LAPORAN PRAKTIKUM
MOBILE PROGRAMING

MODUL 9
NETWORKING & WEB SERVICE (REST API)

Disusun oleh :

FAKHRI FAWWAZ AYDIN 2250081134

AIG - B

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS DAN INFORMATIKA
UNIVERSITAS JENDERAL ACHMAD YANI

2025

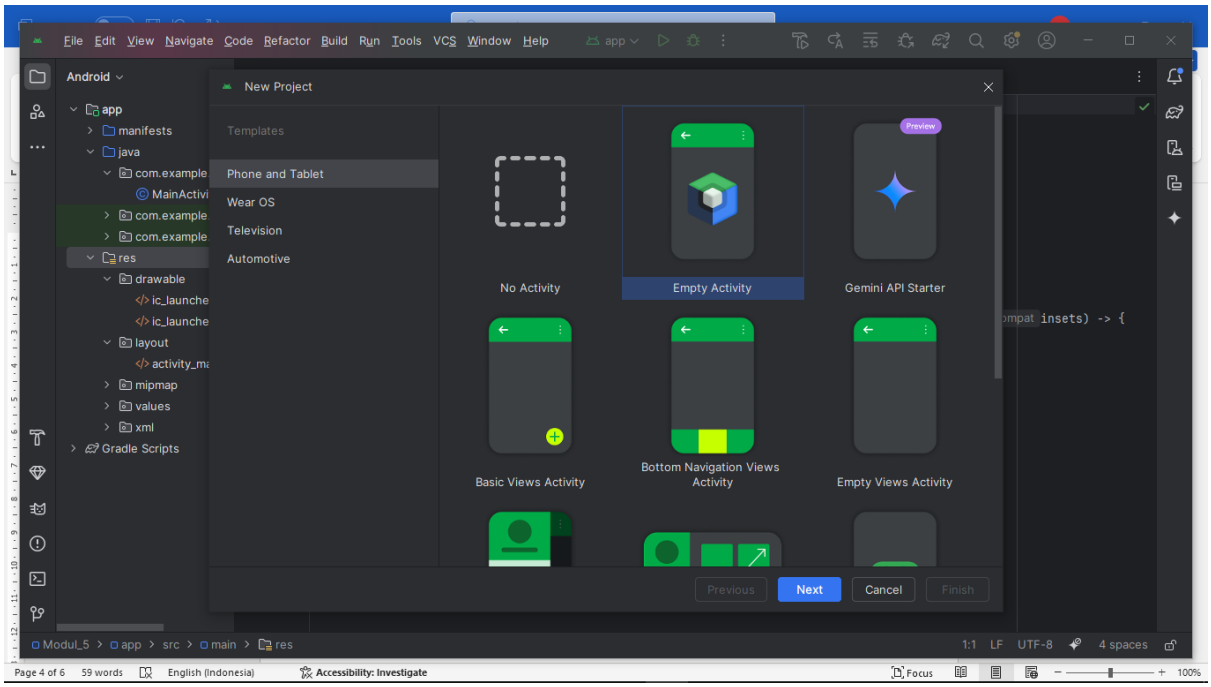
DAFTAR ISI

DAFTAR GAMBAR

BAB I
HASIL PRAKTIKUM

I.1. Latihan 1 Menampilkan data dari GET method

1. Buat project baru dengan menggunakan empty view.



Gambar 1 Project Empty View

2. Kemudian, buat tampilan untuk menampilkan text JSON pada activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
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/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Loading. . ."
        android:id="@+id/tvQuotes"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:textStyle="italic"

android:textAppearance="@style/TextAppearance.MaterialComponents.Sub
title1"
        android:gravity="center"
        android:layout_margin="10dp"
        />
```

```
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/tvQuotes"
        android:id="@+id/tvAnime"

android:textAppearance="@style/TextAppearance.MaterialComponents.Sub
title1"
        android:textStyle="bold"
        android:layout_margin="10dp"
        />
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/tvAnime"
        android:text="Random Quotes"
        android:id="@+id/btnGet"
        />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Post Data >>"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_margin="10dp"
        android:id="@+id/btnPost"
        />

</RelativeLayout>
```

3. Selanjutnya, buat logika pada MainActivity.java untuk mendapatkan data dari endpoint.

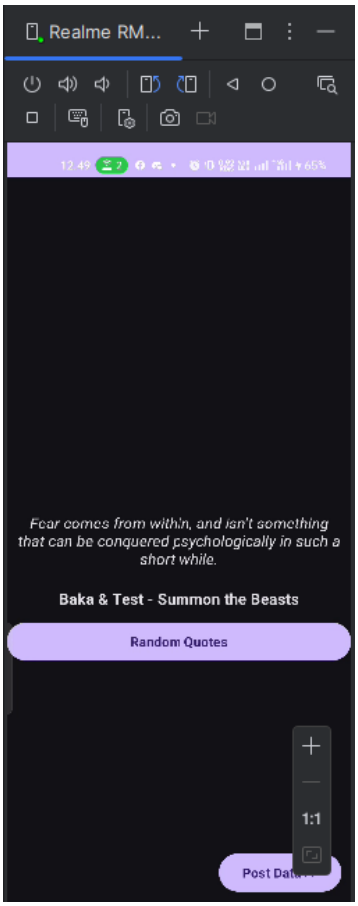
```
package com.example.modul9;

import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class MainActivity extends AppCompatActivity {
    private TextView quotes, anime;
```

```
    private Button btnGet, btnPost;

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

        quotes = findViewById(R.id.tvQuotes);
        anime = findViewById(R.id.tvAnime);
        btnGet = findViewById(R.id.btnGet);
        btnPost = findViewById(R.id.btnPost);

        // Load initial data
        fetchRandomQuote();

        btnGet.setOnClickListener(v -> fetchRandomQuote());

        btnPost.setOnClickListener(v -> {
            Intent i = new Intent(MainActivity.this,
PostActivity.class);
            startActivity(i);
        });
    }

    private void fetchRandomQuote() {
        new
GetData().execute("https://api.animechan.io/v1/quotes/random");
    }

    private class GetData extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... urls) {
            HttpURLConnection conn = null;
            BufferedReader reader = null;

            try {
                URL url = new URL(urls[0]);
                conn = (HttpURLConnection) url.openConnection();
                conn.setRequestMethod("GET");
                conn.setConnectTimeout(10000);
                conn.setReadTimeout(10000);

                int responseCode = conn.getResponseCode();
                if (responseCode != HttpURLConnection.HTTP_OK) {
                    return "Error: HTTP " + responseCode;
                }

                InputStream in = conn.getInputStream();
                reader = new BufferedReader(new
InputStreamReader(in));
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line);
                }
                return sb.toString();
            } catch (Exception e) {
                Log.e("API_ERROR", "Error fetching data", e);
                return null;
            } finally {
```

```
                if (reader != null) {
                    try {
                        reader.close();
                    } catch (IOException e) {
                        Log.e("API_ERROR", "Error closing stream",
e);
                    }
                }
                if (conn != null) {
                    conn.disconnect();
                }
            }
        }

        @Override
        protected void onPostExecute(String result) {
            if (result == null) {
                quotes.setText("Gagal terhubung ke server");
                anime.setText("");
                return;
            }

            Log.d("API_RESPONSE", "Full API response: " + result);
// Tetap biarkan log ini untuk debugging

            try {
                JSONObject jsonResponse = new JSONObject(result);

                // --- Perubahan Utama Ada di Sini ---
                // 1. Ambil objek 'data' terlebih dahulu
                if (jsonResponse.has("data")) {
                    JSONObject dataObject =
jsonResponse.getJSONObject("data");

                    // 2. Dari objek 'data', ambil 'content' untuk
quotes
                    if (dataObject.has("content")) {
                        String quoteText =
dataObject.getString("content");
                        quotes.setText(quoteText);
                    } else {
                        quotes.setText("Kunci 'content' tidak
ditemukan.");
                    }

                    // 3. Dari objek 'data', ambil objek 'anime'
                    if (dataObject.has("anime")) {
                        JSONObject animeObject =
dataObject.getJSONObject("anime");
                        // 4. Dari objek 'anime', ambil 'name'
                        if (animeObject.has("name")) {
                            String animeText =
animeObject.getString("name");
                            anime.setText(animeText);
                        } else {
                            anime.setText("Kunci 'name' dalam
'anime' tidak ditemukan.");
                        }
                    } else {
                        anime.setText("Kunci 'anime' tidak
ditemukan.");
```

```
                    }

                } else {
                    quotes.setText("Kunci 'data' tidak ditemukan.");
                    anime.setText("Respon API: " + result);
                }
                // ------------------------------------

            } catch (JSONException e) {
                quotes.setText("Error parsing data
(JSONException)");
                anime.setText("Respon: " + result);
                Log.e("JSON_ERROR", "Full response (parsing failed):
" + result, e);
            }
        }
    }
}
```

4. Berikut adalah hasilnya



Gambar 2 Hasil mengambil data API

Analisa:

Pada latihan pertama ini, mengimplementasi program sederhana yang menampilkan kutipan acak dari API publik Animechan. Program ini terdiri dari satu Activity utama (MainActivity) yang menginisialisasi antarmuka pengguna dengan dua

TextView (quotes dan anime) dan dua Button (btnGet dan btnPost). Ketika aplikasi dijalankan (onCreate), metode fetchRandomQuote() akan langsung dipanggil untuk memuat data kutipan pertama. Tombol Get akan memicu pemanggilan API ulang, sedangkan tombol Post akan membuka PostActivity

Metode fetchRandomQuote() menjalankan AsyncTask bernama GetData, yang melakukan pemanggilan API secara asynchronous untuk menghindari blokir pada thread utama. Dalam metode doInBackground, koneksi HTTP dibuka menggunakan HttpURLConnection, dan data JSON dari endpoint API https://api.animechan.io/v1/quotes/random dibaca menggunakan BufferedReader. Hasil respon dikembalikan sebagai String dan diproses lebih lanjut dalam metode onPostExecute.

Pada bagian onPostExecute, string hasil API akan di-*parse* ke dalam objek JSONObject. Kode ini diasumsikan bahwa respon API memiliki struktur bersarang dengan objek data, dan di dalamnya terdapat properti content (isi kutipan) serta objek anime yang memiliki name (nama anime).

I.2. Latihan 2 Mengirimkan Data

1. Tambahkan button pada activity_main.xml untuk memanggil activity post.

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Post Data >>"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_margin="10dp"
    android:id="@+id/btnPost"
    />
```

2. Kemudian, tambahkan Event Listener untuk button tersebut pada MainActiviy.java

```
btnPost.setOnClickListener(v -> {
  Intent i = new Intent(MainActivity.this, PostActivity.class);
  startActivity(i);
});
```

3. Selanjut buat tampilan pada post_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/main"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Masukan Nama Kamu"
        android:id="@+id/etNama"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:layout_marginBottom="10dp"
        />
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Kirim Data"
        android:id="@+id/btnSend"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/etNama"
        />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        android:id="@+id/tvRespond"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/btnSend"
        android:layout_marginTop="20dp"
        />

</RelativeLayout>
```

4. Dan buat logika untuk melakukan post pada PostActivity.java

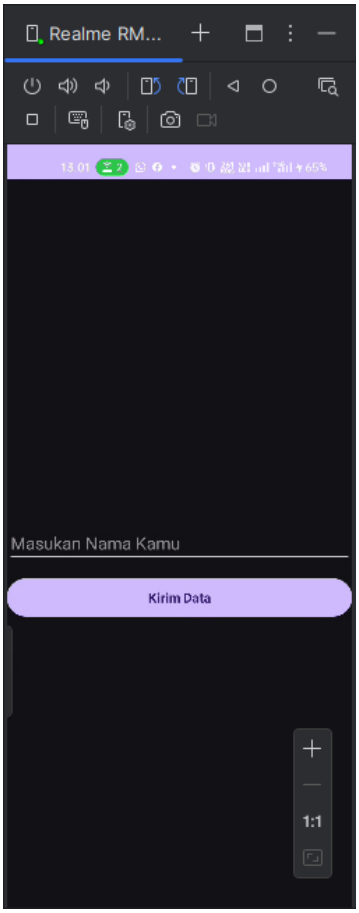
```
package com.example.modul9;

import android.app.ProgressDialog;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.URL;
```

```
import java.nio.charset.StandardCharsets;

public class PostActivity extends AppCompatActivity {
    private EditText etNama;
    private Button btnKirim;
    private TextView tvRespond;

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

        etNama = findViewById(R.id.etNama);
        tvRespond = findViewById(R.id.tvRespond);
        btnKirim = findViewById(R.id.btnSend);

        btnKirim.setOnClickListener(v -> {
            String nama = etNama.getText().toString().trim();
            if (nama.isEmpty()) {
                Toast.makeText(PostActivity.this, "Nama tidak boleh
kosong", Toast.LENGTH_SHORT).show();
                return;
            }

            String url = "https://postman-echo.com/post";
            new SendData().execute(url, nama);
        });
    }

    private class SendData extends AsyncTask<String, Void, String>
{
        private ProgressDialog progressDialog;
        private HttpURLConnection connection;

        @Override
        protected void onPreExecute() {
            progressDialog = new ProgressDialog(PostActivity.this);
            progressDialog.setMessage("Loading...");
            progressDialog.setCancelable(false);
            progressDialog.show();
        }

        @Override
        protected String doInBackground(String... params) {
            String urlString = params[0];
            String name = params[1];

            try {
                URL url = new URL(urlString);
                connection = (HttpURLConnection)
url.openConnection();
                connection.setRequestMethod("POST");
                connection.setConnectTimeout(5000);
                connection.setReadTimeout(5000);
                connection.setDoOutput(true);
                connection.setRequestProperty("Content-Type",
"application/x-www-form-urlencoded");

                String postData = "name=" + name;
                try (OutputStream os =
connection.getOutputStream();
```

```
                    BufferedWriter writer = new BufferedWriter(
                            new OutputStreamWriter(os,
StandardCharsets.UTF_8))) {
                    writer.write(postData);
                    writer.flush();
                }

                int responseCode = connection.getResponseCode();
                if (responseCode == HttpURLConnection.HTTP_OK) {
                    try (InputStream in =
connection.getInputStream();
                         BufferedReader reader = new
BufferedReader(
                                 new InputStreamReader(in,
StandardCharsets.UTF_8))) {
                        StringBuilder result = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null)
{
                            result.append(line);
                        }
                        return result.toString();
                    }
                } else {
                    return "Error: HTTP " + responseCode;
                }
            } catch (IOException e) {
                Log.e("POST_ERROR", "Error sending data", e);
                return "Koneksi Gagal: " + e.getMessage();
            } finally {
                if (connection != null) {
                    connection.disconnect();
                }
            }
        }

        @Override
        protected void onPostExecute(String result) {
            progressDialog.dismiss();

            if (result == null) {
                tvRespond.setText("Tidak ada respons dari server");
                return;
            }

            try {
                if (result.startsWith("Error:")) {
                    tvRespond.setText(result);
                    return;
                }

                JSONObject json = new JSONObject(result);
                JSONObject form = json.getJSONObject("form");
                String nameResponse = form.getString("name");
                tvRespond.setText("Respon: " + nameResponse);
            } catch (JSONException e) {
                Log.e("JSON_ERROR", "Error parsing response", e);
                tvRespond.setText("Format respons tidak valid: " +
result);
            }
        }
```

```
        }
}
```

5. Dan inilah hasilnya



Gambar 3 Post Activity

Analisa:

Pada PostActivity, menangani pengiriman data ke server menggunakan metode HTTP POST. Aktivitas ini memiliki antarmuka yang terdiri dari satu EditText (etNama) untuk input nama, satu Button (btnKirim) untuk mengirim data, dan satu TextView (tvRespond) untuk menampilkan hasil respons dari server. Ketika pengguna menekan tombol "Kirim", aplikasi akan memvalidasi bahwa input tidak kosong. Jika valid, maka proses pengiriman data dilakukan ke endpoint https://postman-echo.com/post, yaitu layanan dummy testing dari Postman yang akan mengembalikan data yang dikirim.

Pengiriman data dilakukan secara asynchronous menggunakan AsyncTask bernama SendData. Di dalam doInBackground, dibangun koneksi HTTP POST menggunakan HttpURLConnection. Data yang dikirim dalam format application/x-www-form-urlencoded dengan parameter name, yang berasal dari input pengguna.

Setelah data dikirim, aplikasi membaca respons dari server, menggabungkan seluruh isi respons ke dalam sebuah string, dan mengembalikannya ke onPostExecute.

Sebelum proses jaringan dimulai, ProgressDialog ditampilkan untuk memberi tahu pengguna bahwa data sedang diproses. Setelah selesai, dialog ditutup dan respons dari server akan di-*parse* sebagai objek JSON. Respons ini diasumsikan memiliki objek form yang berisi pasangan kunci-nilai yang dikirim, termasuk name. Nilai dari name ditampilkan kembali di aplikasi sebagai bentuk konfirmasi.

BAB II
KESIMPULAN

Berdasarkan analisis terhadap dua aktivitas dalam aplikasi Android (MainActivity dan PostActivity), dapat disimpulkan bahwa aplikasi ini dirancang untuk mendemonstrasikan dua jenis komunikasi dengan server: pengambilan data (GET) dan pengiriman data (POST) menggunakan AsyncTask dan HttpURLConnection.

Pada MainActivity, aplikasi mengambil data kutipan dari API publik Animechan dan menampilkannya ke pengguna. Namun, terdapat kekeliruan dalam struktur parsing JSON karena asumsi adanya objek data, padahal struktur respons asli API tersebut tidak memiliki objek tersebut. Meski begitu, mekanisme permintaan dan penanganan respons sudah mencerminkan praktik yang tepat dalam penggunaan koneksi HTTP GET, termasuk penanganan kesalahan jaringan dan parsing.

Sementara itu, pada PostActivity, aplikasi berhasil mengirimkan data input pengguna ke endpoint dummy Postman Echo melalui metode POST. Proses ini mencakup validasi input, pengiriman data dalam format x-www-form-urlencoded, penanganan respons berupa JSON, serta penggunaan ProgressDialog sebagai indikator proses. Struktur parsing JSON pada bagian ini sudah sesuai dengan respons dari layanan postman-echo.com.