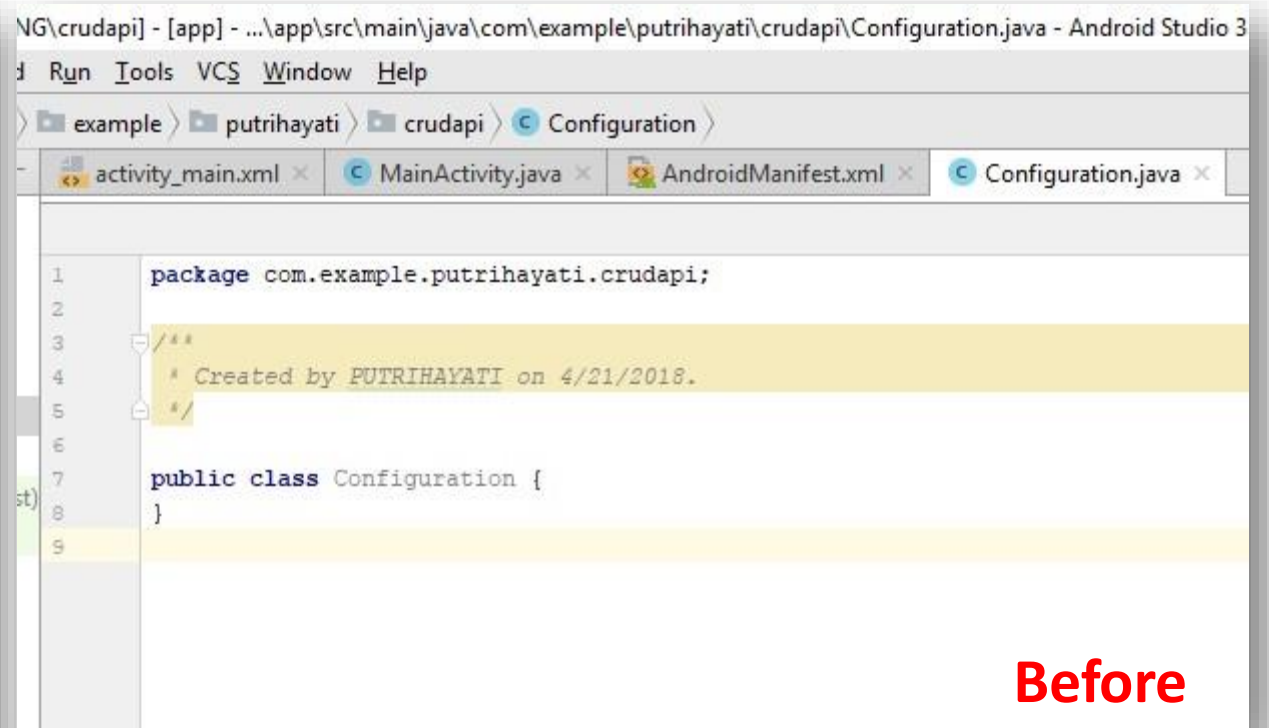
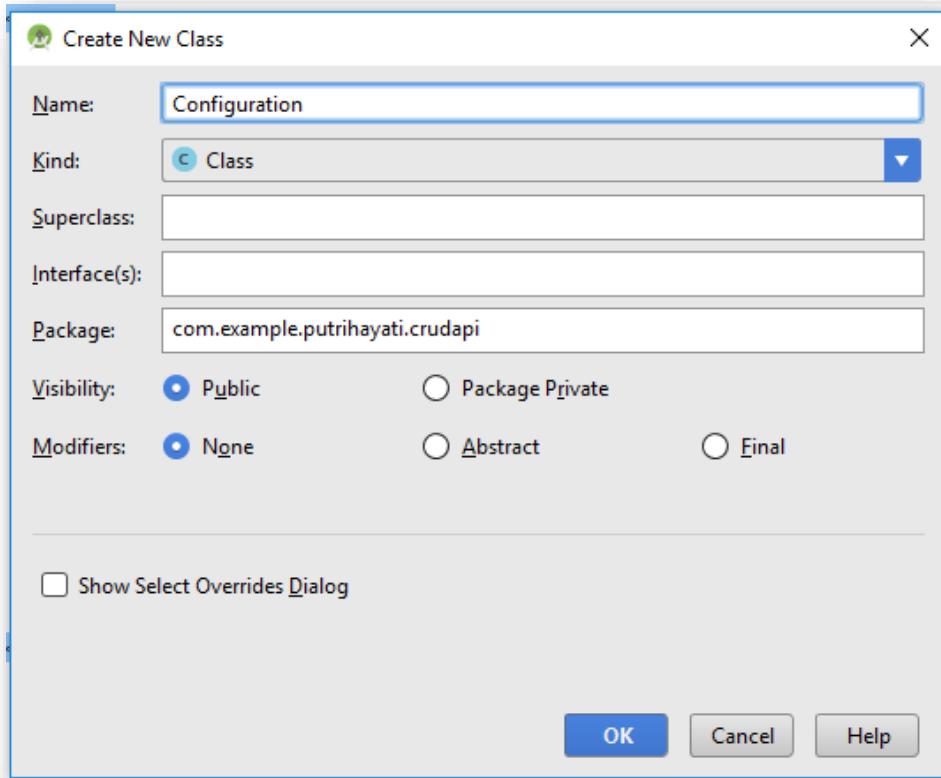


Pilih "AndroidManifest"

- ketik source code seperti pada gambar dibawah ini:



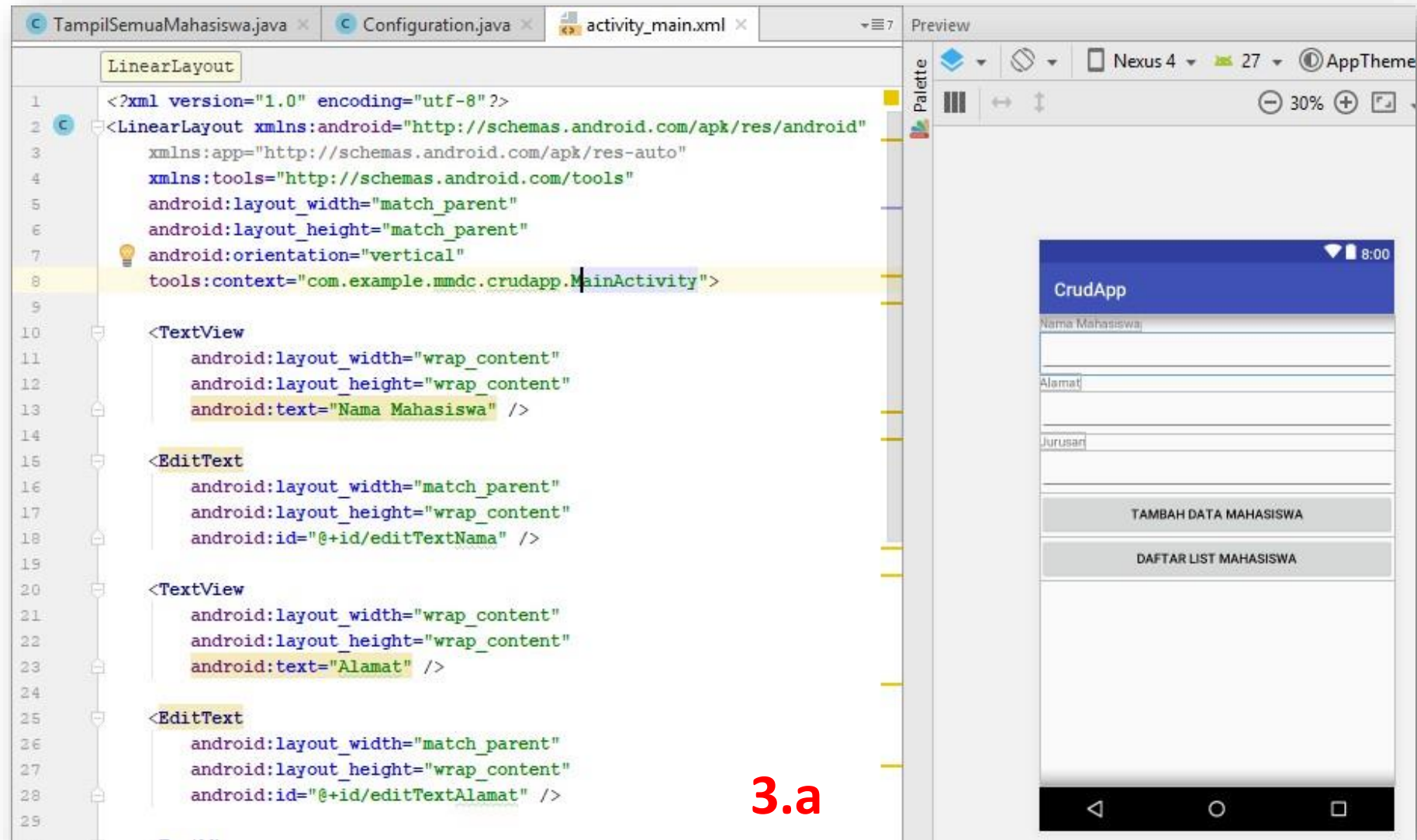
Create -Java Class (untuk konfigurasi ke DB)
seperti pada gambar dibawah ini :



Before

- Koneksi ke database, silahkan menggunakan IP Address & PATH sesuai dengan laptop masing-masing !!

```
Configuration
1 package com.example.putrihayati.crudapi;
2
3 /**
4  * Created by PUTRIHAYATI on 4/21/2018.
5  */
6
7 public class Configuration {
8     public static final String URL_ADD="http://localhost/api/add.php";
9     public static final String URL_GET_MHS = "http://localhost/api/detail.php?id=";
10    public static final String URL_GET_ALL = "http://localhost/api/read.php";
11    public static final String URL_UPDATE_MHS = "http://localhost/api/update.php";
12    public static final String URL_DELETE_MHS = "http://localhost/api/delete.php?id=";
13
14    //Request to PHP
15    public static final String KEY_MHS_ID = "id";
16    public static final String KEY_MHS_NAMA = "nama";
17    public static final String KEY_MHS_ALAMAT = "alamat";
18    public static final String KEY_MHS_JURUSAN = "jurusan";
19
20    //JSON Tags
21    public static final String TAG_JSON_ARRAY="result";
22    public static final String TAG_ID = "id";
23    public static final String TAG_NAMA = "nama";
24    public static final String TAG_ALAMAT = "alamat";
25    public static final String TAG_JURUSAN = "jurusan";
26
27    //ID Mahasiswa
28    public static final String MHS_ID = "mhs_id";
29
30 }
```



3.a

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Jurusan" />

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/editTextJurusan" />

<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Tambah Data Mahasiswa"
    android:id="@+id/buttonAdd" />

<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Daftar List Mahasiswa"
    android:id="@+id/buttonView" />

</LinearLayout>
```

3.b

The screenshot shows a mobile application interface titled "CrudApp". It features three input fields: "Nama Mahasiswa", "Alamat", and "Jurusan". Below these fields are two buttons: "TAMBAH DATA MAHASISWA" and "DAFTAR LIST MAHASISWA". The interface is displayed on a smartphone screen with a status bar at the top showing the time as 8:00 and a navigation bar at the bottom.



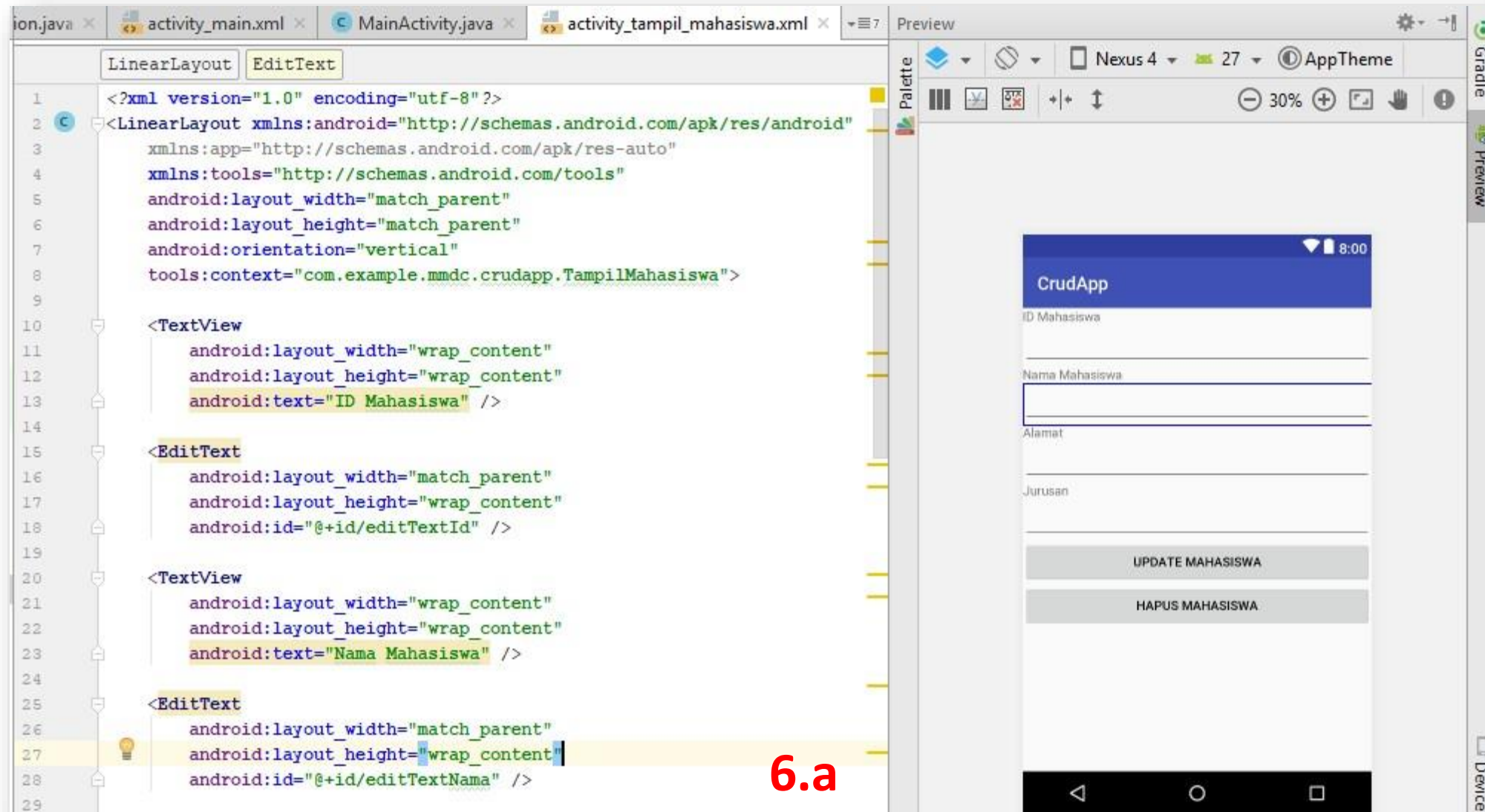


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:layout_height="match_parent"
    tools:context="com.example.mmdc.crudapp.ListView">

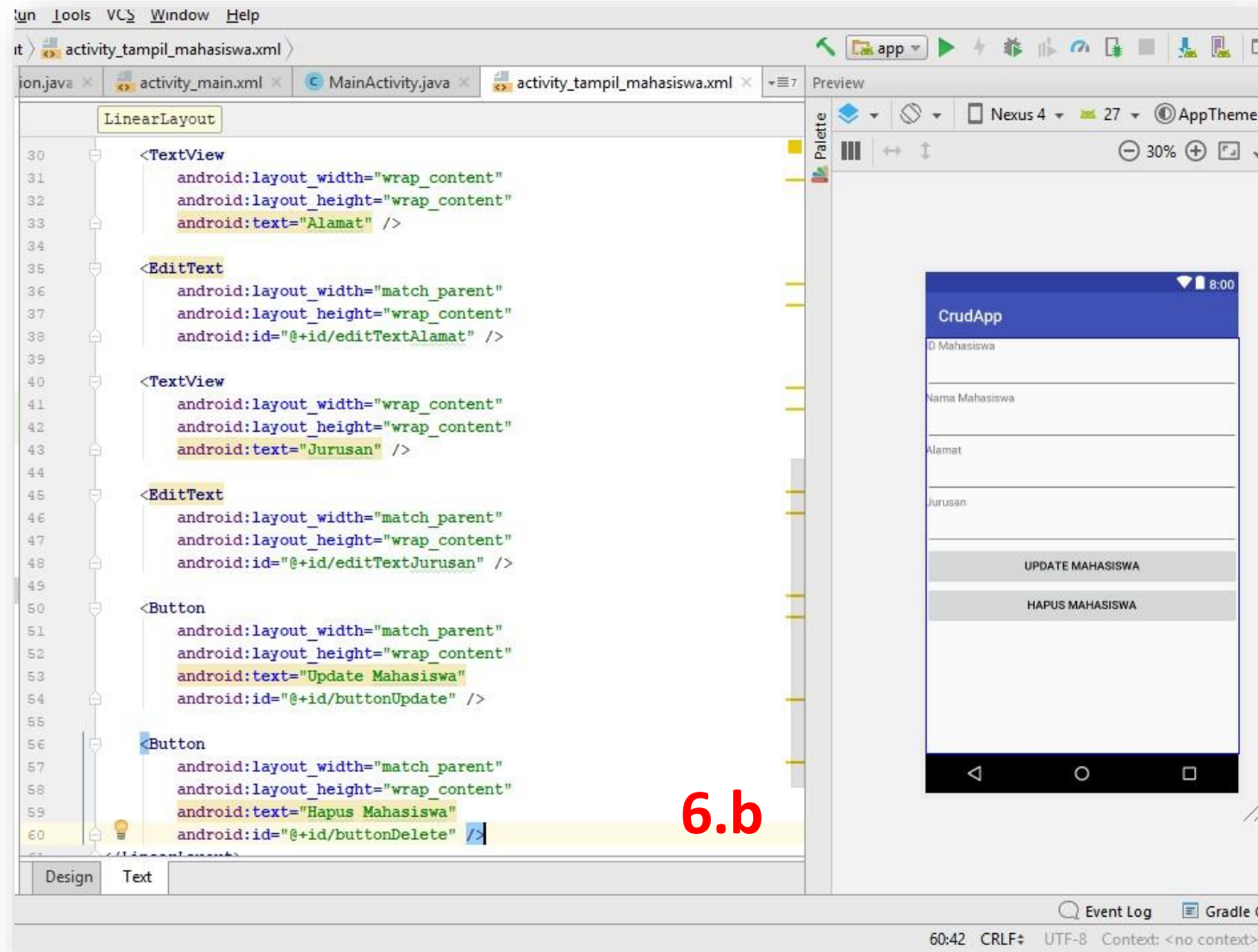
</LinearLayout>
```

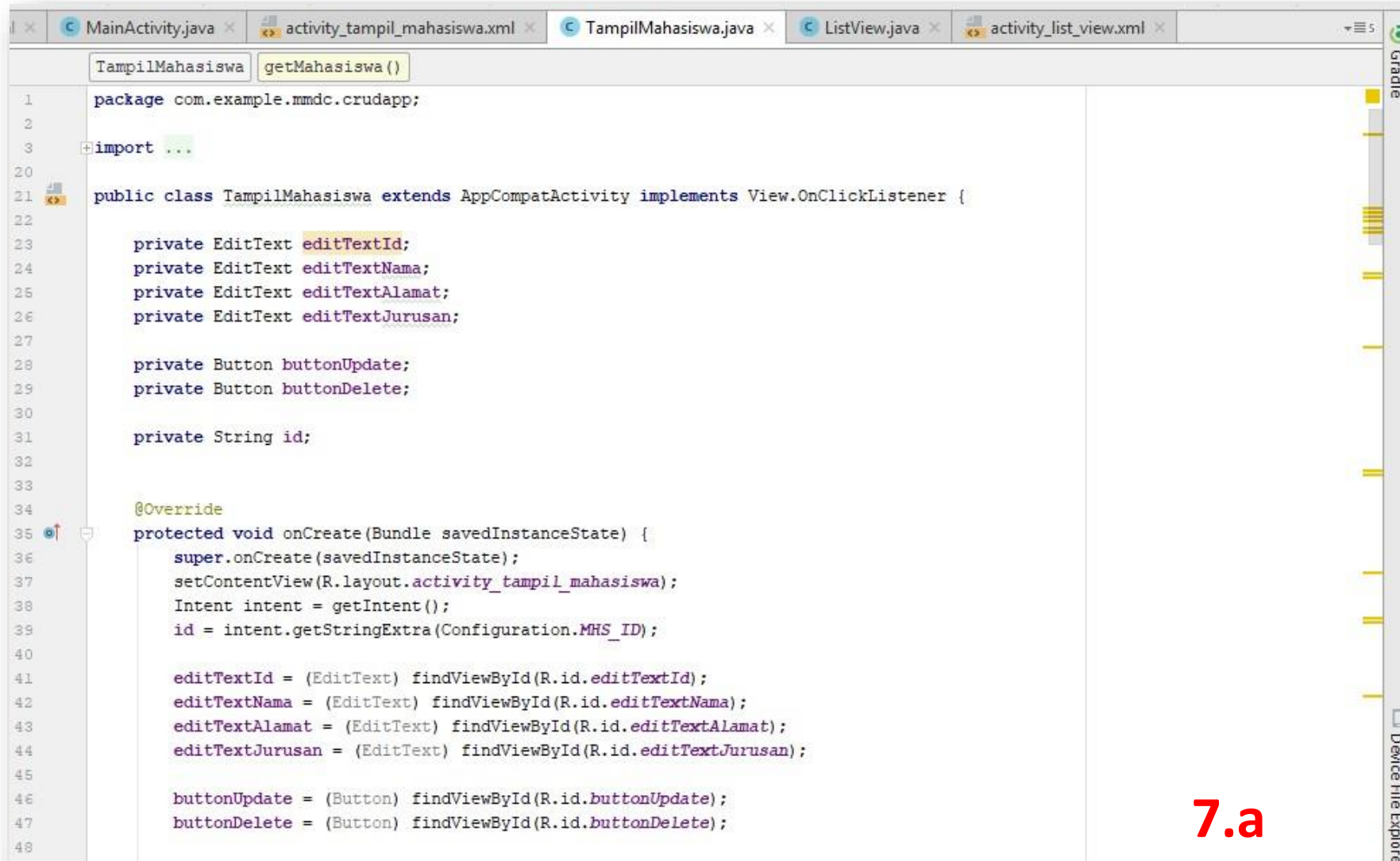
5

Source Code List_view.xml



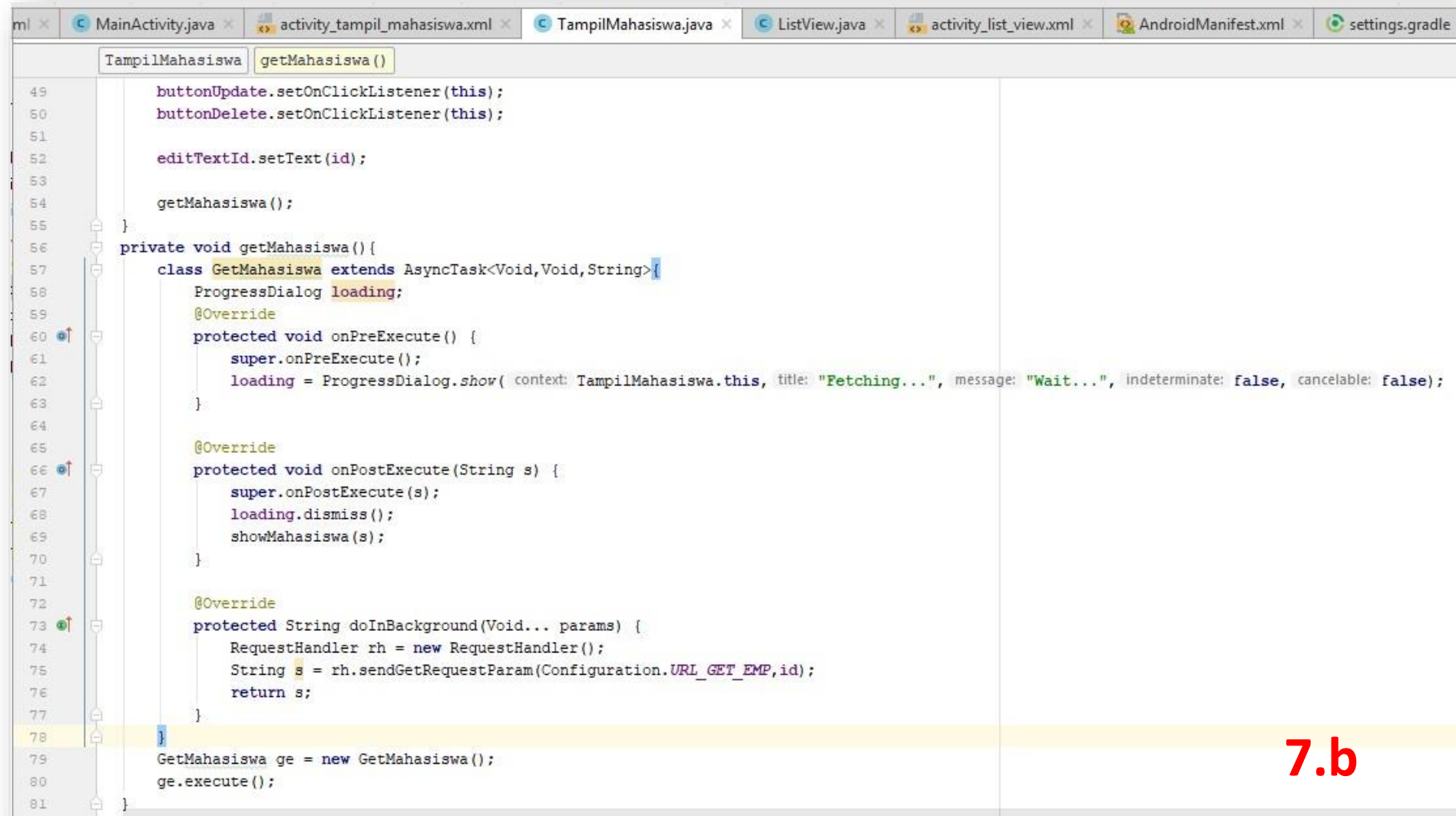
6.a





```
1 package com.example.mmdc.crudapp;
2
3 import ...
4
20
21 public class TampilMahasiswa extends AppCompatActivity implements View.OnClickListener {
22
23     private EditText editTextId;
24     private EditText editTextNama;
25     private EditText editTextAlamat;
26     private EditText editTextJurusan;
27
28     private Button buttonUpdate;
29     private Button buttonDelete;
30
31     private String id;
32
33
34     @Override
35     protected void onCreate(Bundle savedInstanceState) {
36         super.onCreate(savedInstanceState);
37         setContentView(R.layout.activity_tampil_mahasiswa);
38         Intent intent = getIntent();
39         id = intent.getStringExtra(Configuration.MHS_ID);
40
41         editTextId = (EditText) findViewById(R.id.editTextId);
42         editTextNama = (EditText) findViewById(R.id.editTextNama);
43         editTextAlamat = (EditText) findViewById(R.id.editTextAlamat);
44         editTextJurusan = (EditText) findViewById(R.id.editTextJurusan);
45
46         buttonUpdate = (Button) findViewById(R.id.buttonUpdate);
47         buttonDelete = (Button) findViewById(R.id.buttonDelete);
48
```

7.a

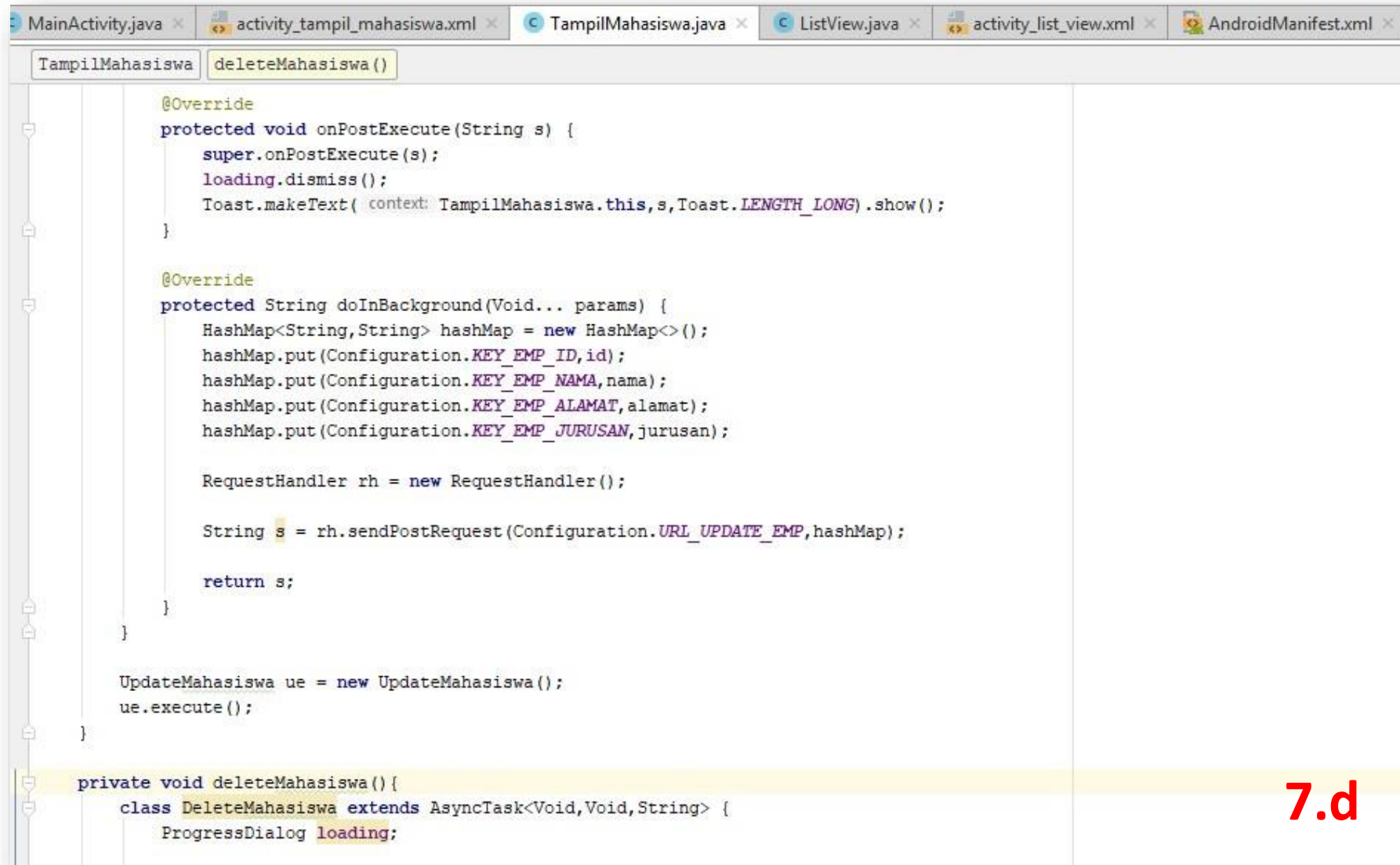


```
49         buttonUpdate.setOnClickListener(this);
50         buttonDelete.setOnClickListener(this);
51
52         editTextId.setText(id);
53
54         getMahasiswa();
55     }
56     private void getMahasiswa() {
57         class GetMahasiswa extends AsyncTask<Void, Void, String> {
58             ProgressDialog loading;
59             @Override
60             protected void onPreExecute() {
61                 super.onPreExecute();
62                 loading = ProgressDialog.show(context: TampilMahasiswa.this, title: "Fetching...", message: "Wait...", indeterminate: false, cancelable: false);
63             }
64
65             @Override
66             protected void onPostExecute(String s) {
67                 super.onPostExecute(s);
68                 loading.dismiss();
69                 showMahasiswa(s);
70             }
71
72             @Override
73             protected String doInBackground(Void... params) {
74                 RequestHandler rh = new RequestHandler();
75                 String s = rh.sendGetRequestParam(Configuration.URL_GET_EMP, id);
76                 return s;
77             }
78         }
79         GetMahasiswa ge = new GetMahasiswa();
80         ge.execute();
81     }
```

7.b

```
MainActivity.java x activity_tampil_mahasiswa.xml x TampilMahasiswa.java x ListView.java x activity_list_view.xml x AndroidManifest.xml x settings.gradle x
TampilMahasiswa getMahasiswa()
82 private void showMahasiswa(String json){
83     try {
84         JSONObject jsonObject = new JSONObject(json);
85         JSONArray result = jsonObject.getJSONArray(Configuration.TAG_JSON_ARRAY);
86         JSONObject c = result.getJSONObject( index: 0);
87         String name = c.getString(Configuration.TAG_NAMA);
88         String desg = c.getString(Configuration.TAG_ALAMAT);
89         String sal = c.getString(Configuration.TAG_JURUSAN);
90
91         editTextNama.setText(name);
92         editTextAlamat.setText(desg);
93         editTextJurusan.setText(sal);
94
95     } catch (JSONException e) {
96         e.printStackTrace();
97     }
98 }
99
100 private void updateMahasiswa(){
101     final String nama = editTextNama.getText().toString().trim();
102     final String alamat = editTextAlamat.getText().toString().trim();
103     final String jurusan = editTextJurusan.getText().toString().trim();
104
105     class UpdateMahasiswa extends AsyncTask<Void,Void,String>{
106         ProgressDialog loading;
107         @Override
108         protected void onPreExecute() {
109             super.onPreExecute();
110             loading = ProgressDialog.show( context: TampilMahasiswa.this, title: "Updating...", message: "Wait...", indeterminate: false, cancelable: false);
111         }
112     }
```

7.c



```
MainActivity.java x activity_tampil_mahasiswa.xml x TampilMahasiswa.java x ListView.java x activity_list_view.xml x AndroidManifest.xml x
TampilMahasiswa deleteMahasiswa()

@Override
protected void onPostExecute(String s) {
    super.onPostExecute(s);
    loading.dismiss();
    Toast.makeText(context, TampilMahasiswa.this, s, Toast.LENGTH_LONG).show();
}

@Override
protected String doInBackground(Void... params) {
    HashMap<String, String> hashMap = new HashMap<>();
    hashMap.put(Configuration.KEY_EMP_ID, id);
    hashMap.put(Configuration.KEY_EMP_NAMA, nama);
    hashMap.put(Configuration.KEY_EMP_ALAMAT, alamat);
    hashMap.put(Configuration.KEY_EMP_JURUSAN, jurusan);

    RequestHandler rh = new RequestHandler();

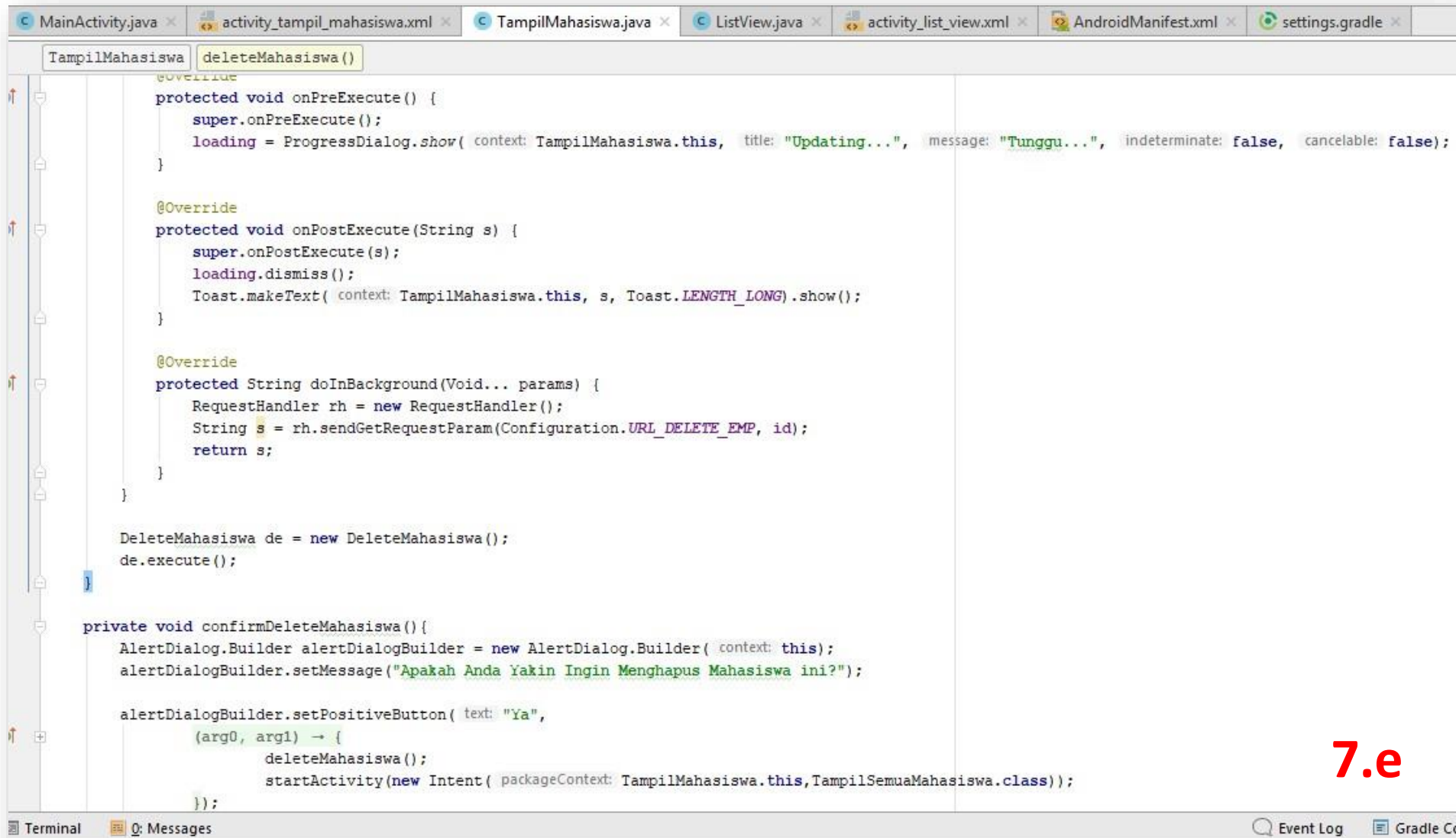
    String s = rh.sendPostRequest(Configuration.URL_UPDATE_EMP, hashMap);

    return s;
}

UpdateMahasiswa ue = new UpdateMahasiswa();
ue.execute();

private void deleteMahasiswa() {
    class DeleteMahasiswa extends AsyncTask<Void, Void, String> {
        ProgressDialog loading;
```

7.d



```
MainActivity.java × activity_tampil_mahasiswa.xml × TampilMahasiswa.java × ListView.java × activity_list_view.xml × AndroidManifest.xml × settings.gradle ×
TampilMahasiswa deleteMahasiswa()
@Override
protected void onPreExecute() {
    super.onPreExecute();
    loading = ProgressDialog.show( context: TampilMahasiswa.this, title: "Updating...", message: "Tunggu...", indeterminate: false, cancelable: false);
}

@Override
protected void onPostExecute(String s) {
    super.onPostExecute(s);
    loading.dismiss();
    Toast.makeText( context: TampilMahasiswa.this, s, Toast.LENGTH_LONG).show();
}

@Override
protected String doInBackground(Void... params) {
    RequestHandler rh = new RequestHandler();
    String s = rh.sendGetRequestParam(Configuration.URL_DELETE_EMP, id);
    return s;
}

DeleteMahasiswa de = new DeleteMahasiswa();
de.execute();

private void confirmDeleteMahasiswa(){
    AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder( context: this);
    alertDialogBuilder.setMessage("Apakah Anda Yakin Ingin Menghapus Mahasiswa ini?");

    alertDialogBuilder.setPositiveButton( text: "Ya",
        (arg0, arg1) → {
            deleteMahasiswa();
            startActivity(new Intent( packageContext: TampilMahasiswa.this,TampilSemuaMahasiswa.class));
        });
}
```

Terminal Messages Event Log Gradle Co

7.e

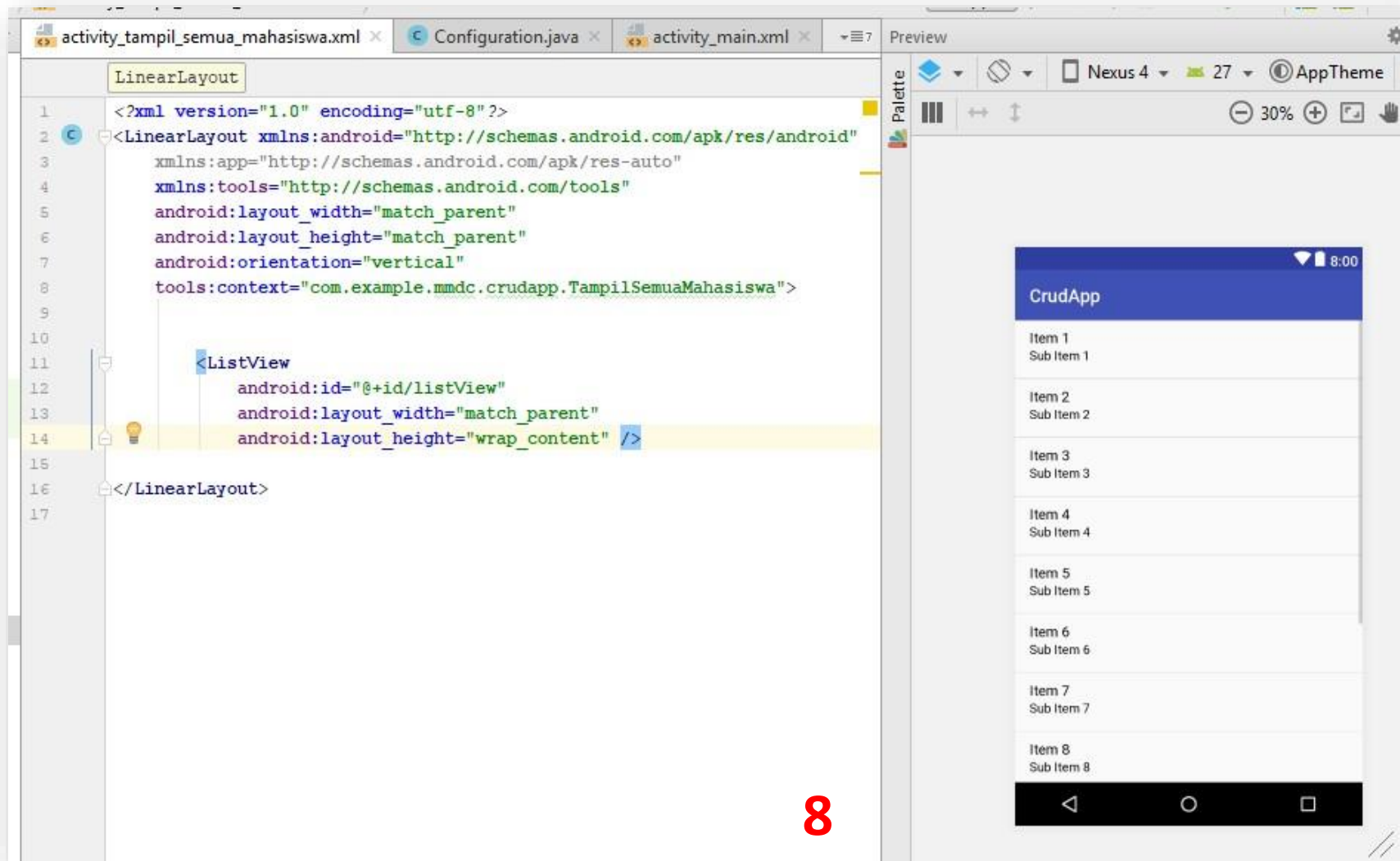
```
        alertDialogBuilder.setNegativeButton( text: "Tidak",
            new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface arg0, int arg1) {

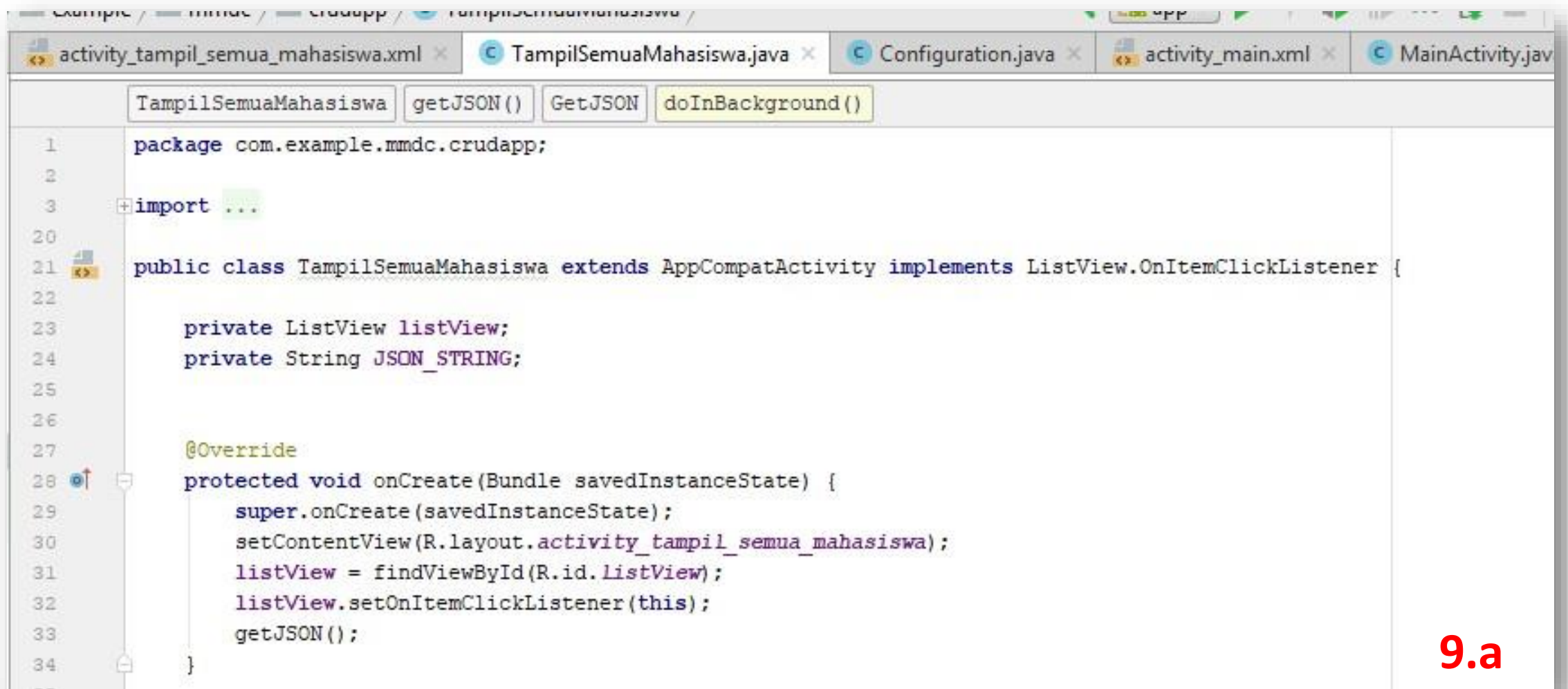
                }
            });

        AlertDialog alertDialog = alertDialogBuilder.create();
        alertDialog.show();
    }

    @Override
    public void onClick(View v) {
        if(v == buttonUpdate){
            updateMahasiswa();
        }

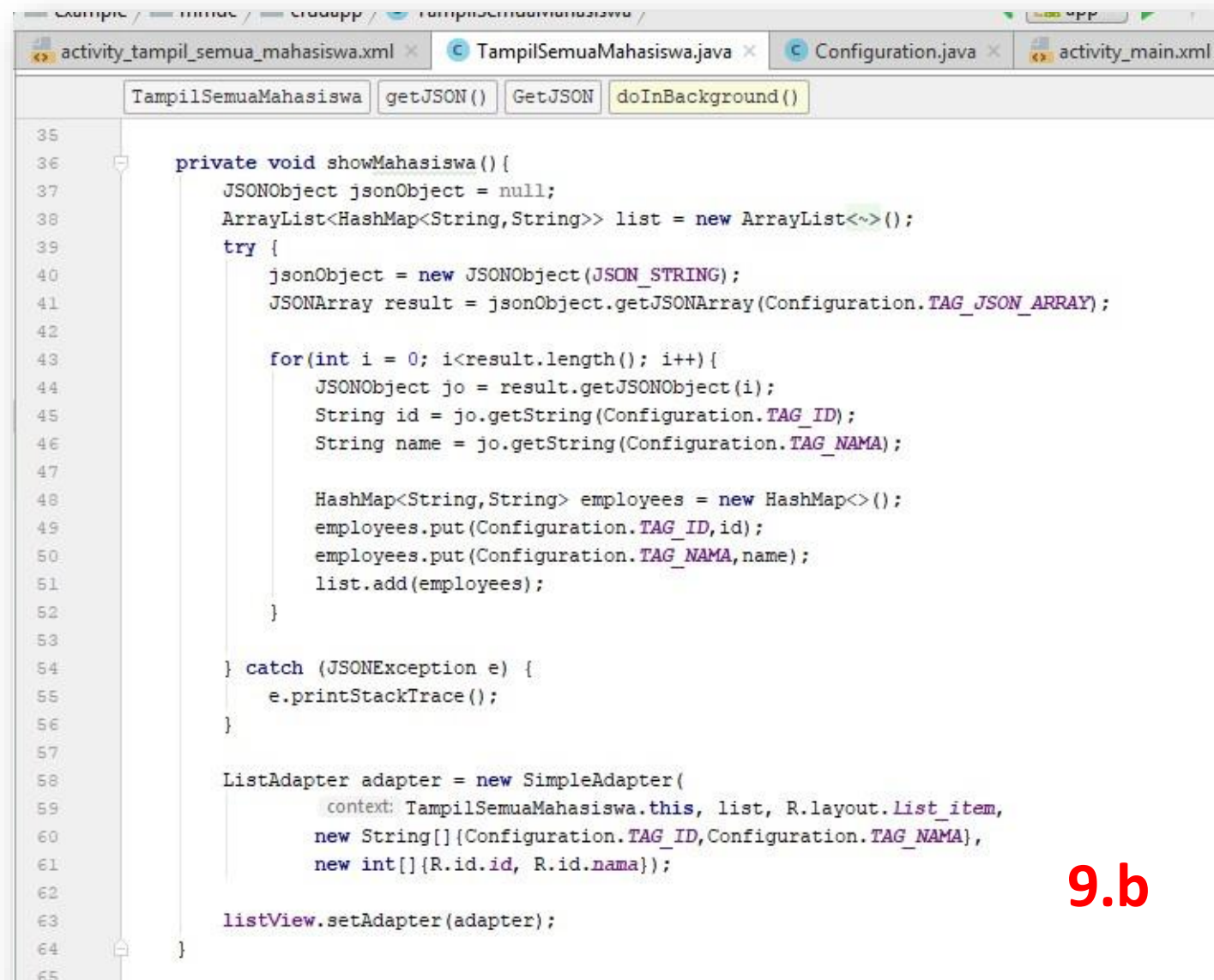
        if(v == buttonDelete){
            confirmDeleteMahasiswa();
        }
    }
}
```





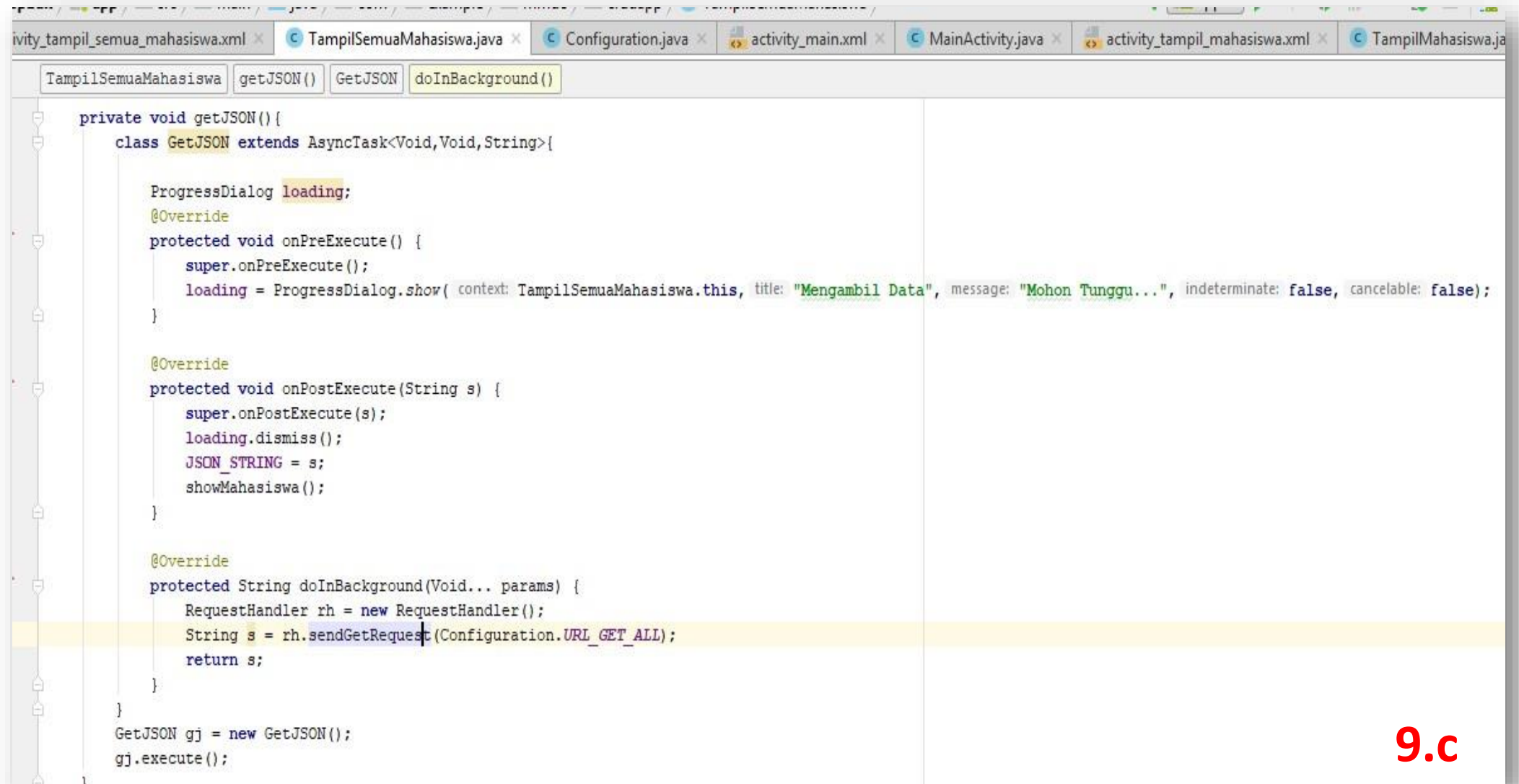
```
1 package com.example.mmdc.crudapp;
2
3 import ...
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21 public class TampilSemuaMahasiswa extends AppCompatActivity implements ListView.OnItemClickListener {
22
23     private ListView listView;
24     private String JSON_STRING;
25
26
27     @Override
28     protected void onCreate(Bundle savedInstanceState) {
29         super.onCreate(savedInstanceState);
30         setContentView(R.layout.activity_tampil_semua_mahasiswa);
31         listView = findViewById(R.id.listView);
32         listView.setOnItemClickListener(this);
33         getJSON();
34     }
35 }
```

9.a



```
35
36 private void showMahasiswa() {
37     JSONObject jsonObject = null;
38     ArrayList<HashMap<String,String>> list = new ArrayList<>();
39     try {
40         jsonObject = new JSONObject(JSON_STRING);
41         JSONArray result = jsonObject.getJSONArray(Configuration.TAG_JSON_ARRAY);
42
43         for(int i = 0; i<result.length(); i++){
44             JSONObject jo = result.getJSONObject(i);
45             String id = jo.getString(Configuration.TAG_ID);
46             String name = jo.getString(Configuration.TAG_NAMA);
47
48             HashMap<String,String> employees = new HashMap<>();
49             employees.put(Configuration.TAG_ID,id);
50             employees.put(Configuration.TAG_NAMA,name);
51             list.add(employees);
52         }
53
54     } catch (JSONException e) {
55         e.printStackTrace();
56     }
57
58     ListAdapter adapter = new SimpleAdapter(
59         context, TampilSemuaMahasiswa.this, list, R.layout.list_item,
60         new String[]{Configuration.TAG_ID,Configuration.TAG_NAMA},
61         new int[]{R.id.id, R.id.nama});
62
63     listView.setAdapter(adapter);
64 }
65
```

9.b



```
private void getJSON() {
    class GetJSON extends AsyncTask<Void, Void, String> {

        ProgressDialog loading;

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            loading = ProgressDialog.show(context: TampilSemuaMahasiswa.this, title: "Mengambil Data", message: "Mohon Tunggu...", indeterminate: false, cancelable: false);
        }

        @Override
        protected void onPostExecute(String s) {
            super.onPostExecute(s);
            loading.dismiss();
            JSON_STRING = s;
            showMahasiswa();
        }

        @Override
        protected String doInBackground(Void... params) {
            RequestHandler rh = new RequestHandler();
            String s = rh.sendGetRequest(Configuration.URL_GET_ALL);
            return s;
        }
    }

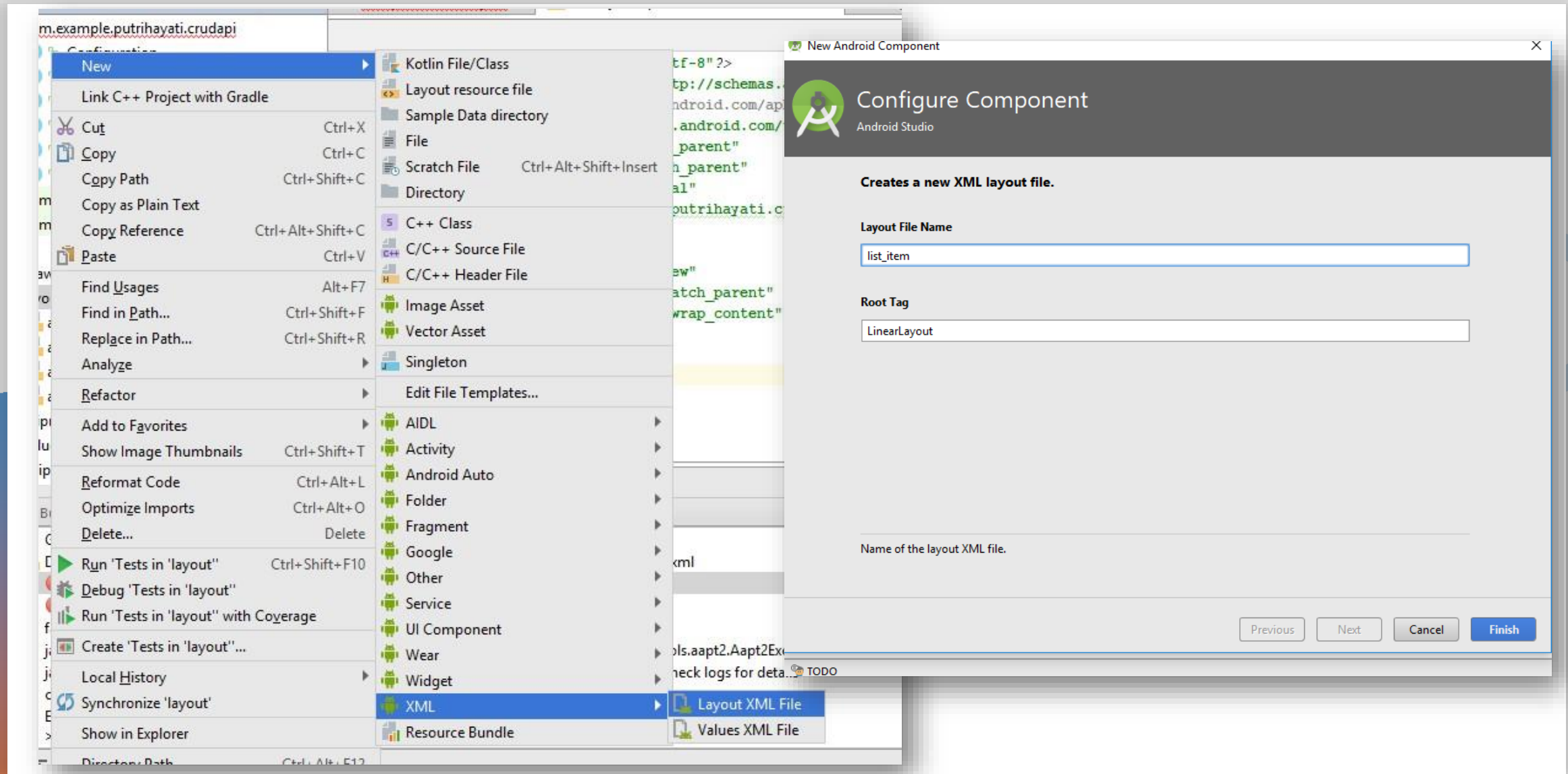
    GetJSON gj = new GetJSON();
    gj.execute();
}
```

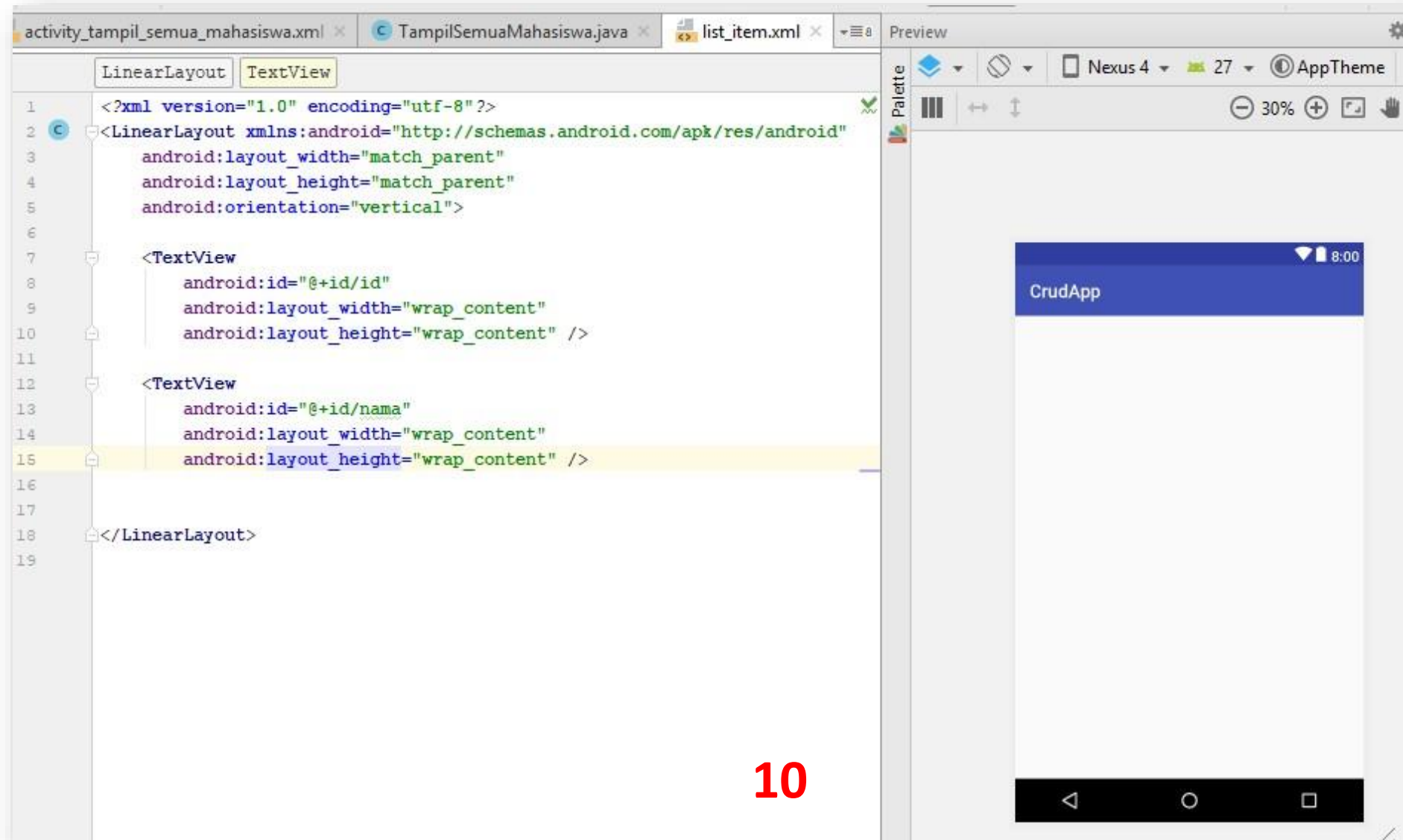
9.c

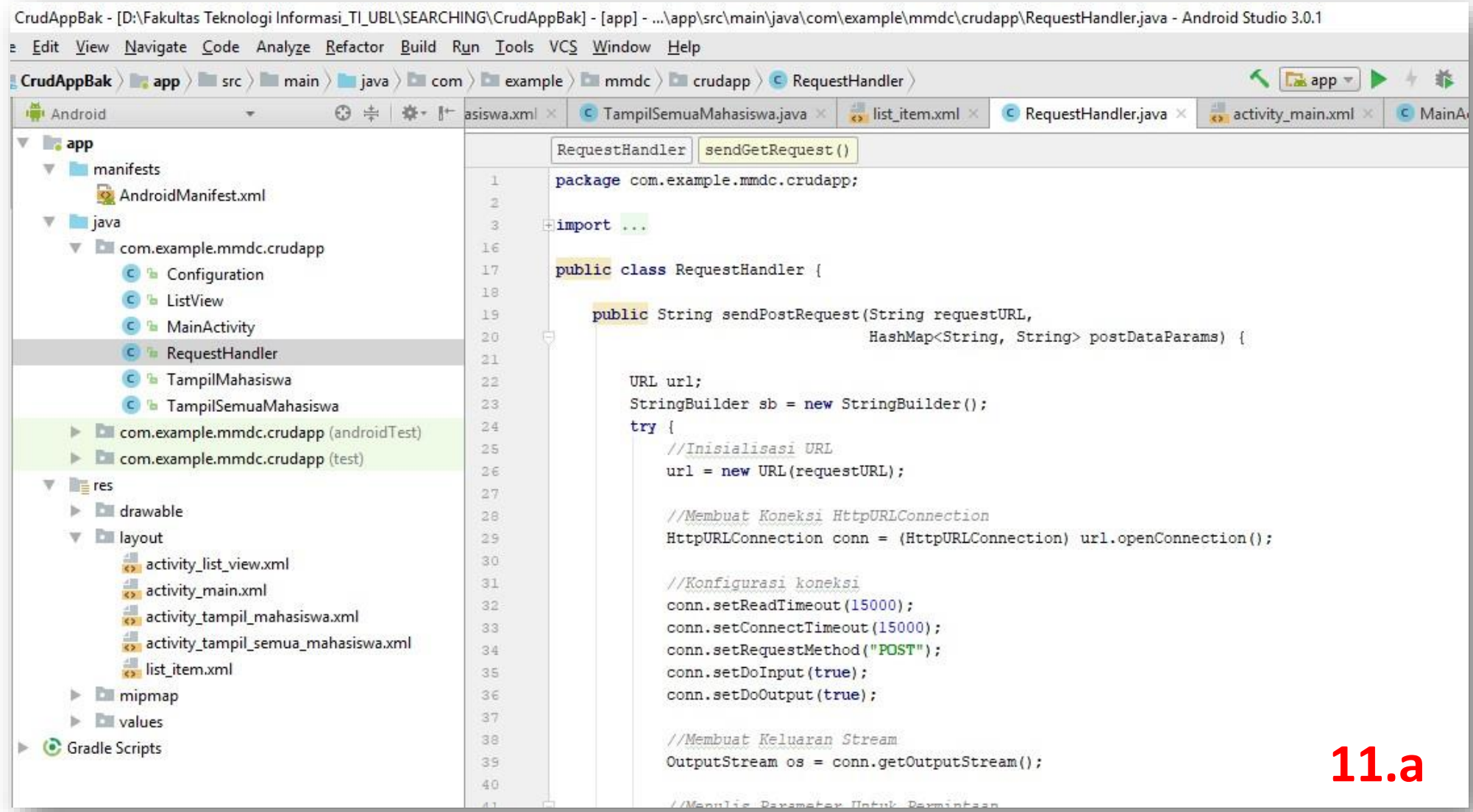
```
94      @Override
95      public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
96          Intent intent = new Intent( packageContext: this, TampilMahasiswa.class);
97          HashMap<String,String> map =(HashMap)parent.getItemAtPosition(position);
98          String mhsid = map.get(Configuration.TAG_ID).toString();
99          intent.putExtra(Configuration.MHS_ID,mhsid);
100          startActivity(intent);
101      }
102  }
103
```

9.d

11.1 Praktikum



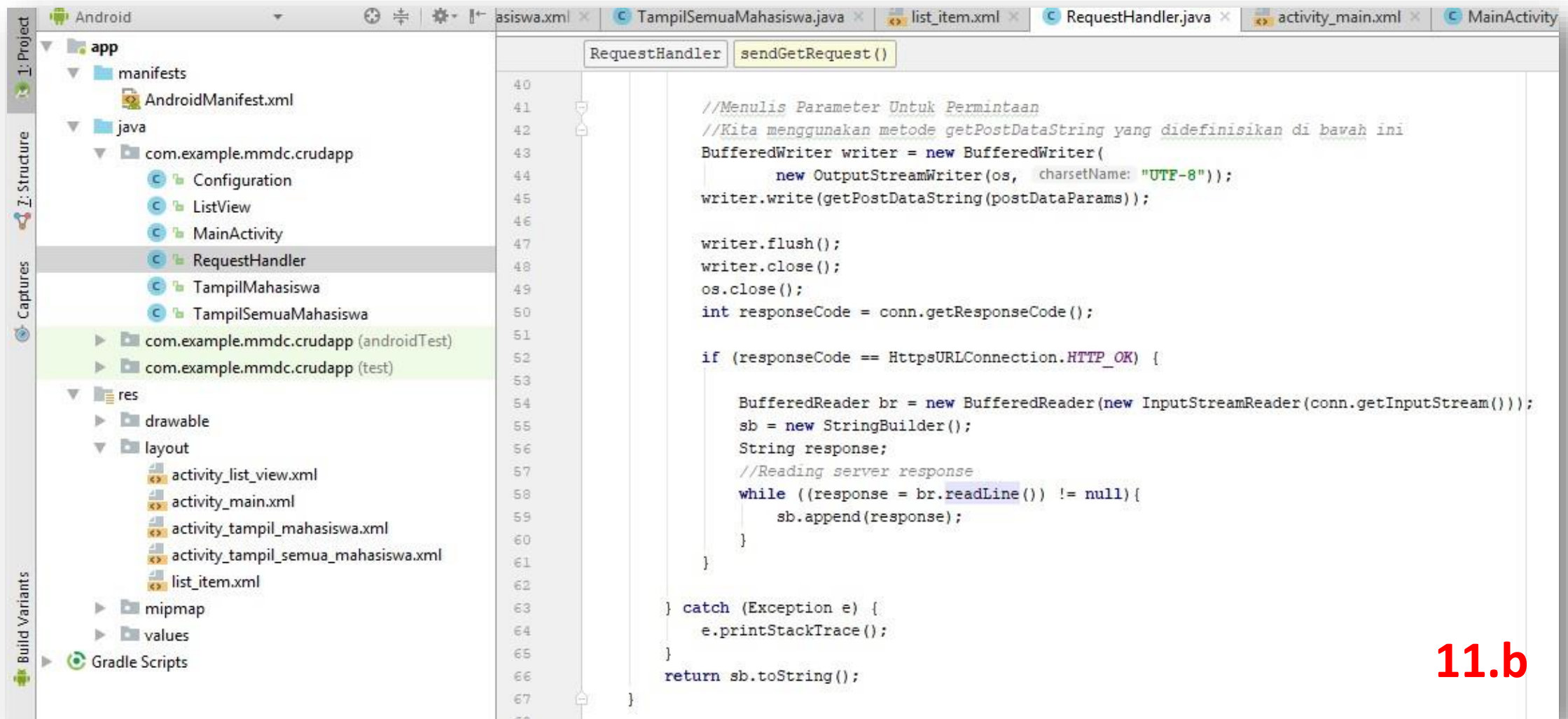




```
CrudAppBak - [D:\Fakultas Teknologi Informasi_TI_UBL\SEARCHING\CrudAppBak] - [app] - ...app\src\main\java\com\example\mmdc\crudapp\RequestHandler.java - Android Studio 3.0.1
Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CrudAppBak > app > src > main > java > com > example > mmdc > crudapp > RequestHandler
Android
app
  manifests
    AndroidManifest.xml
  java
    com.example.mmdc.crudapp
      Configuration
      ListView
      MainActivity
      RequestHandler
      TampilMahasiswa
      TampilSemuaMahasiswa
    com.example.mmdc.crudapp (androidTest)
    com.example.mmdc.crudapp (test)
  res
    drawable
    layout
      activity_list_view.xml
      activity_main.xml
      activity_tampil_mahasiswa.xml
      activity_tampil_semua_mahasiswa.xml
      list_item.xml
    mipmap
    values
  Gradle Scripts

RequestHandler sendGetRequest()
1 package com.example.mmdc.crudapp;
2
3 import ...
16
17 public class RequestHandler {
18
19     public String sendPostRequest(String requestURL,
20                                   HashMap<String, String> postDataParams) {
21
22         URL url;
23         StringBuilder sb = new StringBuilder();
24         try {
25             //Inisialisasi URL
26             url = new URL(requestURL);
27
28             //Membuat Koneksi HttpURLConnection
29             HttpURLConnection conn = (HttpURLConnection) url.openConnection();
30
31             //Konfigurasi koneksi
32             conn.setReadTimeout(15000);
33             conn.setConnectTimeout(15000);
34             conn.setRequestMethod("POST");
35             conn.setDoInput(true);
36             conn.setDoOutput(true);
37
38             //Membuat Keluaran Stream
39             OutputStream os = conn.getOutputStream();
40
41             //Memulis Parameter Untuk Permintaan
```

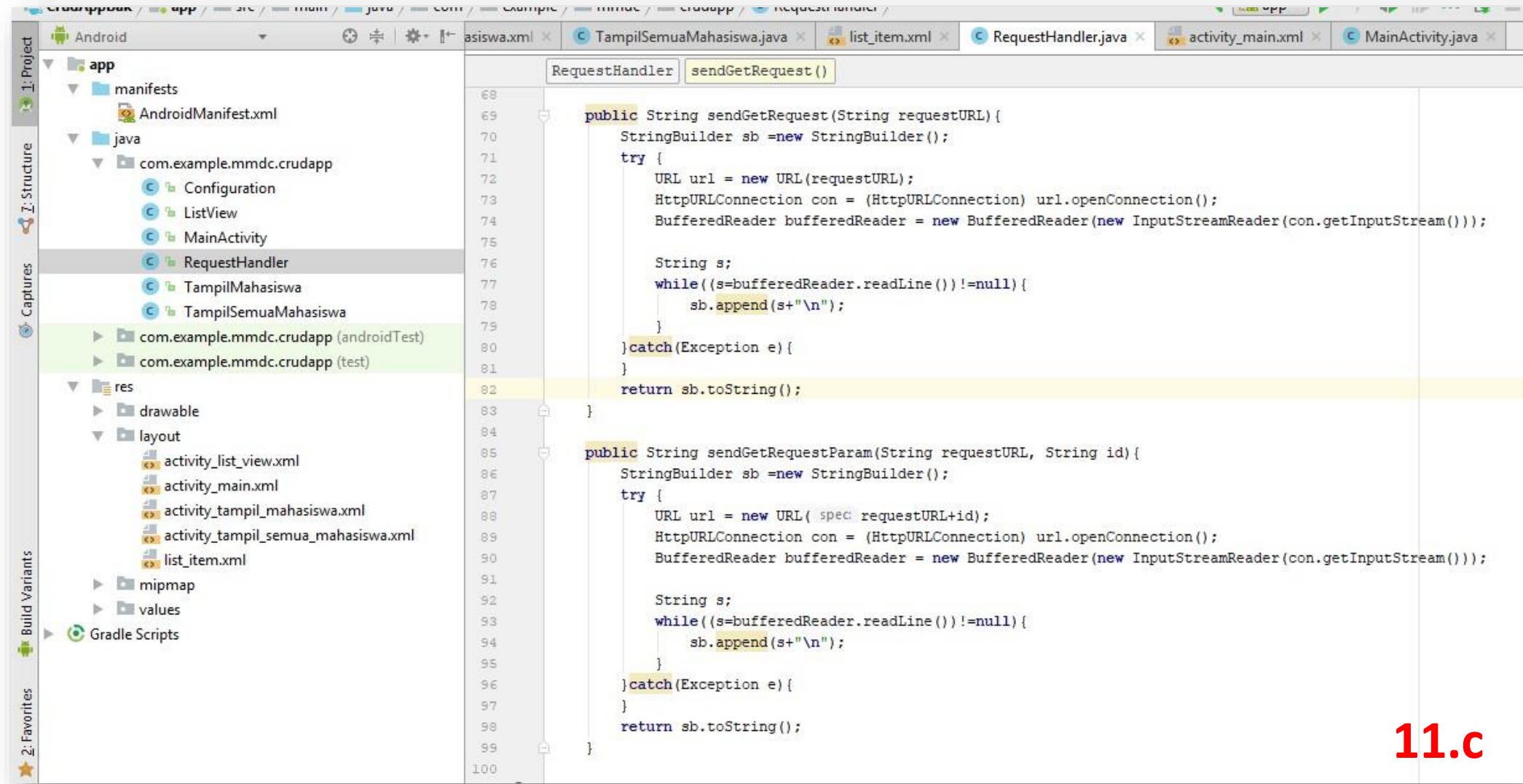
11.a



The screenshot displays the Android Studio interface. On the left, the 'Project' and 'Structure' toolbars are visible. The 'Structure' pane shows the package hierarchy: `com.example.mmdc.crudapp` containing `Configuration`, `ListView`, `MainActivity`, `RequestHandler` (selected), `TampilMahasiswa`, and `TampilSemuaMahasiswa`. Below this, the `res` directory is expanded, showing `drawable` and `layout` folders with various XML files. The main editor window shows the `RequestHandler` class with the `sendGetRequest()` method selected. The code in the editor is as follows:

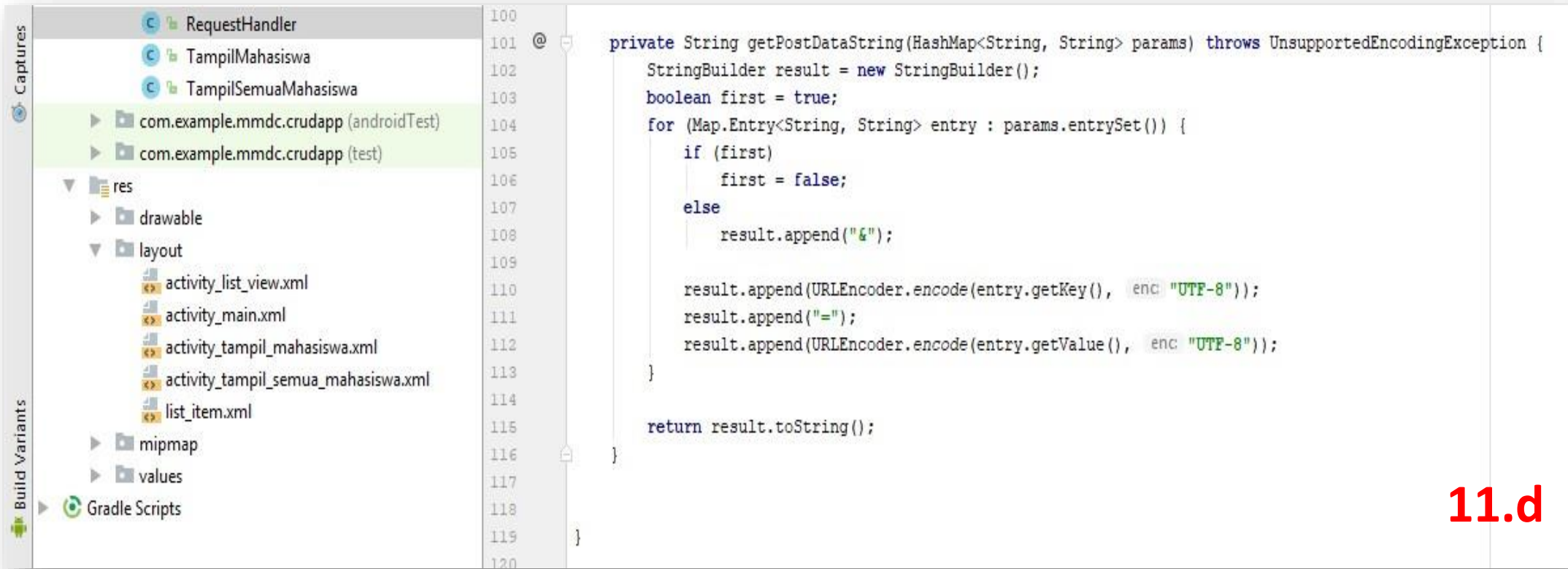
```
40  
41  
42 //Menulis Parameter Untuk Permintaan  
43 //Kita menggunakan metode postDataString yang didefinisikan di bawah ini  
44 BufferedWriter writer = new BufferedWriter(  
45     new OutputStreamWriter(os, charsetName: "UTF-8"));  
46 writer.write(postDataString(postDataParams));  
47  
48 writer.flush();  
49 writer.close();  
50 os.close();  
51 int responseCode = conn.getResponseCode();  
52  
53 if (responseCode == HttpURLConnection.HTTP_OK) {  
54     BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream()));  
55     sb = new StringBuilder();  
56     String response;  
57     //Reading server response  
58     while ((response = br.readLine()) != null) {  
59         sb.append(response);  
60     }  
61 }  
62  
63 } catch (Exception e) {  
64     e.printStackTrace();  
65 }  
66 return sb.toString();  
67 }
```

11.b



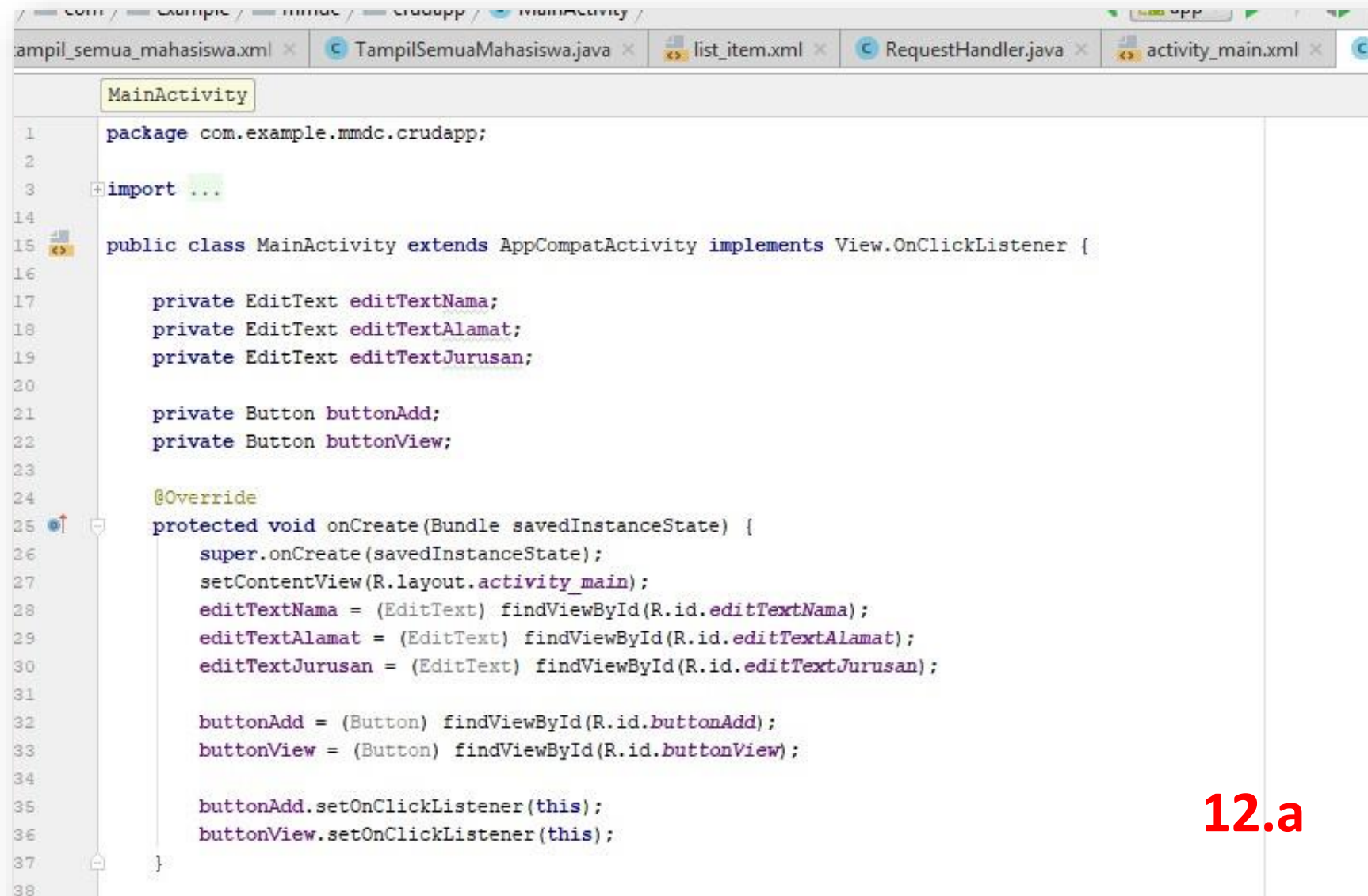
The screenshot displays the Android Studio interface. On the left, the 'Project' tab shows the package structure: `com.example.mmdc.crudapp`. The `RequestHandler` class is highlighted. The main editor window shows the source code for `RequestHandler.java`. The code defines two methods: `sendGetRequest()` and `sendGetRequestParam()`. Both methods use `HttpURLConnection` to open a connection to a specified URL, read the response line by line using `BufferedReader`, and append the data to a `StringBuilder` object. The `sendGetRequest()` method is currently selected in the editor.

```
68  
69  
70 public String sendGetRequest(String requestURL) {  
71     StringBuilder sb = new StringBuilder();  
72     try {  
73         URL url = new URL(requestURL);  
74         HttpURLConnection con = (HttpURLConnection) url.openConnection();  
75         BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(con.getInputStream()));  
76  
77         String s;  
78         while((s=bufferedReader.readLine())!=null) {  
79             sb.append(s+"\n");  
80         }  
81     } catch (Exception e) {  
82     }  
83     return sb.toString();  
84 }  
85  
86 public String sendGetRequestParam(String requestURL, String id) {  
87     StringBuilder sb = new StringBuilder();  
88     try {  
89         URL url = new URL(requestURL+id);  
90         HttpURLConnection con = (HttpURLConnection) url.openConnection();  
91         BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(con.getInputStream()));  
92  
93         String s;  
94         while((s=bufferedReader.readLine())!=null) {  
95             sb.append(s+"\n");  
96         }  
97     } catch (Exception e) {  
98     }  
99     return sb.toString();  
100 }
```



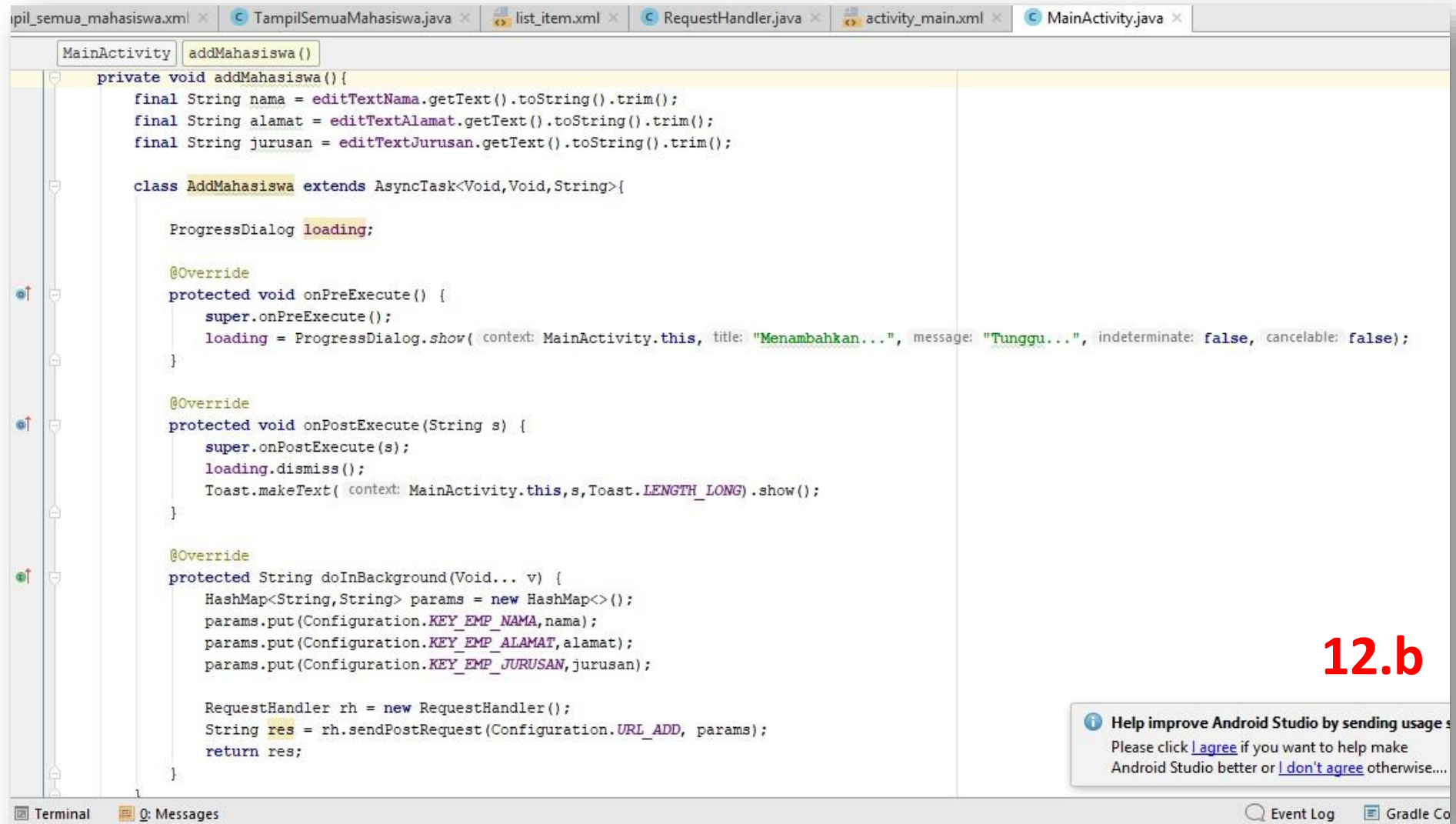
```
100
101 @
102
103 private String getPostDataString(HashMap<String, String> params) throws UnsupportedEncodingException {
104     StringBuilder result = new StringBuilder();
105     boolean first = true;
106     for (Map.Entry<String, String> entry : params.entrySet()) {
107         if (first)
108             first = false;
109         else
110             result.append("&");
111
112         result.append(URLEncoder.encode(entry.getKey(), enc "UTF-8"));
113         result.append("=");
114         result.append(URLEncoder.encode(entry.getValue(), enc "UTF-8"));
115     }
116
117     return result.toString();
118 }
119
120 }
```

11.d



```
1 package com.example.mmdc.crudapp;
2
3 import ...
4
14
15 public class MainActivity extends AppCompatActivity implements View.OnClickListener {
16
17     private EditText editTextNama;
18     private EditText editTextAlamat;
19     private EditText editTextJurusan;
20
21     private Button buttonAdd;
22     private Button buttonView;
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27         setContentView(R.layout.activity_main);
28         editTextNama = (EditText) findViewById(R.id.editTextNama);
29         editTextAlamat = (EditText) findViewById(R.id.editTextAlamat);
30         editTextJurusan = (EditText) findViewById(R.id.editTextJurusan);
31
32         buttonAdd = (Button) findViewById(R.id.buttonAdd);
33         buttonView = (Button) findViewById(R.id.buttonView);
34
35         buttonAdd.setOnClickListener(this);
36         buttonView.setOnClickListener(this);
37     }
38 }
```

12.a



```

MainActivity | addMahasiswa()
private void addMahasiswa(){
    final String nama = editTextNama.getText().toString().trim();
    final String alamat = editTextAlamat.getText().toString().trim();
    final String jurusan = editTextJurusan.getText().toString().trim();

    class AddMahasiswa extends AsyncTask<Void,Void,String>{

        ProgressDialog loading;

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            loading = ProgressDialog.show( context: MainActivity.this, title: "Menambahkan...", message: "Tunggu...", indeterminate: false, cancelable: false);
        }

        @Override
        protected void onPostExecute(String s) {
            super.onPostExecute(s);
            loading.dismiss();
            Toast.makeText( context: MainActivity.this,s,Toast.LENGTH_LONG).show();
        }

        @Override
        protected String doInBackground(Void... v) {
            HashMap<String,String> params = new HashMap<>();
            params.put(Configuration.KEY_EMP_NAMA,nama);
            params.put(Configuration.KEY_EMP_ALAMAT,alamat);
            params.put(Configuration.KEY_EMP_JURUSAN,jurusan);

            RequestHandler rh = new RequestHandler();
            String res = rh.sendPostRequest(Configuration.URL_ADD, params);
            return res;
        }
    }
}

```

12.b

Help improve Android Studio by sending usage s
Please click [I agree](#) if you want to help make
Android Studio better or [I don't agree](#) otherwise....

Terminal | Messages | Event Log | Gradle Co

```
        AddMahasiswa ae = new AddMahasiswa();  
        ae.execute();  
    }  
    @Override  
    public void onClick(View v) {  
        if (v == buttonAdd) {  
            addMahasiswa();  
        }  
  
        if (v == buttonView) {  
            startActivity(new Intent( packageContext: this, TampilSemuaMahasiswa.class));  
        }  
    }  
}
```

Terminal



Messages

ns (8 minutes ago)

12.c

crudapi

Nama Mahasiswa

Alamat

Jurusan

TAMBAH DATA MAHASISWA

DAFTAR LIST MAHASISWA

crudapi

Nama Mahasiswa

putri

Alamat

slipi

Jurusan

fe

TAMBAH DATA MAHASISWA

DAFTAR LIST MAHASISWA

crudapi

Nama Mahasiswa

Putri

Alamat

Jakarta

Jurusan

FTI

TAMBAH DATA MAHASISWA

DAFTAR LIST MAHASISWA

1234567890

QWERTYUIOP

ASDFGHJKL

↑ZXCVBNM↵

Sym🗣English(US)⌨

localhost/phpmyadmin/sql.php?server=1&db=android_api&table=mahasiswa&pos=0

phpMyAdmin

Recent Favorites

- New
- akademik
- android_api
 - New
 - mahasiswa
- contoh
- film
- information_schema
- mysql
- performance_schema
- phpmyadmin
- test

Server: 127.0.0.1 » Database: android_api » Table: mahasiswa

Browse Structure SQL Search Insert Export Import Privileges Operations

Showing rows 0 - 1 (2 total, Query took 0.0012 seconds.)

`SELECT * FROM `mahasiswa``

Profiling [Edit inline] [Edit] [Edit]

Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	id	nama	alamat	jurusan
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	18	Putri	Jakarta	FTI
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	19	putri	slipi	fe

Check all With selected: Edit Copy Delete Export

Console

Press Ctrl+Enter to execute query

```
> SELECT * FROM `mahasiswa`
>
```

192.168.100.7/api/detail

192.168.100.7/api/detail.php?id=18

Koneksi Berhasil {"result":[{"id":"18","nama":"Putri","alamat":"Jakarta","jurusan":"FTI"}]}

Any Question ?

