**APLIKASI POS DENGAN DAJNGO DAN FLUTTER DESKTOP**

* **Kita akan bangun Aplikasi POS menggunakan:**

**🔹 Flutter Desktop sebagai frontend  
🔹 Django sebagai backend  
🔹 MySQL sebagai database utama**

* **STRUKTUR BESAR YANG AKAN KITA BUAT:**

1. **Django Backend:**
   * **Setup project dan koneksi ke MySQL**
   * **Buat model User**
   * **Endpoint API untuk: daftar user, tambah user, login user**
   * **Gunakan Django REST Framework**
2. **Flutter Desktop Frontend:**
   * **Login Screen**
   * **User List Screen**
   * **Form Tambah/Edit User**
   * **Koneksi ke API Django via http package**

* **Bagian 1: Setup Django Backend**

1. **Buat virtual environment & install Django + DRF**

**Command:**

python -m venv env

source env/bin/activate # di Windows: env\Scripts\activate

pip install django djangorestframework mysqlclient

**Catatan: Kalau error mysqlclient, pastikan sudah install:**

* **Linux: sudo apt install libmysqlclient-dev**
* **Windows: install** [**MySQL Connector/C**](https://dev.mysql.com/downloads/connector/c/)

1. **Buat project dan app**

**Command:**

django-admin startproject pos\_backend

cd pos\_backend

python manage.py startapp userapp

**3. Setting settings.py**

* **Tambahkan 'rest\_framework' dan 'userapp' ke INSTALLED\_APPS (settings.py)**

INSTALLED\_APPS = [

.

.

.

'rest\_framework',

'userapp',

]

* **Ganti konfigurasi database (settings.py):**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'posdb',

'USER': 'root',

'PASSWORD': 'passwordkamu',

'HOST': 'localhost',

'PORT': '3306',

}

}

**4. Buat model user (userapp/models.py)**

from django.db import models

class User(models.Model):

ROLE\_CHOICES = (

('admin', 'Admin'),

('kasir', 'Kasir'),

)

name = models.CharField(max\_length=100)

username = models.CharField(max\_length=100, unique=True)

password = models.CharField(max\_length=256)

role = models.CharField(max\_length=10, choices=ROLE\_CHOICES)

def \_\_str\_\_(self):

return self.username

**5. Buat serializer (userapp/serializers.py)**

from rest\_framework import serializers

from .models import User

class UserSerializer(serializers.ModelSerializer):

class Meta:

model = User

fields = '\_\_all\_\_'

**6. Buat views API (userapp/views.py)**

from rest\_framework.decorators import api\_view

from rest\_framework.response import Response

from rest\_framework import status

from .models import User

from .serializers import UserSerializer

import hashlib

@api\_view(['GET'])

def list\_users(request):

users = User.objects.all()

serializer = UserSerializer(users, many=True)

return Response(serializer.data)

@api\_view(['POST'])

def create\_user(request):

data = request.data

data['password'] = hashlib.sha256(data['password'].encode()).hexdigest()

serializer = UserSerializer(data=data)

if serializer.is\_valid():

serializer.save()

return Response(serializer.data, status=status.HTTP\_201\_CREATED)

return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)

@api\_view(['POST'])

def login\_user(request):

username = request.data.get('username')

password = hashlib.sha256(request.data.get('password').encode()).hexdigest()

try:

user = User.objects.get(username=username, password=password)

serializer = UserSerializer(user)

return Response(serializer.data)

except User.DoesNotExist:

return Response({'error': 'Invalid credentials'}, status=status.HTTP\_401\_UNAUTHORIZED)

**7. Tambahkan URL (userapp/urls.py)**

from django.urls import path

from . import views

urlpatterns = [

path('users/', views.list\_users),

path('users/create/', views.create\_user),

path('users/login/', views.login\_user),

]

**Dan di pos\_backend/urls.py:**

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

path('admin/', admin.site.urls),

path('api/', include('userapp.urls')),

]

**8. Migrasi database & runserver**

**Command:**

python manage.py makemigrations

python manage.py migrate

python manage.py runserver

🔗 Sekarang backend kamu aktif di http://127.0.0.1:8000/api/

Contoh:

* GET /api/users/ → list user
* POST /api/users/create/ → tambah user
* POST /api/users/login/ → login

Setelah ini, kita bisa lanjut ke bagian **Flutter Desktop**:

* Menampilkan list user dari API
* Form login
* Tambah user

Kalau kamu sudah berhasil setup backend, aku bantu lanjut ke bagian Flutter-nya.

* **Bagian 2: Setup Flutter Desktop frontend**

**Tujuan Pertama: Manajemen User**

**Fitur yang kita buat sekarang:**

* Login Screen
* User List Screen
* Tambah User Screen

1. **Setup Project Flutter Desktop**

* **Buat project Flutter baru**

**Command:**

flutter create pos\_flutter

cd pos\_flutter

* **Tambahkan dependencies di pubspec.yaml**

dependencies:

flutter:

sdk: flutter

http: ^0.13.6

flutter\_hooks: ^0.20.5

1. **Buat Model User**

**lib/models/user.dart**

class User {

final int? id;

final String name;

final String username;

final String password;

final String role;

User({

this.id,

required this.name,

required this.username,

required this.password,

required this.role,

});

factory User.fromJson(Map<String, dynamic> json) {

return User(

id: json['id'],

name: json['name'],

username: json['username'],

password: json['password'],

role: json['role'],

);

}

Map<String, dynamic> toJson() {

return {

'name': name,

'username': username,

'password': password,

'role': role,

};

}

}

**3. Buat Service untuk koneksi ke Django API**

**lib/services/api\_service.dart**

import 'dart:convert';

import 'package:http/http.dart' as http;

import '../models/user.dart';

class ApiService {

static const String baseUrl = 'http://127.0.0.1:8000/api';

static Future<List<User>> getUsers() async {

final res = await http.get(Uri.parse('$baseUrl/users/'));

if (res.statusCode == 200) {

final data = jsonDecode(res.body) as List;

return data.map((json) => User.fromJson(json)).toList();

} else {

throw Exception('Gagal ambil data user');

}

}

static Future<User?> login(String username, String password) async {

final res = await http.post(

Uri.parse('$baseUrl/users/login/'),

headers: {'Content-Type': 'application/json'},

body: jsonEncode({'username': username, 'password': password}),

);

if (res.statusCode == 200) {

return User.fromJson(jsonDecode(res.body));

} else {

return null;

}

}

static Future<void> addUser(User user) async {

final res = await http.post(

Uri.parse('$baseUrl/users/create/'),

headers: {'Content-Type': 'application/json'},

body: jsonEncode(user.toJson()),

);

if (res.statusCode != 201) {

throw Exception('Gagal menambah user');

}

}

}

**4. Buat UI: Login Screen**

**lib/screens/login\_screen.dart**

import 'package:flutter/material.dart';

import '../services/api\_service.dart';

import 'user\_list\_screen.dart';

class LoginScreen extends StatefulWidget {

const LoginScreen({super.key});

@override

State<LoginScreen> createState() => \_LoginScreenState();

}

class \_LoginScreenState extends State<LoginScreen> {

final usernameController = TextEditingController();

final passwordController = TextEditingController();

void login() async {

final username = usernameController.text;

final password = passwordController.text;

final user = await ApiService.login(username, password);

if (user != null) {

Navigator.pushReplacement(

context,

MaterialPageRoute(builder: (\_) => const UserListScreen()),

);

} else {

ScaffoldMessenger.of(context).showSnackBar(

const SnackBar(content: Text('Login gagal')),

);

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text('Login')),

body: Padding(

padding: const EdgeInsets.all(20),

child: Column(

children: [

TextField(controller: usernameController, decoration: const InputDecoration(labelText: 'Username')),

TextField(controller: passwordController, obscureText: true, decoration: const InputDecoration(labelText: 'Password')),

const SizedBox(height: 20),

ElevatedButton(onPressed: login, child: const Text('Login')),

],

),

),

);

}

}

**5. Buat UI: List User Screen**

**lib/screens/user\_list\_screen.dart**

import 'package:flutter/material.dart';

import '../models/user.dart';

import '../services/api\_service.dart';

import 'add\_user\_screen.dart';

class UserListScreen extends StatefulWidget {

const UserListScreen({super.key});

@override

State<UserListScreen> createState() => \_UserListScreenState();

}

class \_UserListScreenState extends State<UserListScreen> {

late Future<List<User>> users;

@override

void initState() {

super.initState();

users = ApiService.getUsers();

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text('Daftar User')),

body: FutureBuilder<List<User>>(

future: users,

builder: (context, snapshot) {

if (snapshot.hasData) {

final data = snapshot.data!;

return ListView.builder(

itemCount: data.length,

itemBuilder: (\_, i) => ListTile(

title: Text(data[i].name),

subtitle: Text('${data[i].username} (${data[i].role})'),

),

);

} else if (snapshot.hasError) {

return Center(child: Text('Error: ${snapshot.error}'));

}

return const Center(child: CircularProgressIndicator());

},

),

floatingActionButton: FloatingActionButton(

onPressed: () async {

final result = await Navigator.push(

context,

MaterialPageRoute(builder: (\_) => const AddUserScreen()),

);

if (result == true) {

setState(() {

users = ApiService.getUsers();

});

}

},

child: const Icon(Icons.add),

),

);

}

}

**6. Buat UI: Tambah User**

**lib/screens/add\_user\_screen.dart**

import 'package:flutter/material.dart';

import '../models/user.dart';

import '../services/api\_service.dart';

class AddUserScreen extends StatefulWidget {

const AddUserScreen({super.key});

@override

State<AddUserScreen> createState() => \_AddUserScreenState();

}

class \_AddUserScreenState extends State<AddUserScreen> {

final nameController = TextEditingController();

final usernameController = TextEditingController();

final passwordController = TextEditingController();

String role = 'kasir';

void submit() async {

final newUser = User(

name: nameController.text,

username: usernameController.text,

password: passwordController.text,

role: role,

);

try {

await ApiService.addUser(newUser);

Navigator.pop(context, true);

} catch (e) {

ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Gagal tambah user: $e')));

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text('Tambah User')),

body: Padding(

padding: const EdgeInsets.all(20),

child: Column(

children: [

TextField(controller: nameController, decoration: const InputDecoration(labelText: 'Nama')),

TextField(controller: usernameController, decoration: const InputDecoration(labelText: 'Username')),

TextField(controller: passwordController, obscureText: true, decoration: const InputDecoration(labelText: 'Password')),

DropdownButton<String>(

value: role,

items: const [

DropdownMenuItem(value: 'admin', child: Text('Admin')),

DropdownMenuItem(value: 'kasir', child: Text('Kasir')),

],

onChanged: (value) => setState(() => role = value!),

),

const SizedBox(height: 20),

ElevatedButton(onPressed: submit, child: const Text('Simpan')),

],

),

),

);

}

}

**7. Set main.dart ke Login**

import 'package:flutter/material.dart';

import 'screens/login\_screen.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'POS App',

debugShowCheckedModeBanner: false,

home: const LoginScreen(),

);

}

}

**Siap digunakan!**

Kamu bisa:

* Login dengan user yang sudah dibuat di backend
* Melihat list user
* Menambahkan user baru dari Flutter