**LAPORAN PROYEK UAS**

**“** **Aplikasi Mobile Laboratorium Komputer Politeknik Statistika STIS”**

****

**Dosen Pengampu:**

|  |
| --- |
| Ibnu Santoso, S.S.T., M.T |

**Disusun oleh:**

|  |  |
| --- | --- |
| Blessy Munthia Purba | (222212537) (3SI2) |

**Kamis, 31 Januari 2025**

**PROGRAM STUDI D-IV KOMPUTASI STATISTIK**

**POLITEKNIK STATISTIKA STIS**

**2025**

**Soal:**  
Buatlah aplikasi mobile dengan spesifikasi sebagai berikut.

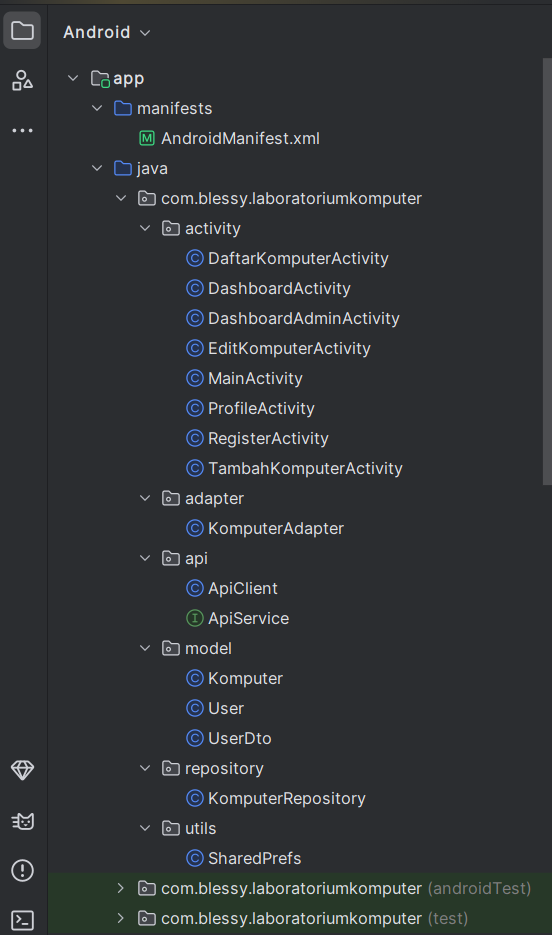
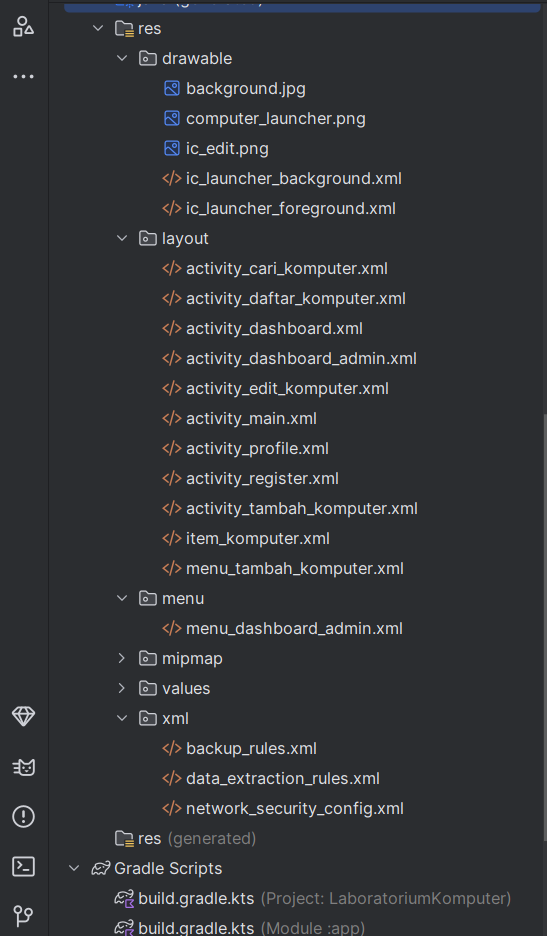
1. Pengguna aplikasi dapat melakukan register dan login serta dapat mengubah profil pengguna.
2. Tambahkan fitur lainnya sesuai dengan topik Anda minimal 3 activity/tampilan layar.

**Ketentuan:**

1. Aplikasi mobile yang dibuat harus menggunakan/mengkonsumsi API dari webservice. Anda dapat menggunakan webservice yang telah Anda kerjakan pada Ujian Tengah Semester (UTS) atau dengan penyempurnaan.
2. Aplikasi dapat dibangun menggunakan native Android atau hybrid/cross-platform native.
3. Kumpulkan hasil pekerjaan Anda dalam laporan berformat PDF yang memuat informasi sebagai berikut:
4. Screenshot setiap tampilan aplikasi yang disertai dengan penjelasan penggunaannya dan penjelasan kode program yang terkait.
5. Tautan video capture aplikasi Anda yang menunjukkan semua fitur berjalan sesuai ketentuan.
6. link git

**Tampilan Aplikasi dengan Penjelasan Kode Program:**

1. *Directory Kode Program:*

** **

1. *Penjelasan Singkat Keperluan Layanan Aplikasi*

Web Service dan Aplikasi Mobile Lab-Komputer Polstat STIS ini dibuat untuk keperluan monitoring perangkat elektronik (komputer) di Polstat STIS, yang akan menyediakan informasi dasar seperti IP Address, kapan dibelinya perangkat, masih bergaransi apa tidak, kapan terakhir pemeliharaan perangkat, dan kondisinya masih baik/rusak/perlu pemeliharaan. Di Aplikasi yang saya buat ini dapat melakukan register terlebih dahulu jika belum memiliki akun, yang digunakan oleh 3 role yaitu:

1). Admin: Pegawai/Staf IT di Polstat STIS

2). Mahasiswa : Angkatan 63-66 Polstat STIS

3). Pengamat: Pejabat/Petinggi di Polstat STIS

1. *Source Code ApiClient dan ApiService di package .api*
2. *ApiClient*

package com.blessy.laboratoriumkomputer.api;  
  
import com.google.gson.Gson;  
import com.google.gson.GsonBuilder;  
  
import retrofit2.Retrofit;  
import retrofit2.converter.gson.GsonConverterFactory;  
  
public class ApiClient {  
 private static final String *BASE\_URL* = "http://192.168.0.114:8088/";  
 private static Retrofit *retrofit*;  
  
 // Menyediakan instance Retrofit  
 public static Retrofit getClient() {  
 if (*retrofit* == null) {  
 Gson gson = new GsonBuilder()  
 .setLenient() // Mengizinkan parsing JSON lebih longgar  
 .create();  
  
 *retrofit* = new Retrofit.Builder()  
 .baseUrl(*BASE\_URL*)  
 .addConverterFactory(GsonConverterFactory.*create*(gson))  
 .build();  
 }  
 return *retrofit*;  
 }  
  
 // Menyediakan ApiService untuk digunakan di bagian lain aplikasi  
 public static ApiService getApiService() {  
 return *getClient*().create(ApiService.class);  
 }  
  
}

Pada kode di atas, Retrofit digunakan untuk melakukan permintaan HTTP dan menangani JSON. BASE\_URL adalah URL dasar tempat semua API dimulai. Karena masih memakai ipAddress (localhost laptop), maka API Web Service harus dijalankan dahulu untuk menjalankan Aplikasi Mobile. Gson digunakan untuk mengonversi data antara objek Java dan JSON. Sedangkan GsonConverterFactory memberi Retrofit kemampuan untuk mengonversi JSON menggunakan Gson. Sementara ApiClient adalah kelas untuk mengelola instance Retrofit dan menyediakan layanan API yang dapat digunakan di tempat lain dalam aplikasi.

1. *ApiService*

package com.blessy.laboratoriumkomputer.api;  
  
import com.blessy.laboratoriumkomputer.model.Komputer;  
import com.blessy.laboratoriumkomputer.model.User;  
import java.util.List;  
import okhttp3.ResponseBody;  
import retrofit2.Call;  
import retrofit2.http.Body;  
import retrofit2.http.DELETE;  
import retrofit2.http.GET;  
import retrofit2.http.Header;  
import retrofit2.http.POST;  
import retrofit2.http.PUT;  
import retrofit2.http.Path;  
import retrofit2.http.Query;  
  
public interface ApiService {  
 @POST("auth/register")  
 Call<ResponseBody> register(@Body User user);  
  
 @POST("auth/login")  
 Call<ResponseBody> login(@Body User user);  
  
 @GET("komputer")  
 Call<List<Komputer>> getKomputer(@Header("Authorization") String token);  
  
 @POST("komputer") // Sesuaikan dengan endpoint yang sesuai di server Anda  
 Call<ResponseBody> tambahKomputer(@Header("Authorization") String token, @Body Komputer komputer);  
  
 @GET("komputer/{id}")  
 Call<Komputer> getKomputerById(  
 @Header("Authorization") String token, // Token untuk otentikasi  
 @Path("id") Long id // ID alumni yang akan diambil  
 );  
  
 @PUT("komputer/{id}")  
 Call<String> editKomputer(  
 @Header("Authorization") String token,  
 @Path("id") Long id,  
 @Body Komputer komputer  
 );  
  
 @GET("komputer/cari")  
 Call<List<Komputer>> cariKomputer(@Header("Authorization") String authorization, @Query("keyword") String keyword);  
  
 @GET("auth/profile")  
 Call<User> getProfile(@Header("Authorization") String token); // Menambahkan header untuk token  
  
 @PUT("auth/profile/edit")  
 Call<User> editProfile(@Header("Authorization") String token, @Body User user);  
  
 @DELETE("auth/profile/delete")  
 Call<Void> deleteAccount(@Header("Authorization") String token, @Body User user);  
}

1. *Source Code SharedPrefs di package .utils*

package com.blessy.laboratoriumkomputer.utils;  
  
import android.content.Context;  
import android.content.SharedPreferences;  
  
public class SharedPrefs {  
 private final SharedPreferences prefs;  
  
 // Konstruktor yang menerima context untuk akses SharedPreferences  
 public SharedPrefs(Context context) {  
 // Menyimpan SharedPreferences dengan nama 'user\_prefs' dan mode PRIVATE  
 this.prefs = context.getSharedPreferences("user\_prefs", Context.*MODE\_PRIVATE*);  
 }  
  
 // Menyimpan role pengguna (misalnya admin, user, dll)  
 public void saveUserRole(String role) {  
 SharedPreferences.Editor editor = prefs.edit();  
 editor.putString("role", role);  
 editor.apply(); // Gunakan apply() agar tidak terjadi blocking operasi  
 }  
  
 // Mendapatkan role pengguna, default null jika tidak ditemukan  
 public String getUserRole() {  
 return prefs.getString("role", null); // null sebagai default  
 }  
  
 // Menyimpan token autentikasi (misalnya token JWT)  
 public void saveToken(String token) {  
 SharedPreferences.Editor editor = prefs.edit();  
 editor.putString("auth\_token", token); // Menyimpan token dengan key 'auth\_token'  
 editor.apply(); // Gunakan apply() agar data tersimpan tanpa blocking UI thread  
 }  
  
 public void saveLoginStatus(boolean status) {  
 SharedPreferences.Editor editor = prefs.edit();  
 editor.putBoolean("is\_logged\_in", status);  
 editor.apply();  
 }  
  
  
  
 // Mendapatkan token autentikasi, jika tidak ada mengembalikan null  
 public String getToken() {  
 return prefs.getString("auth\_token", null); // Mengambil token, null jika tidak ditemukan  
 }  
  
 // Menyediakan method untuk memeriksa apakah token ada atau tidak  
 public boolean isTokenExists() {  
 return getToken() != null;  
 }  
}

1. *Source Code AndroidManifest.xml*

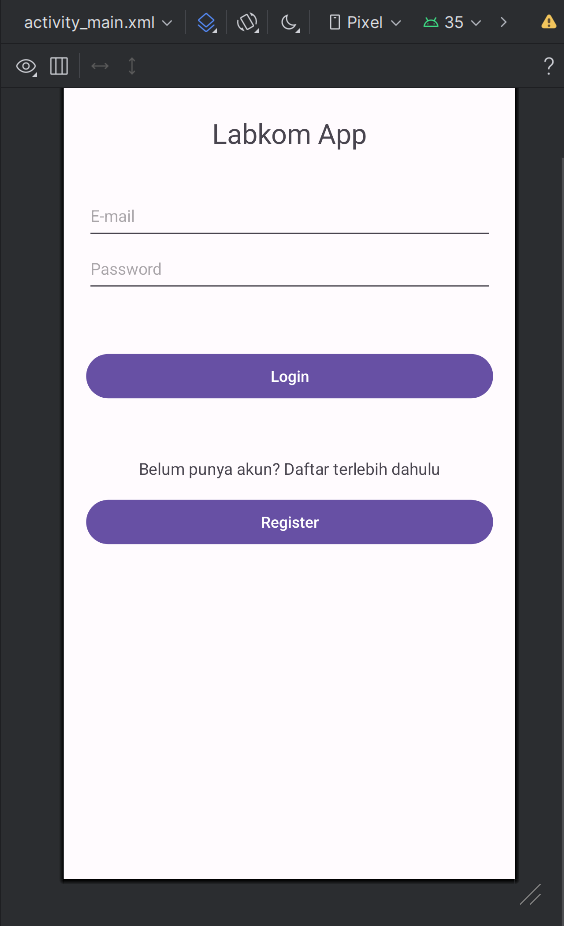
<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools">  
  
 <!-- Permission for accessing external storage -->  
 <uses-permission android:name="android.permission.INTERNET" />  
 <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
  
  
 <application  
 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:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.LaboratoriumKomputer"  
 tools:targetApi="31">  
 <activity  
 android:name=".activity.MainActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 <activity  
 android:name=".activity.DashboardAdminActivity"  
 android:label="Admin Dashboard"  
 android:exported="false" />  
 <activity  
 android:name=".activity.RegisterActivity"  
 android:label="Register"  
 android:exported="false" />  
  
 <activity  
 android:name=".activity.TambahKomputerActivity"  
 android:label="Tambah Data Komputer"  
 android:exported="false" />  
  
 <activity  
 android:name=".activity.EditKomputerActivity"  
 android:label="Edit Data Komputer"  
 android:exported="false" />  
  
 </application>  
  
</manifest>

1. *Source Code KomputerAdapter*

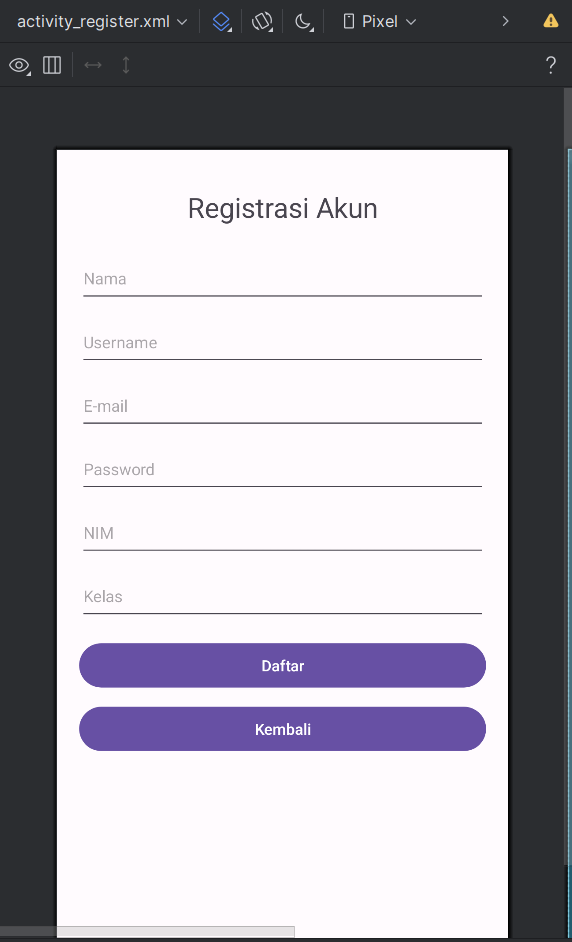
package com.blessy.laboratoriumkomputer.adapter;  
  
import android.content.Context;  
import android.content.Intent;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageView;  
import android.widget.TextView;  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
import com.blessy.laboratoriumkomputer.R;  
import com.blessy.laboratoriumkomputer.model.Komputer;  
import com.blessy.laboratoriumkomputer.activity.EditKomputerActivity;  
  
import java.util.List;  
  
public class KomputerAdapter extends RecyclerView.Adapter<KomputerAdapter.ViewHolder> {  
 private List<Komputer> komputerList;  
 private Context context;  
  
 // Constructor untuk menerima context  
 public KomputerAdapter(Context context) {  
 this.context = context;  
 }  
  
 // Setter untuk list data  
 public void setKomputerList(List<Komputer> komputerList) {  
 this.komputerList = komputerList;  
 notifyDataSetChanged(); // Notifikasi perubahan data  
 }  
  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*item\_komputer*, parent, false);  
 return new ViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull ViewHolder holder, int position) {  
 Komputer komputer = komputerList.get(position);  
 holder.ruangKomputer.setText(komputer.getRuangKomputer());  
 holder.ipAddress.setText(komputer.getIpAddress());  
  
 // Listener untuk tombol edit  
 holder.editKomputerButton.setOnClickListener(v -> {  
 Intent intent = new Intent(context, EditKomputerActivity.class);  
 intent.putExtra("id", komputer.getId()); // Kirim ID komputer  
 context.startActivity(intent);  
 });  
 }  
  
 @Override  
 public int getItemCount() {  
 return (komputerList != null) ? komputerList.size() : 0;  
 }  
  
 // ViewHolder untuk menghubungkan layout item dengan adapter  
 public static class ViewHolder extends RecyclerView.ViewHolder {  
 TextView ruangKomputer, ipAddress;  
 ImageView editKomputerButton;  
  
 public ViewHolder(@NonNull View itemView) {  
 super(itemView);  
 ruangKomputer = itemView.findViewById(R.id.*ruangKomputer*);  
 ipAddress = itemView.findViewById(R.id.*ipAddress*);  
 editKomputerButton = itemView.findViewById(R.id.*edit\_komputer\_button*); // Pastikan ID sesuai XML  
 }  
 }  
}

Secara keseluruhan, KomputerAdapter berfungsi untuk mengelola data dan menampilkan informasi komputer di RecyclerView, serta memungkinkan interaksi seperti membuka halaman untuk mengedit data komputer, menyambungkan data dengan tampilan layout, dan mengubah tampilan item.

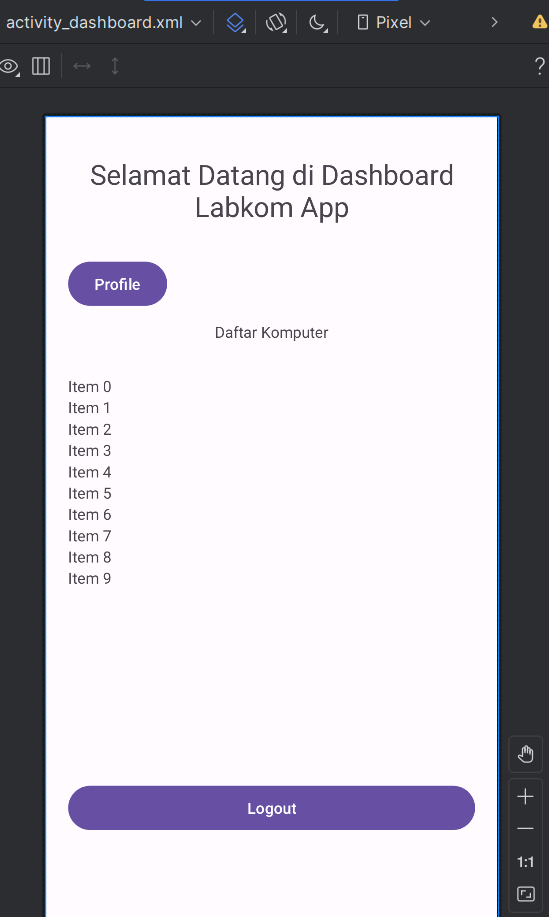
1. *Layout Tiap Activity*
2. *activity\_main.xml*

**

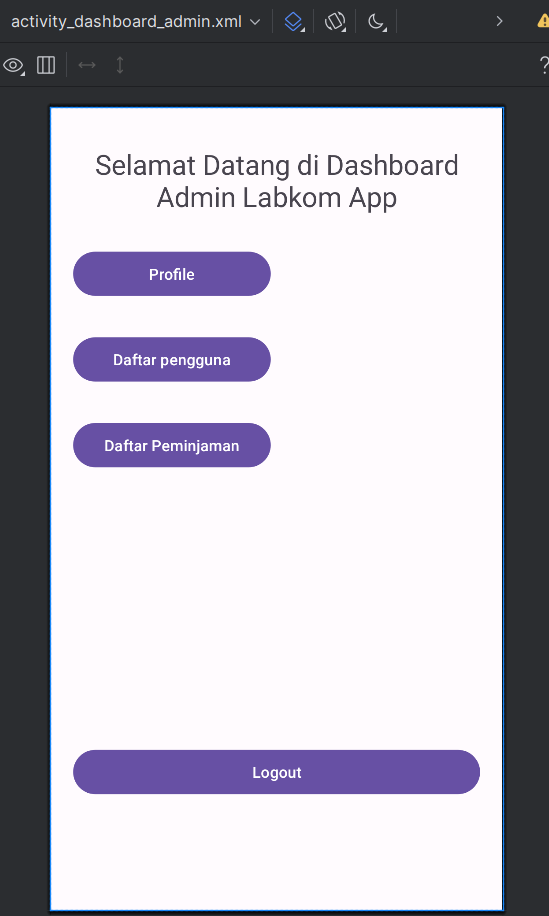
1. *activity\_register.xml*

**

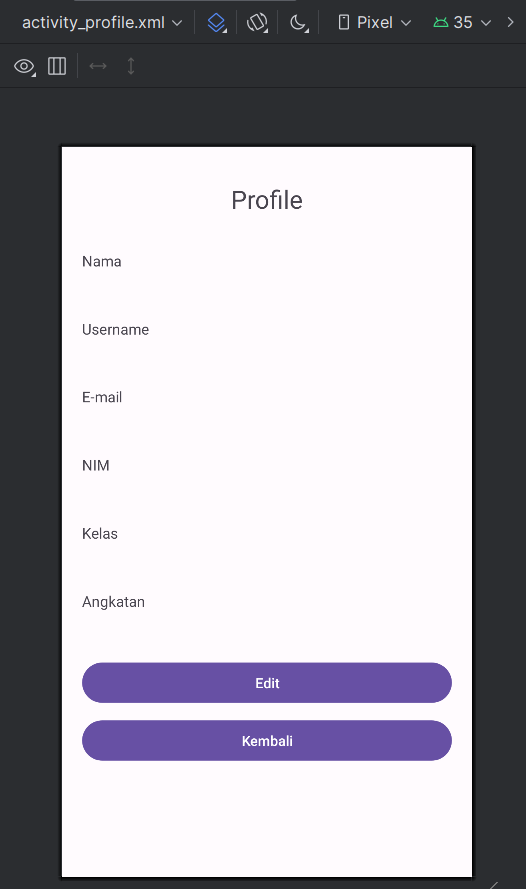
1. *activity\_dashboard.xml*

**

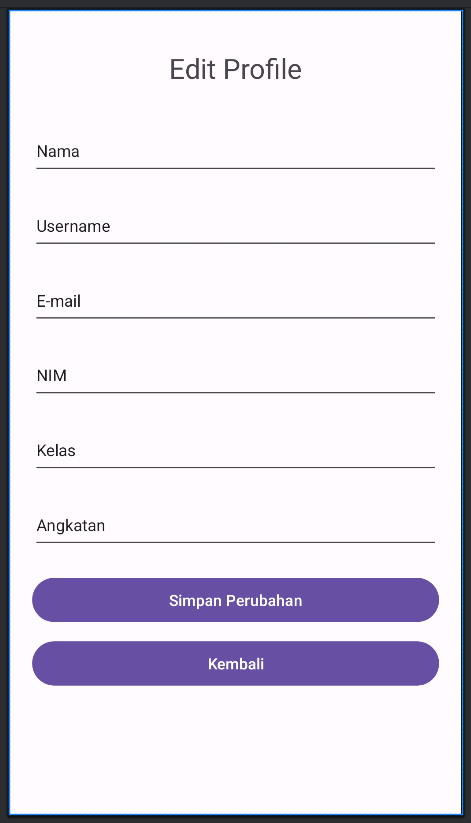
1. *activity\_dashboard\_admin.xml*

**

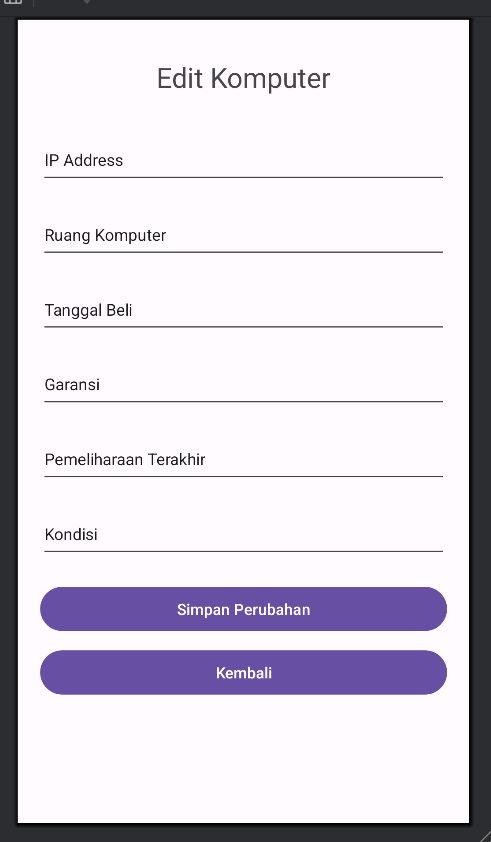
1. *activity\_profile.xml*

**

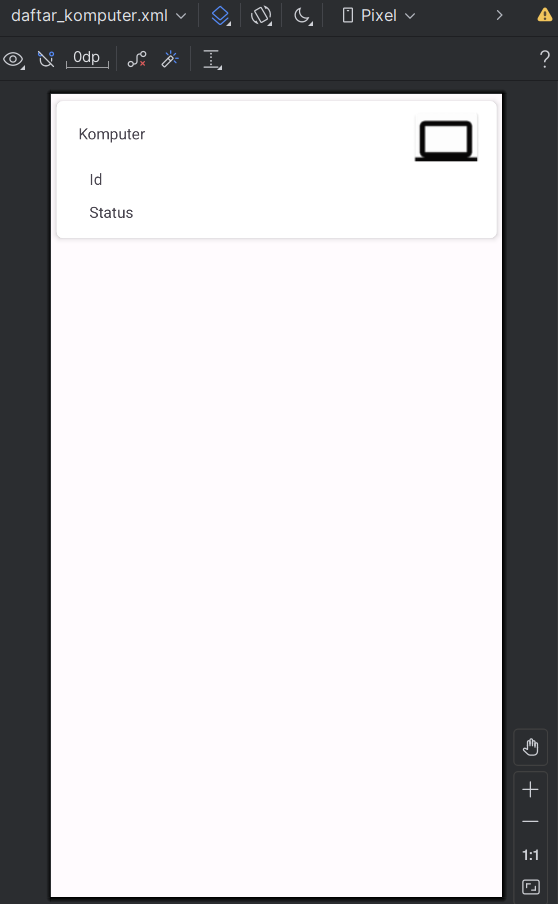
1. *activity\_edit\_profile.xml*

**

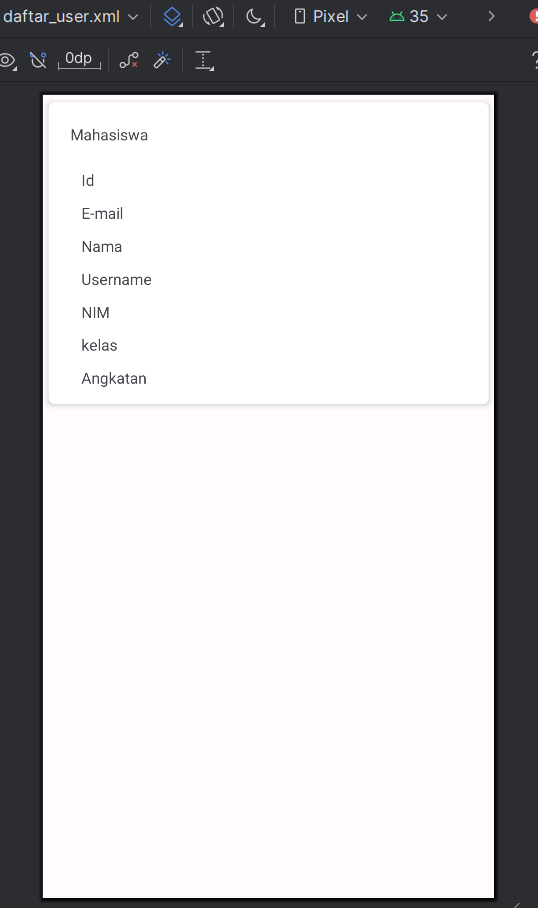
1. *activity\_edit\_komputer.xml*

**

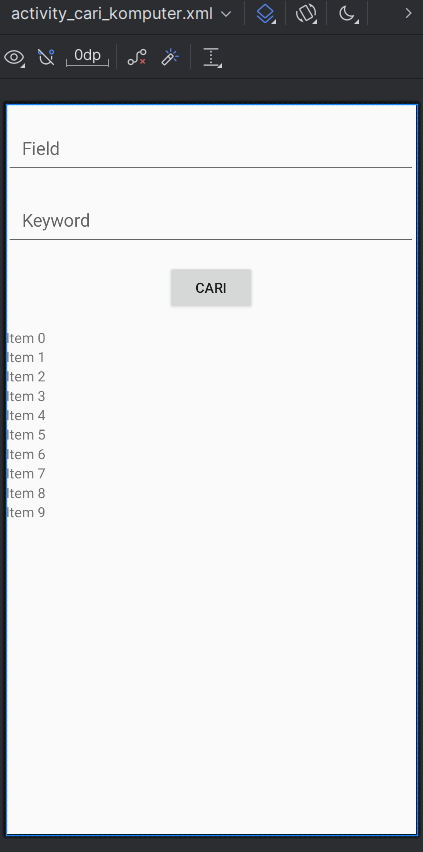
1. *activity\_daftar\_komputer.xml*

**

1. *daftar\_user.xml*

**

1. *activity\_cari\_komputer.xml*

**

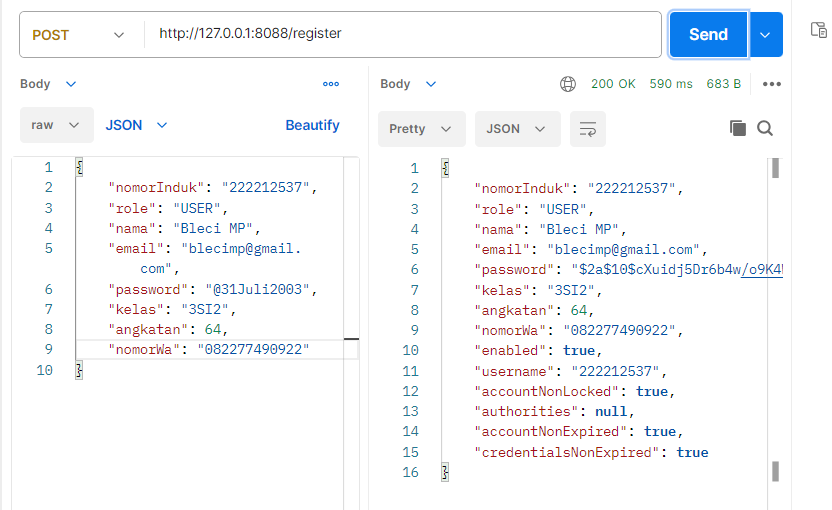
**Alur Penjelasan Video Capture Aplikasi:**

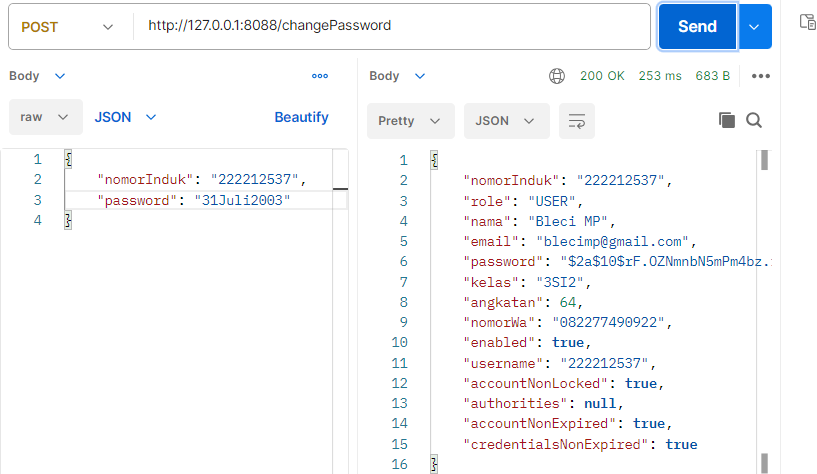
1. Menjelaskan entitas/model yang ada di aplikasi mobile lab-komputer
2. Menampilkan runing API Web Service di Postman sesuai method-method kegunaan Aplikasi Web Service
3. Menjelaskan sedikit penggunaan dan penjelasan kode program di Aplikasi Mobile
4. Menjalankan Aplikasi Mobile di Aplikasi Studio dan memperlihatkan runing yang masih eror.
5. Link tautan video: <https://s.stis.ac.id/VideoCaptureLabkom_blessy>

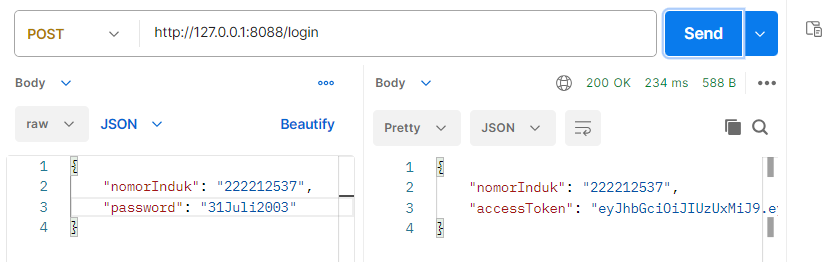
**Link Git:**

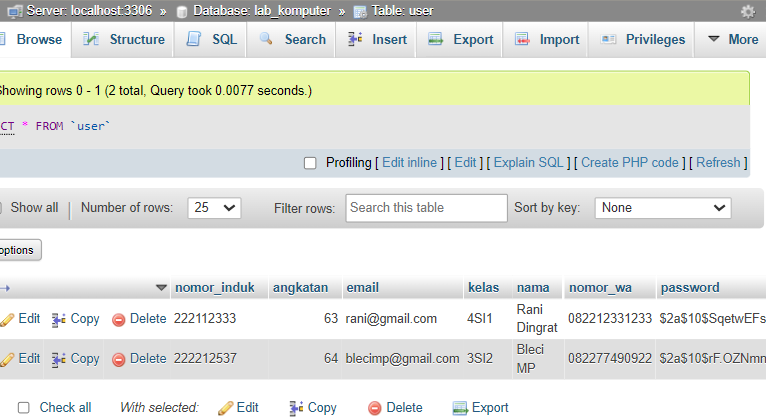
**Lampiran: Hasil Running Web Service API Postman dan Database**

1. Untuk Entitas/Model User

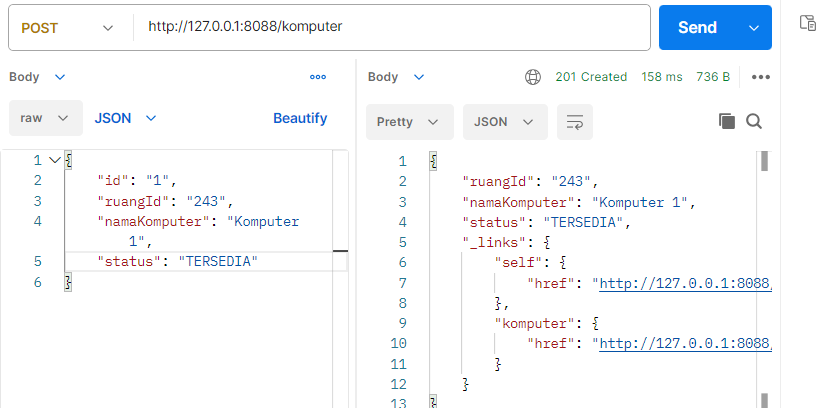
****

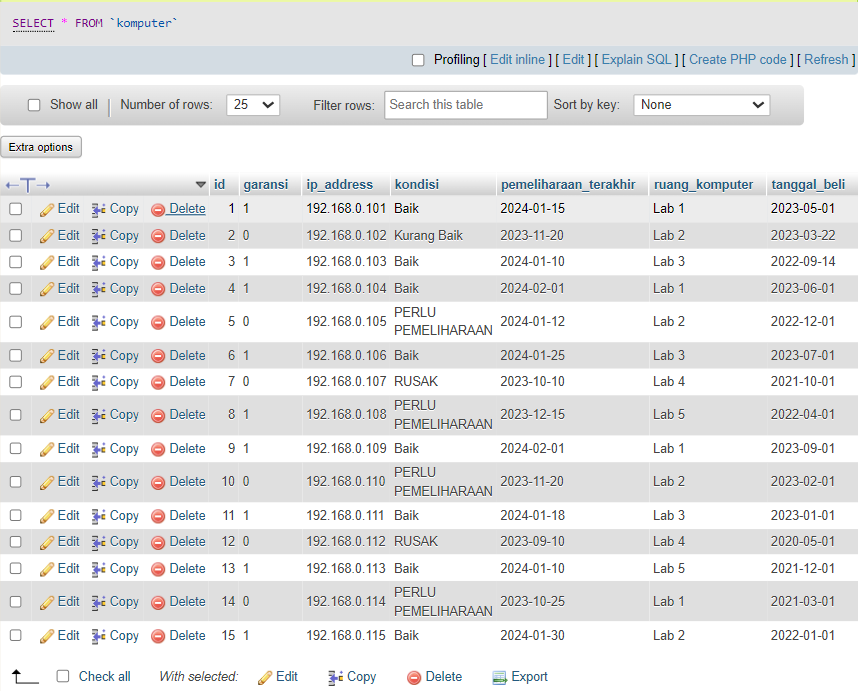
****

****

****

1. Untuk Entitas/Model Komputer

****

****