STMIK KAPUTAMA

PRAKTIKUM

PENGANTAR PEMROGRAMAN BERGERAK

Disusun Oleh:

Kristina Annatasia Br Sitepu, M.Kom

Ratih Puspadini, ST., M.Kom

PERTEMUAN 12
PENYIMPANAN DATA (LANJUTAN)

12.1 TUJUAN PEMBELAJARAN :

- Mahasiswa dapat memahami jenis penyimpanan data dalam pengembangan aplikasi Android.
- Mahasiswa dapat memahami kelebihan dan kekurangan dari masing-masing jenis penyimpanan data.
- Mahasiswa mampu memilih jenis penyimpanan data yang tepat sesuai dengan kebutuhan aplikasi Android yang dikembangkan.
- Mahasiswa mampu menerapkan penyimpanan data dalam aplikasi Android.

12.2 ALAT DAN BAHAN :

a. Laptop/PC
b. Android Studio
c. Sistem Operasi Windows

12.3 MATERI

A. Menghubungkan MySQL dengan Android Studio

Pada saat mengembangkan aplikasi Android, seringkali kita perlu menyimpan dan mengelola data di luar perangkat, terutama untuk aplikasi yang membutuhkan penyimpanan data dalam jumlah besar atau yang datanya harus dibagikan antar pengguna. Salah satu pendekatan yang umum digunakan adalah dengan menyimpan data pada database server, dan MySQL adalah salah satu sistem manajemen basis data yang populer digunakan untuk tujuan ini. Dalam materi ini, kita akan belajar bagaimana menghubungkan aplikasi Android dengan database MySQL untuk mengelola data secara efisien.

1. Apa itu MySQL?

MySQL adalah sistem manajemen basis data relasional (RDBMS) yang populer, bersifat open-source, dan menggunakan Structured Query Language (SQL) untuk pengelolaan data. MySQL sering digunakan dalam pengembangan aplikasi web dan mobile karena memiliki performa tinggi, reliabilitas yang baik, serta didukung komunitas yang luas. MySQL cocok digunakan pada server untuk menangani data dalam jumlah besar dan memungkinkan aplikasi untuk berinteraksi dengan basis data melalui bahasa SQL.

2. Arsitektur Client-Server di Aplikasi Android

Aplikasi Android yang mengelola data di database MySQL menggunakan arsitektur client-server:

- Client : Aplikasi Android yang berjalan di perangkat pengguna berperan sebagai client, yang mengirimkan permintaan (request) ke server untuk melakukan operasi pada data, seperti menyimpan data, membaca data, dan lain sebagainya.
- Server : Server adalah tempat MySQL berjalan, berisi data yang disimpan dan diakses oleh aplikasi. Selain MySQL, server ini juga akan menjalankan aplikasi berbasis PHP untuk membantu proses komunikasi antara aplikasi Android dan database MySQL.

3. Mengapa Menggunakan PHP sebagai Perantara?

Android tidak secara langsung mendukung koneksi ke database MySQL yang berada di server. Karena itu, kita memerlukan skrip antara, dalam hal ini PHP, untuk berkomunikasi dengan MySQL:

- PHP adalah bahasa pemrograman server-side yang mudah diintegrasikan dengan MySQL.
- Skrip PHP bertindak sebagai API (Application Programming Interface) yang menerima permintaan dari aplikasi Android, kemudian meneruskan permintaan ini ke MySQL dan mengembalikan hasilnya ke aplikasi dalam format yang dapat dimengerti Android, seperti JSON (JavaScript Object Notation).

4. Teknik API REST (Representational State Transfer)

API adalah cara yang umum digunakan untuk memungkinkan aplikasi mengakses sumber daya data di server:

- REST API adalah gaya arsitektur yang banyak digunakan untuk membuat API, di mana setiap operasi dilakukan menggunakan HTTP request sederhana (GET, POST, PUT, DELETE).
- Dalam hal ini, kita akan menggunakan REST API untuk melakukan operasi CRUD (Create, Read, Update, Delete) dari aplikasi Android ke database MySQL.

5. Format Data: JSON

Setelah data berhasil diambil dari MySQL melalui skrip PHP, data tersebut akan dikirim ke aplikasi Android dalam format JSON. JSON adalah format data berbasis teks yang ringan dan mudah dibaca, cocok untuk pengiriman data melalui jaringan, dan mudah diproses oleh

aplikasi Android. JSON sangat berguna dalam mengirimkan data yang kompleks antara aplikasi dan server dengan cara yang efisien.

6. Mekanisme Koneksi antara Android dan MySQL melalui PHP

Proses koneksi dari aplikasi Android ke MySQL melibatkan beberapa tahapan:

1) Koneksi ke Internet: Aplikasi Android harus memiliki izin untuk mengakses internet karena komunikasi dengan server dilakukan melalui jaringan.
2) Mengirim Request ke Server: Aplikasi Android mengirimkan request HTTP ke server, misalnya untuk meminta data atau mengirim data baru.
3) Proses di Server oleh PHP: Server menerima request dari Android dan skrip PHP memprosesnya. Jika request berupa operasi baca data, PHP akan mengambil data dari MySQL, sedangkan jika request berupa simpan data, PHP akan memasukkan data baru ke MySQL.
4) Pengiriman Data Kembali ke Android: Jika ada data yang diminta, server mengirimkan respons dalam format JSON yang akan diproses oleh aplikasi Android.

7. Implementasi CRUD (Create, Read, Update, Delete)

Operasi CRUD adalah inti dari pengelolaan data dalam database:

- Create: Menambah data baru ke dalam tabel di MySQL.
- Read: Membaca data dari tabel.
- Update: Mengubah data yang ada di tabel.
- Delete: Menghapus data dari tabel.

Aplikasi Android akan mengirimkan permintaan ini melalui HTTP ke server, dan skrip PHP di server akan mengeksekusi perintah tersebut di MySQL.

Manfaat Menghubungkan Android Studio ke MySQL

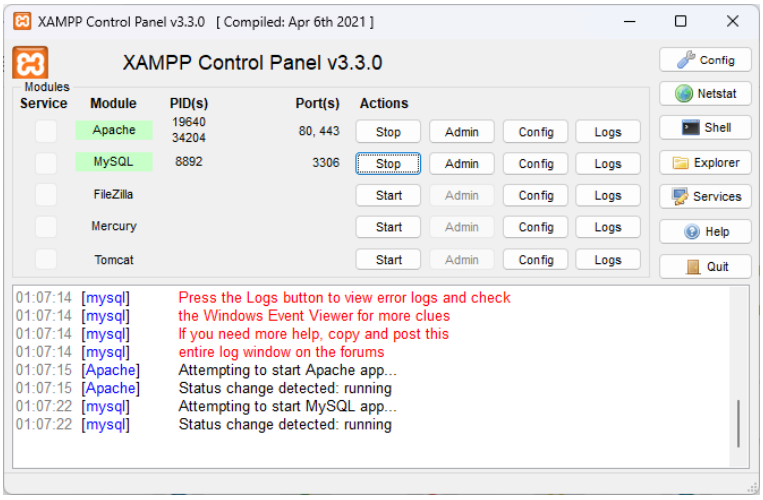
Dengan menghubungkan Android Studio ke MySQL:

- Aplikasi Android dapat menyimpan dan mengambil data dari server, memungkinkan data diakses oleh banyak pengguna.
- Data yang disimpan di server dapat di-backup dan diatur dengan lebih baik.
- Data dapat diakses dari berbagai perangkat yang terhubung ke aplikasi, sehingga sangat cocok untuk aplikasi multi-user seperti e-commerce, media sosial, dan manajemen tugas.

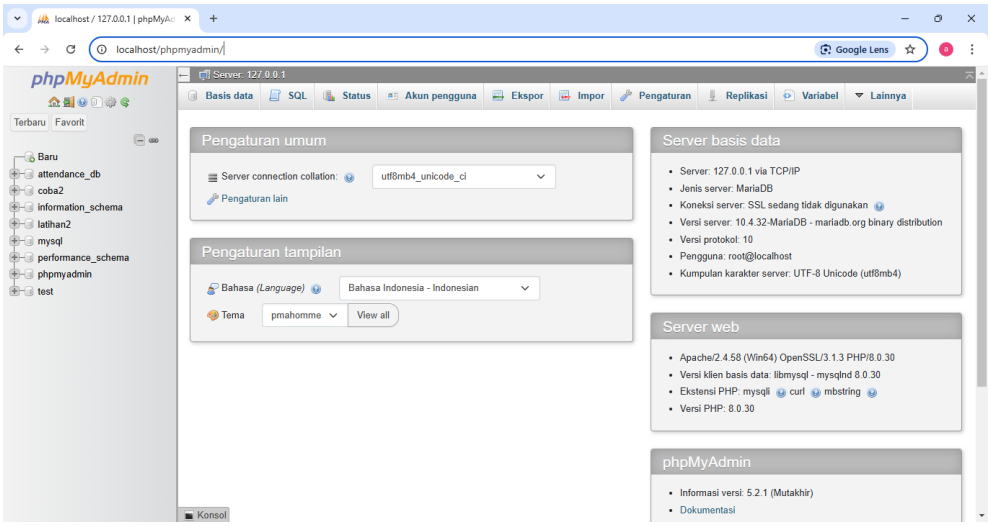
Dengan memahami konsep dasar ini, kita dapat mulai melakukan implementasi teknis untuk menghubungkan aplikasi Android dengan database MySQL menggunakan skrip PHP sebagai perantara, yang nantinya akan berfungsi sebagai API REST yang akan berinteraksi dengan aplikasi Android.

12.4 PRAKTIKUM

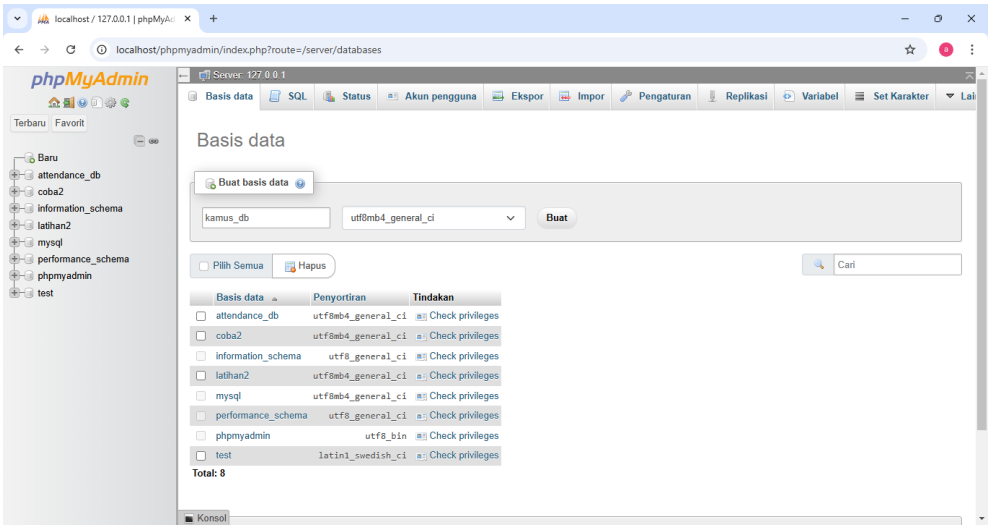
1) Aktifkan terlebih dahulu MySql dan Apache yang ada di Xampp



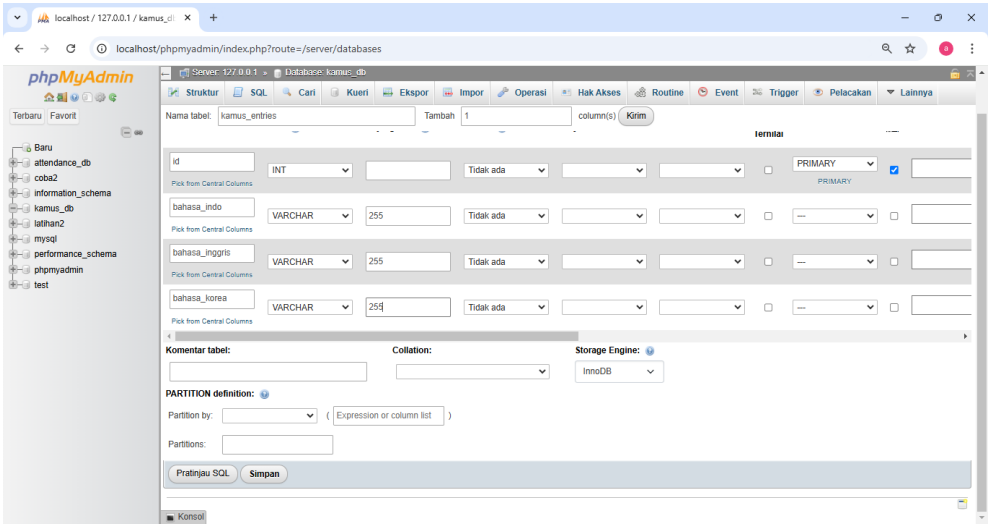
2) Kemudian ketikkan http://localhost/phpmyadmin/ pada peramban.



3) Buat database baru di MySQL bernama kamus_db



4) Buat table pada database tersebut dengan nama kamus_entries dengan struktur table sebagai berikut :



5) Selanjutnya, buatlah file PHP untuk menangani permintaan CRUD. Simpan file tersebut pada C:\xampp\htdocs\Kamus

6) Berikut adalah struktur file PHP untuk setiap operasi:

a. Koneksi ke Database (db.php):

```
<?php
$servername = "localhost";  // Ganti dengan nama server MySQL Anda
$username = "root";         // Ganti dengan username MySQL Anda
$password = "";             // Ganti dengan password MySQL Anda
$dbname = "kamus_db";       // Nama database yang Anda gunakan
```

```
// Membuat koneksi ke database
$conn = new mysqli($servername, $username, $password, $dbname);

// Memeriksa koneksi
if ($conn->connect_error) {
    die("Koneksi gagal: " . $conn->connect_error);
}
?>
```

b. Tambah Entri (create_entry.php):

```
<?php
// Koneksi ke database
include_once 'db.php';

// Menerima data dari request POST
$data = json_decode(file_get_contents("php://input"));

if (isset($data->bahasaIndo) && isset($data->bahasaInggris) && isset($data->bahasaKorea)) {
    $bahasaIndo = $data->bahasaIndo;
    $bahasaInggris = $data->bahasaInggris;
    $bahasaKorea = $data->bahasaKorea;

    // Query SQL untuk menyimpan data
    $query = "INSERT INTO kamus_entries (bahasa_indo, bahasa_inggris, bahasa_korea) VALUES (?, ?, ?)";
    $stmt = $conn->prepare($query);

    // Mengikat parameter ke statement SQL
    $stmt->bind_param("sss", $bahasaIndo, $bahasaInggris, $bahasaKorea);

    // Mengeksekusi query
    if ($stmt->execute()) {
        echo json_encode(array("message" => "Data berhasil ditambahkan."));
```

```
    } else {
        echo json_encode(array("message" => "Gagal menambahkan data."));
    }
} else {
    echo json_encode(array("message" => "Data tidak lengkap."));
}
?>
```

c. Ambil Semua Entri (read_entries.php):

```
<?php
require_once 'db.php';

$sql = "SELECT * FROM kamus_entries";
$result = $conn->query($sql);

$entries = array();

if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        $entry = array(
            "id" => $row["id"],
            "bahasa_indo" => $row["bahasa_indo"],
            "bahasa_inggris" => $row["bahasa_inggris"],
            "bahasa_korea" => $row["bahasa_korea"]
        );
        array_push($entries, $entry);
    }
    echo json_encode($entries);
} else {
    echo json_encode(array());
}

$conn->close();
?>
```

d. Perbarui Entri (update_entry.php):

```
<?php
include 'db.php';

$data = json_decode(file_get_contents("php://input"));

if (isset($data->id) && isset($data->bahasa_indo) && isset($data->bahasa_inggris)
&& isset($data->bahasa_korea)) {
    $id = $data->id;
    $bahasaIndo = $data->bahasa_indo;
    $bahasaInggris = $data->bahasa_inggris;
    $bahasaKorea = $data->bahasa_korea;

    $query = "UPDATE kamus_entries SET bahasa_indo = ?, bahasa_inggris = ?,
bahasa_korea = ? WHERE id = ?";
    $stmt = $conn->prepare($query);
    $stmt->bind_param("sssi", $bahasaIndo, $bahasaInggris, $bahasaKorea, $id);

    if ($stmt->execute()) {
        echo json_encode(["message" => "Data berhasil diperbarui."]);
    } else {
        echo json_encode(["message" => "Gagal memperbarui data."]);
    }
} else {
    echo json_encode(["message" => "Data tidak lengkap."]);
}
?>
```

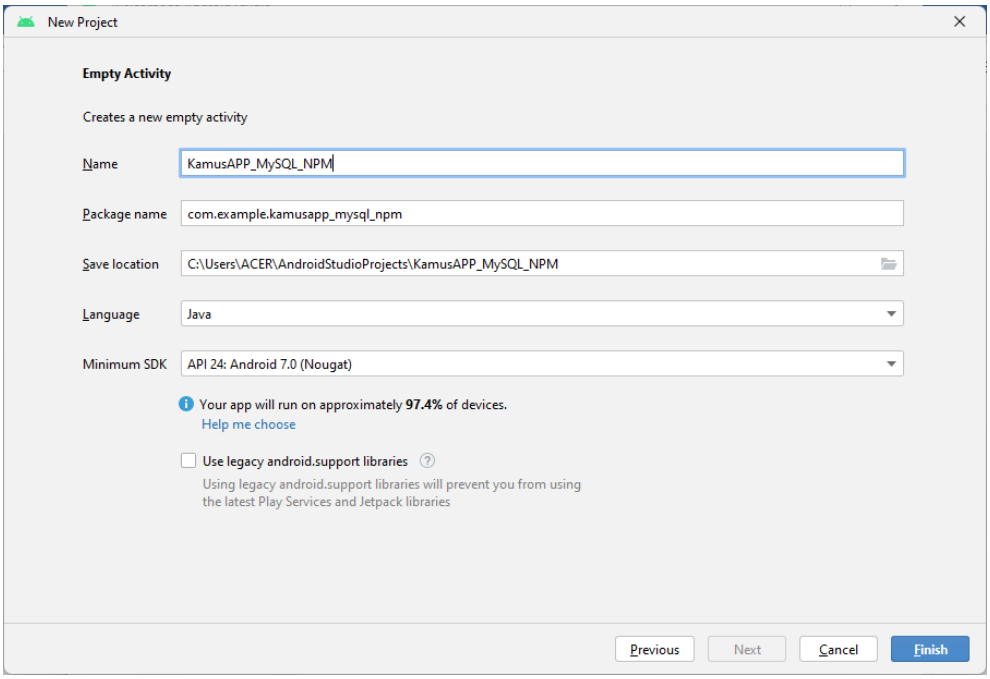
e. Hapus Entri (delete_entry.php):

```
<?php
include 'db.php'; // Pastikan file ini terhubung dengan database

// Pastikan request yang datang adalah DELETE
if ($_SERVER['REQUEST_METHOD'] === 'DELETE') {
```

```
    // Membaca input JSON
    $data = json_decode(file_get_contents("php://input"));

    // Cek apakah ID ada dalam request
    if (isset($data->id)) {
        $id = $data->id;
    } else {
        echo json_encode(["success" => false, "message" => "ID tidak ditemukan."]);
        exit();
    }

    // Debug: Tampilkan ID yang diterima
    error_log("ID yang diterima untuk penghapusan: " . $id);

    // Cek apakah ID ada di database
    $query_check = "SELECT * FROM kamus_entries WHERE id = ?";
    $stmt_check = $conn->prepare($query_check);
    $stmt_check->bind_param("i", $id);
    $stmt_check->execute();
    $result_check = $stmt_check->get_result();

    if ($result_check->num_rows > 0) {
        // Data ditemukan, lakukan penghapusan
        $query_delete = "DELETE FROM kamus_entries WHERE id = ?";
        $stmt_delete = $conn->prepare($query_delete);
        $stmt_delete->bind_param("i", $id);
        $stmt_delete->execute();

        if ($stmt_delete->affected_rows > 0) {
            echo json_encode(["success" => true, "message" => "Data berhasil
dihapus."]);
        } else {
            echo json_encode(["success" => false, "message" => "Gagal menghapus
data."]);
        }
```

```
    } else {
        // Data dengan ID tersebut tidak ditemukan
        echo json_encode(["success" => false, "message" => "Data dengan ID tersebut tidak ditemukan."]);
    }

    // Menutup statement
    $stmt_check->close();
    $stmt_delete->close();
} else {
    echo json_encode(["success" => false, "message" => "Metode request tidak valid."]);
}
?>
```

7) Buat projek baru pada android studio dengan nama KamusApp_MySql_NPMMasing-masing.



8) Tambahkan dependensi Retrofit di build.gradle:

```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.12.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation 'com.squareup.retrofit2:retrofit:2.9.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
    implementation 'com.squareup.okhttp3:okhttp:4.9.0'
    implementation 'com.squareup.okhttp3:logging-interceptor:4.9.0'
    implementation 'androidx.recyclerview:recyclerview:1.2.1'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.2.1'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.6.1'
}
```

Klik Sync Now untuk mengunduh library.

9) Membuat Model Data (KamusEntry.java)

```
package com.example.kamusapp_mysql_npm;

import com.google.gson.annotations.SerializedName;

public class KamusEntry {
    @SerializedName("id")
    private String id;

    @SerializedName("bahasa_indo")
    private String bahasaIndo;

    @SerializedName("bahasa_inggris")
    private String bahasaInggris;

    @SerializedName("bahasa_korea")
    private String bahasaKorea;

    // Constructor
    public KamusEntry(String id, String bahasaIndo, String
bahasaInggris, String bahasaKorea) {
        this.id = id;
        this.bahasaIndo = bahasaIndo;
        this.bahasaInggris = bahasaInggris;
        this.bahasaKorea = bahasaKorea;
    }

    // Getter and Setter
    public String getId() {
        return id;
    }

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

    public String getBahasaIndo() {
```

```
        return bahasaIndo;
    }

    public void setBahasaIndo(String bahasaIndo) {
        this.bahasaIndo = bahasaIndo;
    }

    public String getBahasaInggris() {
        return bahasaInggris;
    }

    public void setBahasaInggris(String bahasaInggris) {
        this.bahasaInggris = bahasaInggris;
    }

    public String getBahasaKorea() {
        return bahasaKorea;
    }

    public void setBahasaKorea(String bahasaKorea) {
        this.bahasaKorea = bahasaKorea;
    }
}
```

10) Menambahkan Retrofit

ApiService.java

```
package com.example.kamusapp_mysql_npm;

import retrofit2.Call;
import retrofit2.http.*;

import java.util.List;

public interface ApiService {
    // Mendapatkan semua data kamus
    @GET("read_entries.php")
    Call<List<KamusEntry>> getAllEntries();

    // Menambahkan data baru ke kamus
    @POST("create_entry.php")
    Call<KamusEntry> createEntry(@Body KamusEntry entry);

    // Memperbarui data kamus yang dipilih
    @PUT("update_entry.php/{id}")
    Call<Void> updateEntry(@Path("id") int id, @Body
KamusEntry entry);

    // Menghapus data kamus berdasarkan ID
    @DELETE("delete_entry.php/{id}")
    Call<Void> deleteEntry(@Path("id") int id);
}
```

RetrofitClient.java

```
package com.example.kamusapp_mysql_npm;

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

public class RetrofitClient {
    private static final String BASE_URL =
"http://10.0.2.2/kamus/";
    private static Retrofit retrofit;

    public static Retrofit getInstance() {
        if (retrofit == null) {
            // Menambahkan logging interceptor untuk debug
            HttpLoggingInterceptor logging = new
HttpLoggingInterceptor();

logging.setLevel(HttpLoggingInterceptor.Level.BODY);

            OkHttpClient client = new OkHttpClient.Builder()
                    .addInterceptor(logging)
                    .build();

            // Membangun Retrofit instance
            retrofit = new Retrofit.Builder()
                    .baseUrl(BASE_URL)
                    .client(client) // Tambahkan OkHttpClient
dengan logging

.addConverterFactory(GsonConverterFactory.create())
                    .build();
        }
        return retrofit;
    }
}
```

11) Buat Layout RecyclerView (kamus_item.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
```

```
        android:id="@+id/txtIndo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Bahasa Indonesia"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/txtInggris"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Bahasa Inggris" />

    <TextView
        android:id="@+id/txtKorea"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Bahasa Korea" />
</LinearLayout>
```

12) Membuat Adapter RecyclerView (KamusAdapter.java)

```
package com.example.kamusapp_mysql_npm;

import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.recyclerview.widget.RecyclerView;

import java.util.List;

public class KamusAdapter extends
RecyclerView.Adapter<KamusAdapter.ViewHolder> {
    private List<KamusEntry> entries;
    private OnItemClickListener onItemClickListener;

    public interface OnItemClickListener {
        void onItemClick(KamusEntry entry);
    }

    public KamusAdapter(List<KamusEntry> entries,
OnItemClickListener onItemClickListener) {
        this.entries = entries;
        this.onItemClickListener = onItemClickListener;
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
```

```
        View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.kamu
s_item, parent, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(ViewHolder holder, int
position) {
        KamusEntry entry = entries.get(position);

        // Set data ke ViewHolder
        holder.bahasaIndo.setText(entry.getBahasaIndo());

holder.bahasaInggris.setText(entry.getBahasaInggris());
        holder.bahasaKorea.setText(entry.getBahasaKorea());

        // Debugging
        //Log.d("KamusAdapter", "Binding data: " +
entry.getBahasaIndo() + ", " + entry.getBahasaInggris() + ", "
+ entry.getBahasaKorea());

        holder.itemView.setOnClickListener(v -> {
            if (onItemClickListener != null) {
                onItemClickListener.onItemClick(entry);
            }
        });
    }


    @Override
    public int getItemCount() {
        return entries.size();
    }

    public class ViewHolder extends RecyclerView.ViewHolder {
        TextView bahasaIndo, bahasaInggris, bahasaKorea;

        public ViewHolder(View itemView) {
            super(itemView);
            bahasaIndo = itemView.findViewById(R.id.txtIndo);
            bahasaInggris =
itemView.findViewById(R.id.txtInggris);
            bahasaKorea =
itemView.findViewById(R.id.txtKorea);
        }
    }
}
```

13) Ubah koding pada MainActivity.java hingga seperti berikut :

```
package com.example.kamusapp_mysql_npm;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.*;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private RecyclerView recyclerView;
    private KamusAdapter adapter;
    private List<KamusEntry> entries = new ArrayList<>();
    private EditText inputIndo, inputInggris, inputKorea;
    private Button btnAdd, btnUpdate, btnDelete;
    private ProgressBar progressBar;
    private ApiService apiService;
    private KamusEntry selectedEntry;

    private boolean isDestroyed = false; // Flag untuk
lifecycle

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

        // Inisialisasi Views
        recyclerView = findViewById(R.id.recyclerView);
        inputIndo = findViewById(R.id.inputIndo);
        inputInggris = findViewById(R.id.inputInggris);
        inputKorea = findViewById(R.id.inputKorea);
        btnAdd = findViewById(R.id.btnAdd);
        btnUpdate = findViewById(R.id.btnUpdate);
        btnDelete = findViewById(R.id.btnDelete);
        progressBar = findViewById(R.id.progressBar);

        // Konfigurasi RecyclerView
        recyclerView.setLayoutManager(new
LinearLayoutManager(this));
        adapter = new KamusAdapter(entries, entry -> {
            // Isi data ke EditText ketika item diklik
            selectedEntry = entry;
            inputIndo.setText(entry.getBahasaIndo());
```

```
            inputInggris.setText(entry.getBahasaInggris());
            inputKorea.setText(entry.getBahasaKorea());
        });
        recyclerView.setAdapter(adapter);

        // Inisialisasi Retrofit API
        apiService =
RetrofitClient.getInstance().create(ApiService.class);

        // Memuat data dari API
        loadEntries();

        // Tombol Add, Update, Delete
        btnAdd.setOnClickListener(view -> addEntry());
        btnUpdate.setOnClickListener(view -> updateEntry());
        btnDelete.setOnClickListener(view ->
confirmDeleteEntry());
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        isDestroyed = true; // Tandai bahwa activity telah
dihancurkan
        Log.d("MainActivity", "onDestroy called");
    }

    private void loadEntries() {
        showLoading(true);
        apiService.getAllEntries().enqueue(new
Callback<List<KamusEntry>>() {
            @Override
            public void onResponse(Call<List<KamusEntry>>
call, Response<List<KamusEntry>> response) {
                if (isDestroyed) return; // Jangan lanjutkan
jika activity sudah dihancurkan

                showLoading(false);
                if (response.isSuccessful() && response.body()
!= null) {
                    entries.clear();
                    entries.addAll(response.body());
                    adapter.notifyDataSetChanged();
                } else {
                    showToast("Gagal memuat data!");
                }
            }

            @Override
            public void onFailure(Call<List<KamusEntry>> call,
Throwable t) {
                if (isDestroyed) return; // Jangan lanjutkan
jika activity sudah dihancurkan
```

```
                showLoading(false);
                showToast("Error: Tidak bisa mengakses API!");
            }
        });
    }

    private void addEntry() {
        String indo = inputIndo.getText().toString().trim();
        String inggris =
inputInggris.getText().toString().trim();
        String korea = inputKorea.getText().toString().trim();

        if (indo.isEmpty() || inggris.isEmpty() ||
korea.isEmpty()) {
            showToast("Semua field harus diisi!");
            return;
        }

        showLoading(true);
        KamusEntry newEntry = new KamusEntry("", indo,
inggris, korea);

        apiService.createEntry(newEntry).enqueue(new
Callback<KamusEntry>() {
            @Override
            public void onResponse(Call<KamusEntry> call,
Response<KamusEntry> response) {
                if (isDestroyed) return;

                showLoading(false);
                if (response.isSuccessful() && response.body()
!= null) {
                    entries.add(response.body());
                    adapter.notifyDataSetChanged();
                    clearInputFields();
                    showToast("Berhasil menambahkan data!");
                } else {
                    showToast("Gagal menambahkan data!");
                }
            }

            @Override
            public void onFailure(Call<KamusEntry> call,
Throwable t) {
                if (isDestroyed) return;

                showLoading(false);
                showToast("Error: " + t.getMessage());
            }
        });
    }
```

```
    private void updateEntry() {
        if (selectedEntry == null) {
            showToast("Pilih data untuk diperbarui!");
            return;
        }

        String indo = inputIndo.getText().toString().trim();
        String inggris =
inputInggris.getText().toString().trim();
        String korea = inputKorea.getText().toString().trim();

        if (indo.isEmpty() || inggris.isEmpty() ||
korea.isEmpty()) {
            showToast("Semua field harus diisi!");
            return;
        }

        selectedEntry.setBahasaIndo(indo);
        selectedEntry.setBahasaInggris(inggris);
        selectedEntry.setBahasaKorea(korea);

        showLoading(true);


apiService.updateEntry(Integer.parseInt(selectedEntry.getId())
, selectedEntry).enqueue(new Callback<Void>() {
            @Override
            public void onResponse(Call<Void> call,
Response<Void> response) {
                if (isDestroyed) return;

                showLoading(false);
                if (response.isSuccessful()) {
                    loadEntries();
                    clearInputFields();
                    selectedEntry = null;
                    showToast("Berhasil memperbarui data!");
                } else {
                    showToast("Gagal memperbarui data!");
                }
            }

            @Override
            public void onFailure(Call<Void> call, Throwable
t) {
                if (isDestroyed) return;

                showLoading(false);
                showToast("Error: " + t.getMessage());
            }
        });
    }
```

```
    private void confirmDeleteEntry() {
        if (selectedEntry == null) {
            showToast("Pilih data untuk dihapus!");
            return;
        }

        if (!isFinishing() && !isDestroyed) {
            new AlertDialog.Builder(this)
                    .setTitle("Hapus Data")
                    .setMessage("Apakah Anda yakin ingin
menghapus data ini?")
                    .setPositiveButton("Ya", (dialog, which) -
> deleteEntry())
                    .setNegativeButton("Tidak", null)
                    .show();
        }
    }

    private void deleteEntry() {
        if (selectedEntry == null) {
            showToast("Pilih data untuk dihapus!");
            return;
        }

        String id = selectedEntry.getId();

        showLoading(true);


apiService.deleteEntry(Integer.parseInt(id)).enqueue(new
Callback<Void>() {
            @Override
            public void onResponse(Call<Void> call,
Response<Void> response) {
                if (isDestroyed) return;

                showLoading(false);
                if (response.isSuccessful()) {
                    loadEntries();
                    selectedEntry = null;
                    showToast("Data berhasil dihapus!");
                } else {
                    showToast("Gagal menghapus data!");
                }
            }

            @Override
            public void onFailure(Call<Void> call, Throwable
t) {
                if (isDestroyed) return;

                showLoading(false);
                showToast("Error: " + t.getMessage());
```

```
            }
        });
    }

    private void clearInputFields() {
        inputIndo.setText("");
        inputInggris.setText("");
        inputKorea.setText("");
    }

    private void showLoading(boolean isLoading) {
        runOnUiThread(() ->
progressBar.setVisibility(isLoading ? View.VISIBLE :
View.GONE));
    }

    private void showToast(String message) {
        if (!isFinishing() && !isDestroyed()) {
            runOnUiThread(() -> Toast.makeText(this, message,
Toast.LENGTH_SHORT).show());
        }
    }
}
```

14) Buat Layout Activity (activity_main.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <!-- Input Fields -->
    <EditText
        android:id="@+id/inputIndo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Bahasa Indonesia" />

    <EditText
        android:id="@+id/inputInggris"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Bahasa Inggris" />

    <EditText
        android:id="@+id/inputKorea"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:hint="Bahasa Korea" />

    <!-- Buttons -->
    <Button
        android:id="@+id/btnAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tambah Data" />

    <Button
        android:id="@+id/btnUpdate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Perbarui Data" />

    <Button
        android:id="@+id/btnDelete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hapus Data" />

    <ProgressBar
        android:id="@+id/progressBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:visibility="gone" />

    <!-- RecyclerView untuk menampilkan data -->
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

15) Pada folder XML tambahkan file network_security_config.xml

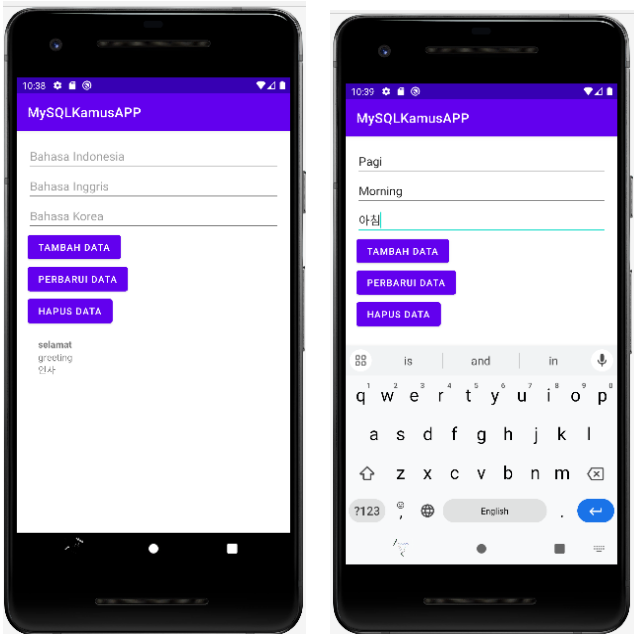
```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">10.0.2.2</domain>
    </domain-config>
</network-security-config>
```

16) Ubah file AndoirdManifest.xml menjadi seperti berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission
```

```
android:name="android.permission.INTERNET" />

    <application
        android:usesCleartextTraffic="true"

android:networkSecurityConfig="@xml/network_security_config"
        android:allowBackup="true"

android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.KamusAPP_MySQL_NPM"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

17) Jalankan project pada emulator hingga muncul output sebagai berikut :



12.5 TUGAS

1. Demokan project tersebut hingga muncul di emulator!
2. Tambahkan minimal 5 kosakata baru ke dalam kamus yang sudah ada. Pastikan kosakata tersebut mencakup terjemahan untuk setiap kombinasi bahasa (Inggris-Indonesia, Indonesia-Korea, dll.).
3. Tugas dikumpul dalam format pdf dengan nama file : NPM_Tugaske?
4. Didalam file tersebu dituliskan :
   - NPM
   - Nama Lengkap
   - Program Studi
   - Kelas