

**LAPORAN TUGAS BESAR PEMROGRAMAN BERBASIS MOBILE
APLIKASI KIDS DAYCARE**



Dosen Pengampu :

Adi Wahyu Pribadi, S.Si., M.Kom.

Disusun Oleh:

Anisa Al – Harani (4521210021)

Yosua Jelianfero Solissa (4521210053)

Adelia Nurlina Putri (4521210059)

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PANCASILA
JAKARTA
2024**

I. Deskripsi Aplikasi

Dengan semakin meningkatnya kebutuhan akan layanan penitipan anak yang profesional dan terpercaya, aplikasi daycare muncul sebagai solusi teknologi modern untuk mengelola berbagai aspek operasional pusat penitipan anak. Latar belakang pengembangan aplikasi ini berakar dari kebutuhan untuk meningkatkan efisiensi, transparansi, dan kualitas layanan di daycare. Seiring dengan meningkatnya jumlah orang tua yang bekerja dan mempercayakan perawatan anak mereka kepada penyedia layanan daycare, muncul tuntutan yang lebih tinggi terhadap standar pengelolaan dan pemantauan anak-anak selama berada di daycare.

Aplikasi daycare dirancang untuk meningkatkan efisiensi dan efektivitas pengelolaan pusat penitipan anak, serta memastikan keamanan dan kenyamanan anak-anak selama berada di daycare. Aplikasi ini memainkan peran penting dalam mempermudah tugas pengelola daycare serta memberikan ketenangan bagi orang tua dengan menyediakan berbagai fitur yang mendukung pemantauan, pencatatan, dan komunikasi secara real-time.

Pencatatan kehadiran adalah salah satu fitur utama yang memungkinkan staf daycare mencatat waktu kedatangan dan kepulangan anak dengan mudah. Ini membantu memastikan anak-anak dipantau sejak mereka tiba hingga mereka pulang, dan data ini bisa berguna dalam berbagai situasi administrasi dan keamanan.

Pemantauan kesehatan merupakan fitur vital lainnya, di mana aplikasi ini memungkinkan pencatatan suhu tubuh dan kondisi kesehatan harian anak. Informasi ini sangat penting untuk mendeteksi dini adanya tanda-tanda penyakit, memastikan anak-anak selalu dalam kondisi sehat, dan mematuhi regulasi kesehatan yang berlaku.

Dalam hal nutrisi, aplikasi daycare menyediakan fitur untuk pencatatan makan dan minum, di mana jenis dan jumlah makanan serta minuman yang dikonsumsi anak dapat dicatat dengan rinci. Ini tidak hanya membantu memastikan anak mendapatkan nutrisi yang cukup, tetapi juga memudahkan orang tua mengetahui pola makan anak mereka selama di daycare.

Selain itu, aplikasi ini juga mengakomodasi pencatatan kebersihan dan istirahat, mencatat penggunaan toilet atau diaper serta waktu istirahat anak. Informasi ini penting untuk menjaga kebersihan dan kesehatan anak serta memastikan mereka mendapatkan istirahat yang cukup selama di daycare.

Fitur pemberian botol susu merupakan pencatatan waktu dan jenis susu yang diberikan kepada anak, apakah itu ASI, susu formula, atau susu biasa. Hal ini membantu memastikan kebutuhan nutrisi anak terpenuhi sesuai dengan arahan dari orang tua.

Aplikasi daycare juga menyertakan catatan tambahan seperti pemberian vitamin dan kebutuhan khusus lainnya yang perlu dibawa oleh orang tua. Memastikan semua kebutuhan anak terpenuhi dan orang tua dapat mempersiapkan segala sesuatu yang diperlukan dengan baik.

Terakhir, laporan harian yang diberikan oleh aplikasi ini memungkinkan orang tua menerima ringkasan lengkap tentang aktivitas dan kondisi anak selama berada di daycare, termasuk aspek kesehatan, makan, minum, tidur, dan lainnya. Laporan ini memberikan gambaran menyeluruh yang membantu orang tua merasa lebih tenang dan terinformasi tentang keseharian anak mereka.

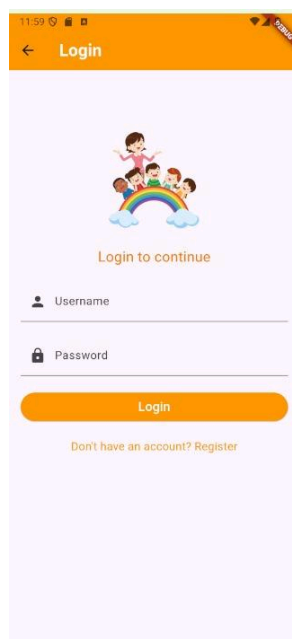
Secara keseluruhan, aplikasi daycare dibuat untuk memastikan bahwa anak-anak mendapatkan perawatan terbaik, mempermudah pengelolaan operasional daycare, dan meningkatkan komunikasi serta transparansi dengan orang tua, sehingga menciptakan lingkungan yang aman dan nyaman bagi anak-anak.

II. Layout

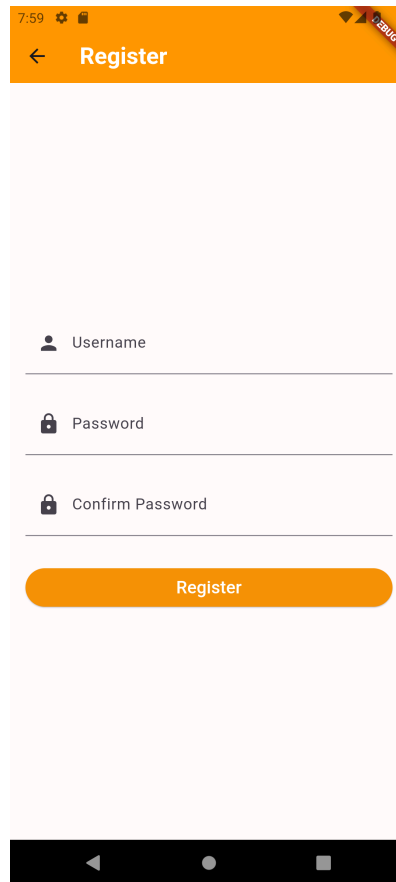
A. Tampilan Orang Tua

1. Login

a. Login



b. Register



7:59

← Register

Username

Password

Confirm Password

Register

This screenshot shows the 'Register' screen of a mobile application. At the top, there is an orange header bar with a back arrow and the title 'Register'. The main area has a light pink background. It contains three input fields: 'Username' with a person icon, 'Password' with a lock icon, and 'Confirm Password' with a lock icon. Below these fields is an orange rounded button labeled 'Register'. The bottom of the screen shows the standard Android navigation bar.

2. Menu



11:59

← Menu



Welcome yosuasolissa@gmail.com

Daftarkan Anak Anda

Lihat Daftar Anak Anda

This screenshot shows the 'Menu' screen of the mobile application. It features an orange header bar with a back arrow and the title 'Menu'. The background is light purple. In the center, there is a colorful illustration of five children playing under a rainbow. Below the illustration, the text 'Welcome yosuasolissa@gmail.com' is displayed. At the bottom, there are two orange rounded buttons: 'Daftarkan Anak Anda' and 'Lihat Daftar Anak Anda'. The top status bar shows the time as 11:59.

3. Input Data Anak

5:18

← Input Data Anak

Informasi Pribadi Anak

👤

Nama lengkap

📅

Tanggal lahir

♂️

Jenis kelamin

▼

🏠

Alamat rumah

Informasi Kontak Darurat

👤

Nama kontak

👥

Hubungan

☎️

Nomor telepon darurat

Riwayat Medis

🏥

Riwayat medis anak

Simpan

4. Lihat Aktifitas Anak

12:00

← Laporan Aktivitas Anak

1.0

2.0

1.0

1.0

1.0

2.0

Meals

Toilet

Rest

Bottle

Vitamin

Item Needs

Nama

25

Arrival

8:57 AM

Kondisi

23

Tanggal

2024-06-30 08:59:31

Suhu Tubuh

23°C

Makanan

comment	Food	Quantity
23	23	Some
43	asd	Some

Toilet

Waktu	Jenis	Kondisi
8:58 AM	Potty	Wet

12:01

← Laporan Aktivitas Anak

08:58 AM Potty Wet

Istirahat

Mulai Istirahat	Akhir Istirahat
08:58:00	08:58:00

Botol

Waktu	ML	Tipe
8:58 AM	67	Formula

Mandi

Pagi	8:59 AM
Siang	8:59 AM

Vitamin

Nama	Jumlah	Waktu
67	21	08:59:00

Catatan Untuk Orang Tua

Hand towel: 1

Diapers: 14

B. Tampilan Pengasuh

1. Login

11:58

Welcome to Kids Daycare!

Login

Register

11:59

Login

Login to continue

Username

Password

Login

Don't have an account? Register

2. Input Aktivitas Anak

12:02

Input Child Activity for Anak 11

Child Activity Information:

Child Name

Anak 11

Input Date Time

Arrival Time

Body Temperature

Conditions

Meals:

Breakfast

Food

Quantity

None

Comment

Snack

Food

12:02

Input Child Activity for Anak 11

Quantity

None

Comment

Dinner

Food

Quantity

None

Comment

Fluids

Food

Quantity

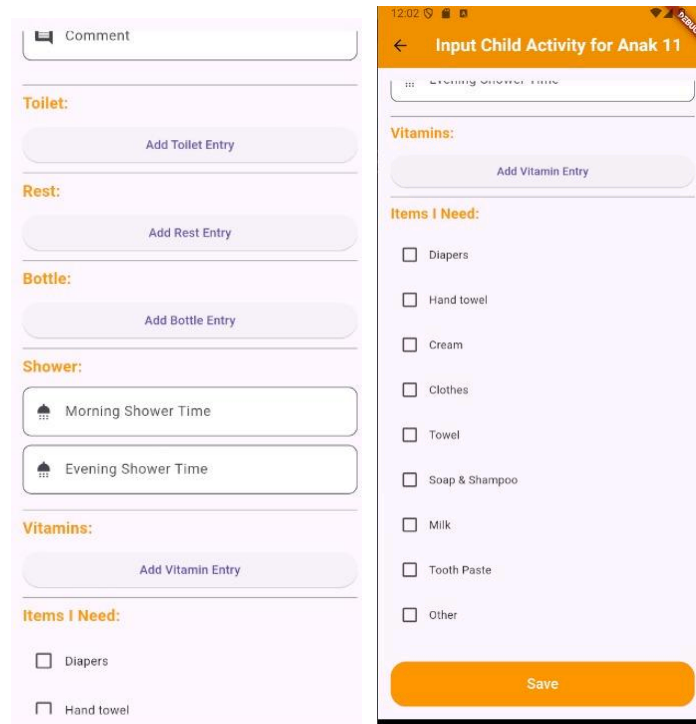
None

Comment

Other

Food

Quantity



III. Library

```
import 'dart:convert';
import 'dart:async';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';
import 'package:auto_spacing/auto_spacing.dart';
```

Beberapa library yang digunakan untuk aplikasi ini diantaranya :

Library	Deskripsi
<code>dart:convert</code>	Digunakan untuk encoding dan decoding data. Data dapat dikonversi dari dan ke format JSON, UTF-8, dan base64.
<code>dart:async</code>	Menyediakan dukungan untuk asynchronous programming termasuk Future, Stream, Timer, dll.
<code>package:flutter/material.dart</code>	Bagian dari Flutter SDK dan menyediakan komponen UI berbasis material design.
<code>package:http/http.dart</code>	Digunakan untuk melakukan permintaan

	HTTP seperti mengirim permintaan GET, POST, PUT, DELETE, dll ke API.
package:auto_spacing/auto_spacing.dart	Digunakan untuk mengatur spasi secara otomatis dalam aplikasi Flutter

IV. Source Code

A. activity_input.dart

```
import 'dart:convert';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';

class ToiletEntry {
  TextEditingController timeController = TextEditingController();
  String type = 'Diaper';
  String condition = 'DR';

  TextEditingController notesController = TextEditingController();
}

class RestEntry {
  TextEditingController startController = TextEditingController();
  TextEditingController endController = TextEditingController();
}

class BottleEntry {
  TextEditingController timeController = TextEditingController();
  TextEditingController mlController = TextEditingController();
  String type = 'Breast';
}

class VitaminEntry {
  TextEditingController timeController = TextEditingController();
  TextEditingController nameController = TextEditingController();
  TextEditingController amountController = TextEditingController();
}

class ActivityInputScreen extends StatefulWidget {
  final String childName="ian";
  final dynamic nama_anak;
  final String id_anak;

  bool StatusPut=false;
  final String id_pengasuh;

  ActivityInputScreen({required this.id_anak,required
this.nama_anak,required this.id_pengasuh});

  @override
  _ActivityInputScreenState createState() =>
_ActivityInputScreenState();
}
```



```

class _ActivityInputScreenState extends State<ActivityInputScreen>
{
    final TextEditingController _namaAnakController =
TextEditingController();
    final TextEditingController _tanggalController =
TextEditingController();
    final TextEditingController _arrivalController =
TextEditingController();
    final TextEditingController _bodyTemperatureController =
TextEditingController();
    final TextEditingController _conditionsController =
TextEditingController();

    // Meal controllers
    final List<TextEditingController> _mealFoodControllers =
List.generate(6, (index) => TextEditingController());
    final List<TextEditingController> _mealCommentControllers =
List.generate(6, (index) => TextEditingController());
    final List<String> _mealQuantities = List.generate(6, (index) =>
'None');

    List<ToiletEntry> _toiletEntries = [];
    List<RestEntry> _restEntries = [];
    List<BottleEntry> _bottleEntries = [];
    List<VitaminEntry> _vitaminEntries = [];

String quantityFinal="1";
    final TextEditingController _showerMorningController =
TextEditingController();
    final TextEditingController _showerEveningController =
TextEditingController();

    final Map<String, dynamic> _itemNeeds = {
'Diapers': {'checked': false, 'quantity': '1', 'visible':
false},
'Hand towel': {'checked': false, 'quantity': '1', 'visible':
false},
'Cream': {'checked': false, 'quantity': '1', 'visible': false},
'Clothes': {'checked': false, 'quantity': '1', 'visible':
false},
'Towel': {'checked': false, 'quantity': '1', 'visible': false},
'Soap & Shampoo': {'checked': false, 'quantity': '1',
'visible': false},
'Milk': {'checked': false, 'quantity': '1', 'visible': false},
'Tooth Paste': {'checked': false, 'quantity': '1', 'visible':
false},
'Other': {'checked': false, 'quantity': '1', 'visible': false},
};
    final TextEditingController _otherItemController =
TextEditingController();
    // List<dynamic> anak=[];

late String id_laporan="";
late List<Map<String,dynamic>> itemneed=[];
late List<Map<String,dynamic>> meals=[];
late List<Map<String,dynamic>> toilet=[];
late List<Map<String,dynamic>> rest=[];
late List<Map<String,dynamic>> vitamin=[];

```

```

late List<Map<String,dynamic>>bottle=[];

int indexItemneed=0;
// Future<void>PutDataAnak(String id_anak)async{
//   try {

//     final response = await
http.post(Uri.parse('http://10.0.2.2/daycare_api/put_Data_anak.php'
),body: {'id_anak':id_anak});
//   if (response.statusCode==200) {
//     setState(() {
//       print("Put ID Anak Berhasil...");
//       anak=jsonDecode(response.body);
//       widget.StatusPut=true;
//     });

//   } else {
//     setState(() {

//       widget.StatusPut=false;
//     });
//     print("Put Id Anak Gagal");

//   }
//   }
//   catch(e){
//     print("Put Id Anak Gagal");

//     print(e);

//   }
// }

// TOILET
bool statusPostToilet=false;
Future<void>postToilet(String value,String value2,String
value3,String value4)async{
  try{

    final response = await
http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_toilet.p
hp'),body: {
"waktu":value,
"type":value2,
"kondisi":value3,
"catatan":value4,
'id_laporan':id_laporan,
  });
  var $result=jsonDecode(response.body);
  if($result["status"]=="Berhasil"){
    print("Api Toilet Berhasil di gunakan...");
    setState(() {
      statusPostToilet=true;
    });
  }
}

}

```

```

        catch(e){
            setState(() {
                print("Api Toilet Gagal di gunakan...");
                statusPostToilet=false;
            });
            print(e);
        }
    }

    // MEALLL
    bool statusPostMeal=false;
    Future<void>postMeal(String value,String value2,String
    value3)async{
        try{

            final response = await
            http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_meals.ph
            p'),body: {
                "food":value,
                "quantity":value2,
                "comment":value3,
                'id_laporan':id_laporan,
            });

            var $result=jsonDecode(response.body);
            if($result["status"]=="Berhasil"){

            print("Api Meal Berhasil di gunakan...");
                setState(() {
                    statusPostMeal=true;
                });
            }

            }
            catch(e){
                setState(() {
                    print("Api Meal Gagal di gunakan...");
                    statusPostMeal=false;
                });
                print(e);
            }
        }

        // RESTT
        bool statusPostReast=false;
        Future<void>postReast(String value,String value2)async{
            try{

                final response = await
                http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_rest.php
                '),body: {
                    "start_time":value,
                    "end_time":value2,
                    'id_laporan':id_laporan,
                });

                var $result=jsonDecode(response.body);
                if($result["status"]=="Berhasil"){

```

```

print("Api Rest Berhasil di gunakan...");
    setState(() {
        statusPostReast=true;
    });
}

}
catch(e){
    setState(() {
        statusPostReast=false;
print("Api Rest Gagal di gunakan...");
    });
    print(e);
}
}

// Bottle
bool statusPostBottle = false;
Future<void> postBottle(String time, String ML, String type) async
{
    try {
        final response = await http.post(
Uri.parse('http://10.0.2.2/daycare_api/post_Data_bottle.php'),
        body: {
            "time": time,
            "ML": ML,
            "type": type,
            'id_laporan':id_laporan,
        },
    );

    var result = jsonDecode(response.body);
    if (result["status"] == "Berhasil") {

print("Api Bottle Berhasil di gunakan...");
        setState(() {
            statusPostBottle = true;
        });
    } else {
        print("POST bottle GAGAL: ${result['message']}");
        setState(() {
            statusPostBottle = false;
        });
    }
    } catch (e) {
        print("Error pada saat POST bottle: $e");
        setState(() {
            statusPostBottle = false;
        });
    }
}

bool statusPostShower = false;

Future<void> postShower(String morningShower, String eveningShower)

```

```

async {
    try {
        final response = await http.post(
Uri.parse('http://10.0.2.2/daycare_api/post_Data_shower.php'),
        body: {
            "morning_shower": morningShower,
            "evening_shower": eveningShower,
            'id_laporan':id_laporan,
        },
    );

    var result = jsonDecode(response.body);
    if (result["status"] == "Berhasil") {

print("Api Shower Berhasil di gunakan...");
        statusPostShower = true;
    } else {
        print("Api shower GAGAL: ${result['message']}");
        statusPostShower = false;
    }
    } catch (e) {
        print("Error pada saat API POST shower: $e");
        statusPostShower = false;
    }
}

bool statusPostVitamin = false;

//----- POST TO
Vitamin-----
Future<void> postVitamin(String vitaminName, String amount, String
time) async {
    try {
        final response = await http.post(
Uri.parse('http://10.0.2.2/daycare_api/post_Data_vitamin.php'),
        body: {
            "vitamin_name": vitaminName,
            "amount": amount,
            "time": time,
            'id_laporan':id_laporan,
        },
    );

    var result = jsonDecode(response.body);
    if (result["status"] == "Berhasil") {
print("Api Vitamin Berhasil di gunakan...");
        statusPostVitamin = true;
    } else {
        print("POST vitamin GAGAL: ${result['message']}");
        statusPostVitamin = false;
    }
    } catch (e) {
        print("Error pada saat POST vitamin: $e");
        statusPostVitamin = false;
    }
}

```

```

}

//----- POST TO
Item-----

bool statusPostItem = false;

Future<void> postItem(String key, String quantity) async {
  if (quantity=='none' || quantity=='None') {
    quantity='1';
  }
  try {
    final response = await http.post(
      Uri.parse('http://10.0.2.2/daycare_api/post_Data_item.php'),
      body: {
        "key": key,
        "quantity": quantity,
        'id_laporan':id_laporan,
      },
    );

    var result = jsonDecode(response.body);
    if (result["status"] == "Berhasil"){
      print("Api Item Berhasil di gunakan...");
      statusPostItem = true;
    } else {
      print("POST item GAGAL: ${result['message']}");
      statusPostItem = false;
    }
  } catch (e) {
    print("Error pada saat POST item: $e");
    statusPostItem = false;
  }
}

Future<bool> laporanAnak(String $id_anak)async {
  print(_tanggalController.text);
  if ($id_anak!='') {
    if (widget.nama_anak!="") {
      try {
        if
        (_arrivalController.text=='' || _bodyTemperatureController.text=='' ||
        _conditionsController.text=='') {

          return false;
        }
      }
      else{
        print(_tanggalController);
        print(_bodyTemperatureController.text);
        print(_conditionsController.text);
        print(_arrivalController.text);
        var response= await
http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_laporana
nak.php'),body: {
          'id_anak':widget.id_anak.toString(),
          'arrival_time':_arrivalController.text,
          'temperature':_bodyTemperatureController.text,
          'kondisi':_conditionsController.text,

```

```

        'timestamp':_tanggalController.text,
    });
print(response.body);
    if (response.statusCode==200 || response.statusCode==201 ) {

        var result=jsonDecode(response.body);
        if (result['status']!="Gagal") {
            if (result['new_id']!=" ") {
print("Api Laporan Anak Berhasil di gunakan...");
                setState(() {

                    id_laporan=result['new_id'].toString();
                });
                return true;
            }else{
setState(() {

print("Api Laporan Anak Gagal di gunakan...");
                id_laporan="reset";
            });

                return false;
            }
        }else{
            // print(result);
            return false;
        }
        print("AMBIL ID NYA LALU TARO DI TABEL LAIN");

    }
    else{
        return false;
    }
}

} catch (e) {
    print(e);
}

    return true;
    }else{
        return false;
    }

    }else{return false;}
}

Future<void> updateDataAnak() async{
    bool initialpost= await laporanAnak(widget.id_anak.toString());

    if (initialpost==true && id_laporan!="reset") {
        print("ID LAPORAN : ${id_laporan}");
        bool initialpost2=false;
        try {

```

```

        print("-----INFO
STATUS-----");
        print("laporan anak selesai, dan selanjutnya post tabel lainnya
dengan laporan anak menjadi foreign key");

print("-----
-----");
if (initialpost2==false && widget.nama_anak!="") {

    //----- POST TO
MEALS-----
    if (meals.length>0) {
        for (var i = 0; i < meals.length; i++) {
            postMeal(meals[i]['Food'], meals[i]['Quantity'],
meals[i]['Comment']);
            if (statusPostMeal) {
                print('POST MEAL SUCCES....');
            }
            else{
                print("POST MEAL GAGAL");
            }
        }
    }
    //----- POST TO
TOILET-----
    if (toilet.length>0) {

        for (var i = 0; i < toilet.length; i++) {
            postToilet(toilet[i]['waktu'], toilet[i]['type'],
toilet[i]['kondisi'], toilet[i]['notes']);
            if (statusPostToilet) {
                print('POST Toilet SUCCES....');
            }
            else{
                print("Post TOILET GAGAL");
            }
        }
    }

    //----- POST TO
Rest-----
    if (rest.length>0) {
        for (var i = 0; i < rest.length; i++) {

            postReast(rest[i]['start_time'], rest[i]['end_time']);
            if (statusPostReast) {
                print('POST Rest SUCCES....');
            }
            else{
                print("POST REST GAGAL");
            }
        }
    }

    //----- POST TO
Bottle-----

```



```

    if (bottle.length>0) {
        for (var i = 0; i < bottle.length; i++) {

            postBottle(bottle[i]['time'], bottle[i]['ML'],bottle[i]['type']);
            if (statusPostBottle) {
                print('POST bottle SUCCES....');
            }
            else{
                print("POST bottle GAGAL");
            }
        }
    }

    //----- POST TO
    Shower-----

    if (_showerMorningController.text!="" &&
        _showerEveningController.text!="") {

        postShower(_showerEveningController.text,
            _showerEveningController.text);
        if (statusPostShower) {
            print('POST shower SUCCES....');
        }
        else{
            print("POST shower GAGAL");
        }
    }

    //----- POST TO
    vitamin-----

    if (vitamin.length>0) {
        for (var i = 0; i < vitamin.length; i++) {

            postVitamin(vitamin[i]['vitamin_name'], vitamin[i]['amount'],
                vitamin[i]['time']);
            if (statusPostVitamin) {
                print('POST Vitamin SUCCES....');
            }
            else{
                print("POST Vitamin GAGAL");
            }
        }
    }

    //----- POST TO
    Item-----

    if (itemneed.length>0) {
        for (var i = 0; i < itemneed.length; i++) {
            postItem(itemneed[i]['key'], itemneed[i]['quantity']);
            if (statusPostItem) {
                print('POST Item SUCCES....');
            }
            else{
                print("GAGAL");
            }
        }
    }
}

```

```

initialpost2=true;
}
}
catch (e) {
    print(e);
    print("UPDATE INITIAL POST BERMASALAH...");
    initialpost2=false;
}

    if (initialpost2==true) {
        print("-----INFO
STATUS-----");
        print("-----UPDATE POST BERHASIL TANPA KENDALA,
TERIMA KASIH----- ");
    print("-----
-----");

        }
        else{
            print("-----WARNING
INFO-----");
            print("-----UPATE POST MENGALMI KENDALA,DAN GAGAL
DI EKSEKUSI-----");
        print("-----
-----");

        }
    }
    else{
        print("-----INFO
STATUS-----");
        print("-----UPATE POST MEMBUTUHKAN BEBERAPA
ATTRIBUT, ULANGI----- ");
    print("-----
-----");

    }

    }

    @override
    void initState() {
        print("DATA ID ANAK : ");
        print(widget.id_anak);
        super.initState();
        // print(widget.result)
    }

```

```

        // print(anak[0]['nama_lengkap']);
        _namaAnakController.text =widget.nama_anak; // Mengisi nama
anak dari parameter
    }

    void _submitActivity(BuildContext context) {
        // Logic to save child activity data
        ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(content: Text('Child activity data saved
successfully'))),
        );
    }

    void _addNewToiletEntry() {
        setState(() {
            _toiletEntries.add(ToiletEntry());
        });
    }

    void _addNewRestEntry() {
        setState(() {
            _restEntries.add(RestEntry());
        });
    }

    void _addNewBottleEntry() {
        setState(() {
            _bottleEntries.add(BottleEntry());
        });
    }

    void _addNewVitaminEntry() {
        setState(() {
            _vitaminEntries.add(VitaminEntry());
        });
    }

    void _selectDate() async {
        DateTime? pickedDate = await showDatePicker(
            context: context,
            initialDate: DateTime.now(),
            firstDate: DateTime(2000),
            lastDate: DateTime(2101),
            builder: (BuildContext context, Widget? child) {
                return Theme(
                    data: ThemeData.light().copyWith(
                        primaryColor: Colors.orange,
                        colorScheme: ColorScheme.light(primary: Colors.orange,
secondary: Colors.orange),
                        buttonTheme: ButtonThemeData(textTheme:
ButtonTextTheme.primary),
                    ),
                    child: child!,
                );
            },
        );
        if (pickedDate != null) {
            setState(() {

```

```

        _tanggalController.text = "${pickedDate.toLocal()}.split('
')[0];
    });
}
}

void _selectTime(TextEditingController controller,int
index,String Object,String label) async {
    TimeOfDay? pickedTime = await showTimePicker(
        context: context,
        initialTime: TimeOfDay.now(),
        builder: (BuildContext context, Widget? child) {
            return Theme(
                data: ThemeData.light().copyWith(
                    primaryColor: Colors.orange,
                    colorScheme: ColorScheme.light(primary: Colors.orange,
secondary: Colors.orange),
                    buttonTheme: ButtonThemeData(textTheme:
ButtonTextTheme.primary),
                ),
                child: child!,
            );
        },
    );
    if (pickedTime != null) {
        setState(() {
            controller.text = pickedTime.format(context);
            if (Object=="Toilet") {

Map<String,dynamic> toiletNew={
'key':index,
'waktu':"none",
'type':"none",
'kondisi':"none",
'notes':controller.text,
};
if (toilet.length<=0) {
    toilet.add(toiletNew);
}
bool isFind=false;
for (var i = 0; i < toilet.length; i++) {
    if (toilet[i]['key']==index) {
        setState(() {

            isFind=true;
        });
        toilet[i]['waktu']=controller.text;
        break;

    }
}
if (isFind) {
}else{
    toilet.add(toiletNew);
}

}
}
}

```

```

// -----REST
OBJECT-----
    else if(Object=='Rest'){

Map<String,dynamic> restNew={
'key':index,
'start_time':"none",
'end_time':"none",
};
if (rest.length<=0) {
    rest.add(restNew);
}
bool isFind=false;
for (var i = 0; i < rest.length; i++) {
    if (rest[i]['key']==index) {
        setState(() {

            isFind=true;
        });
        if (label=='Start Time') {

            rest[i]['start_time']=controller.text;
        }else{

            rest[i]['end_time']=controller.text;
        }
        break;
    }
}
if (isFind) {
}
else{
    rest.add(restNew);
}

    }
// -----BOTTLE
OBJECT-----
    else if(Object=='Bottle'){

Map<String,dynamic> bottleNew={
'key':index,
'time':"none",
'ML':"none",
'type':"none",
};
if (bottle.length<=0) {
    bottle.add(bottleNew);
}
bool isFind=false;
for (var i = 0; i < bottle.length; i++) {
    if (bottle[i]['key']==index) {
        setState(() {
            bottle[i]['time']=controller.text;
            isFind=true;

```

```

    });
    break;

    }
}
if (isFind) {
} else {
    bottle.add(bottleNew);
}

// VITAMIN
    }
    else if (Object == 'Vitamin') {

Map<String, dynamic> vitaminNew = {
    'key': index,
    'vitamin_name': "none",
    'amount': "none",
    'time': "none",
};
if (vitamin.length <= 0) {
    vitamin.add(vitaminNew);
}
bool isFind = false;
for (var i = 0; i < vitamin.length; i++) {
    if (vitamin[i]['key'] == index) {
        setState(() {
            vitamin[i]['time'] = controller.text;
            isFind = true;
        });
        break;
    }
}
if (isFind) {
} else {
    vitamin.add(vitaminNew);
}

    }
    });
}
}

Widget _buildRestEntry(int index) {
    final RestEntry restEntry = _restEntries[index];
    return Column(
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: [
            Padding(
                padding: const EdgeInsets.only(top: 10, bottom: 5),
                child: Text(
                    'Rest Entry ${index + 1}',
                    style: TextStyle(
                        fontSize: 16,

```

```

        fontWeight: FontWeight.bold,
        color: Colors.black87,
      ),
    ),
  ),
  _buildTimeField(restEntry.startController, 'Start Time',
Icons.access_time,index,"Rest"),
  _buildTimeField(restEntry.endController, 'End Time',
Icons.access_time,index,"Rest"),
  SizedBox(height: 10),
],
);
}

@override
Widget build(BuildContext context) {
// print(widget.anak);
return Scaffold(
  appBar: AppBar(
    title: Text(
      'Input Child Activity for ${ widget.nama_anak}',
      style: TextStyle(
        color: Colors.white,
        fontWeight: FontWeight.bold,
      ),
    ),
    backgroundColor: Colors.orange,
  ),
  body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: SingleChildScrollView(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: <Widget>[
          _buildSectionTitle('Child Activity Information:'),
          _buildTextField(_namaAnakController, 'Child Name',
Icons.child_care,0,true,'Name'),
          _buildDateField(_tanggalController, 'Date',
Icons.calendar_today, _selectDate),
          _buildTimeField(_arrivalController, 'Arrival Time',
Icons.access_time,0,'Arrival'),
          _buildTextField(_bodyTemperatureController, 'Body
Temperature', Icons.thermostat,0,true,'Body'),
          _buildTextField(_conditionsController, 'Conditions',
Icons.health_and_safety,0,true,'Kondisi'),
          SizedBox(height: 20),
          Divider(color: Colors.grey),
          _buildSectionTitle('Meals:'),
          _buildMealSection('Breakfast', 0),
          _buildMealSection('Snack', 1),
          _buildMealSection('Lunch', 2),
          _buildMealSection('Dinner', 3),
          _buildMealSection('Fluids', 4),
          _buildMealSection('Other', 5),
          Divider(color: Colors.grey),
          _buildSectionTitle('Toilet:'),
          Column(

```

```

        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: [
          for (int i = 0; i < _toiletEntries.length; i++)
            _buildToiletEntry(i),
          ElevatedButton(
            onPressed: _addNewToiletEntry,
            child: Text('Add Toilet Entry'),
          ),
        ],
      ),
    Divider(color: Colors.grey),
    _buildSectionTitle('Rest:'),
    Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
        for (int i = 0; i < _restEntries.length; i++)
          _buildRestEntry(i),
        ElevatedButton(
          onPressed: _addNewRestEntry,
          child: Text('Add Rest Entry'),
        ),
      ],
    ),
    Divider(color: Colors.grey),
    _buildSectionTitle('Bottle:'),
    Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
        for (int i = 0; i < _bottleEntries.length; i++)
          _buildBottleEntry(i),
        ElevatedButton(
          onPressed: _addNewBottleEntry,
          child: Text('Add Bottle Entry'),
        ),
      ],
    ),
    Divider(color: Colors.grey),
    _buildSectionTitle('Shower:'),
    _buildTimeField(_showerMorningController, 'Morning
Shower Time', Icons.shower, 0, 'Shower'),
    _buildTimeField(_showerEveningController, 'Evening
Shower Time', Icons.shower, 0, 'Shower'),
    Divider(color: Colors.grey),
    _buildSectionTitle('Vitamins:'),
    Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
        for (int i = 0; i < _vitaminEntries.length; i++)
          _buildVitaminEntry(i),
        ElevatedButton(
          onPressed: _addNewVitaminEntry,
          child: Text('Add Vitamin Entry'),
        ),
      ],
    ),
    Divider(color: Colors.grey),
    _buildSectionTitle('Items I Need:'),
    Column(

```



```

        crossAxisAlignment: CrossAxisAlignment.start,
        children: _itemNeeds.keys.map<Widget>((key) {

            return Padding(
                padding: const EdgeInsets.symmetric(vertical:
4.0),
                child: Row(
                    children: <Widget>[
                        Checkbox(
                            value: _itemNeeds[key]['checked'],
                            onChanged: (bool? newValue) {
                                setState(() {

Map<String,dynamic> itemneedNew={
'key':key,
'quantity':"none",
});
if (itemneed.length>0) {
bool isFind=false;
for (var i = 0; i < itemneed.length; i++) {
    if (itemneed[i]['key']==key) {
        setState(() {

            isFind=true;
            itemneed.removeAt(i);
        });

        break;

    }
}
if (isFind) {
}else{
    itemneed.add(itemneedNew);
}
}else{

    itemneed.add(itemneedNew);
}

                indexItemneed+=1;

                                _itemNeeds[key]['checked'] =
newValue!;

                                _itemNeeds[key]['visible'] =
newValue;

                                });
                            },
                        ),
                        Expanded(
                            child: Text(key),
                        ),

```

```

        if (key == 'Other' &&
        _itemNeeds[key]['checked'])
            Expanded(
                child: TextField(
                    onChanged: (index){
                        for (var i = 0; i <
itemneed.length; i++) {
                            if (itemneed[i]['name']=='Other')
                            {
                                itemneed[i]['name_other']=_otherItemController.text;
                                }
                            }
                        },
                        controller: _otherItemController,
                        decoration: InputDecoration(
                            labelText: 'Specify Other Item',
                            border: OutlineInputBorder(
                                borderRadius:
BorderRadius.circular(10),
                                ),
                            ),
                        ),
                    ),
                if (key != 'Other' &&
        _itemNeeds[key]['visible'])
                    Expanded(
                        child: _buildDropdownField(
                            key,
                            'Quantity',
                            Icons.shopping_cart,

                            List<String>.generate(100, (index) =>
(index + 1).toString()),

                            _itemNeeds[key]['quantity'],
                            (newValue) {
                                setState(() {

                                    _itemNeeds[key]['quantity'] =
newValue!;

                                });
                            },
                            "ItemNeed"
                        ),
                    ),
                ],
            ),
        );
    }).toList(),
),
    SizedBox(height: 30),
    ElevatedButton(
        onPressed: (){
            updateDataAnak();
            Navigator.pop(context);
        },

```

```

        style: ElevatedButton.styleFrom(
          padding: EdgeInsets.symmetric(vertical: 15),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(20),
          ),
          backgroundColor: Colors.orange,
        ),
        child: Text(
          'Save',
          style: TextStyle(fontSize: 18, color:
Colors.white),
        ),
      ),
    ],
  ),
),
);
}

Widget _buildSectionTitle(String title) {
  return Padding(
    padding: const EdgeInsets.only(bottom: 10),
    child: Text(
      title,
      style: TextStyle(
        fontSize: 18,
        fontWeight: FontWeight.bold,
        color: Colors.orange,
      ),
    ),
  );
}

Widget _buildTextField(TextEditingController controller, String
label, IconData icon,int index,bool isSingle,String Object) {
  return Padding(
    padding: const EdgeInsets.only(bottom: 10),
    child: TextField(
      controller: controller,
      onChanged: (value){
if (isSingle==false) {
if (Object=='Meals') {

Map<String,dynamic> Item={
'key':index,
'Food':"none",
'Comment':"none",
'Quantity':'none',
};
if (meals.length<=0) {
  meals.add(Item);
}
bool isFind=false;
for (var i = 0; i < meals.length; i++) {
  if (meals[i]['key']==index) {
    setState(() {

```

```

        isFind=true;
    });
    if (label=='Food') {
        meals[i]['Food']=controller.text;
    }
    if (label=='Comment') {

        meals[i]['Comment']=controller.text;
    }
    if (label=='Quantity') {
        meals[i]['Quantity']=controller.text;
    }
    break;

    }
}
if (isFind) {
}else{
    meals.add(Item);
}
}
// TOILET
else if(Object=='Toilet'){

Map<String,dynamic> toiletNew={
'key':index,
'entry':"none",
'type':"none",
'kondisi':'none',
'notes':controller.text,
};
if (toilet.length<=0) {
    toilet.add(toiletNew);
}
bool isFind=false;
for (var i = 0; i < toilet.length; i++) {
    if (toilet[i]['key']==index) {
        setState(() {

            isFind=true;
        });
        if (label=='Notes') {
            toilet[i]['notes']=controller.text;
        }
        break;

    }
}
if (isFind) {
}else{
    toilet.add(toiletNew);
}
}

// VITAMINNN

```

```

else if(Object=='Vitamin'){

Map<String,dynamic> vitaminNew={
'key':index,
'vitamin_name':"none",
'amount':"none",
'time':'none',
};
if (vitamin.length<=0) {
    vitamin.add(vitaminNew);
}
bool isFind=false;
for (var i = 0; i < vitamin.length; i++) {
    if (vitamin[i]['key']==index) {
        setState(() {

            isFind=true;
            if (label=="Vitamin Name") {

                vitamin[i]['vitamin_name']=controller.text;
            }else{
                vitamin[i]['amount']=controller.text;

            }

        });
        break;

    }
}
if (isFind) {
}else{
    vitamin.add(vitaminNew);
}
}

// Bottle

else if(Object=='Bottle'){

Map<String,dynamic> bottleNew={
'key':index,
'time':"none",
'ML':"none",
'type':'none',

};
if (bottle.length<=0) {
    bottle.add(bottleNew);
}
bool isFind=false;
for (var i = 0; i < bottle.length; i++) {
    if (bottle[i]['key']==index) {
        setState(() {

            isFind=true;

```

```

        bottle[i]['ML']=controller.text;
    });
    break;

    }
}
if (isFind) {
}else{
    bottle.add(bottleNew);
}
}
}

    } ,
    decoration: InputDecoration(
        labelText: label,
        prefixIcon: Icon(icon),
        filled: true,
        fillColor: Colors.white,
        border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
        ),
    ),
),
),
);
}

Widget _buildDateField(TextEditingController controller, String
label, IconData icon, VoidCallback onTap) {
    return Padding(
        padding: const EdgeInsets.only(bottom: 10),
        child: TextField(
            controller: controller,
            readOnly: true,
            onTap: onTap,
            decoration: InputDecoration(
                labelText: "Input Date Time",
                prefixIcon: Icon(icon),
                filled: true,
                fillColor: Colors.white,
                border: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                ),
            ),
        ),
    );
}

Widget _buildTimeField(TextEditingController controller, String
label, IconData icon,int index,String Object) {
    return Padding(
        padding: const EdgeInsets.only(bottom: 10),
        child: TextField(
            controller: controller,
            readOnly: true,
            onTap: () {

                _selectTime(controller,index,Object,label);
            },

```

```

        decoration: InputDecoration(
          labelText: label,
          prefixIcon: Icon(icon),
          filled: true,
          fillColor: Colors.white,
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
          ),
        ),
      ),
    ),
  );
}

Widget _buildDropdownField(dynamic index,String label, IconData
icon, List<String> options, String currentValue,
ValueChanged<String?> onChanged,String Object ) {
  return Padding(
    padding: const EdgeInsets.only(bottom: 10),
    child: InputDecorator(

      decoration: InputDecoration(
        labelText: label,

        prefixIcon: Icon(icon),
        filled: true,
        fillColor: Colors.white,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(10),
        ),
      ),
      child: DropdownButtonHideUnderline(
        child: DropdownButton<String>(
          value: currentValue,
          isDense: true,

          onChanged: (value){
if (Object=='Meals') {

Map<String,dynamic> mealsNew={
'key':index,
'Food':"None",
'Comment':"None",
'Quantity':'None',

};
if (meals.length<=0) {

  meals.add(mealsNew);
}else{
}
bool isFind=false;
for (var i = 0; i < meals.length; i++) {
  if (meals[i]['key']==index) {
    setState(() {

      isFind=true;
      print(value);

```

```

        meals[i]['Quantity']=value;
    });

    break;

}
}
if (isFind) {
}else{
    meals.add(mealsNew);
}

}

if (Object=="Toilet") {

Map<String,dynamic> toiletNew={
'key':index,
'waktu':"none",
'type':"none",
'kondisi':'none',
'notes':'none',
};
if (toilet.length<=0) {
    toilet.add(toiletNew);
}
bool isFind=false;
for (var i = 0; i < toilet.length; i++) {
    if (toilet[i]['key']==index) {
        setState(() {

            isFind=true;
        });
        if (label=='Type') {
            toilet[i]['type']=value;
        }else
        if (label=='Condition') {
            toilet[i]['kondisi']=value;
        }
        break;

    }
}
if (isFind) {
}else{
    toilet.add(toiletNew);
}
}

// BOTTOL
if (Object=="Bottle") {

Map<String,dynamic> bottleNew={
'key':index,

```



```

'time':"none",
'ML':"none",
'type':'none',

});
if (bottle.length<=0) {
    bottle.add(bottleNew);
}
bool isFind=false;
for (var i = 0; i < bottle.length; i++) {
    if (bottle[i]['key']==index) {
        setState(() {

            isFind=true;
            bottle[i]['type']=value;
        });
        break;

    }
}
if (isFind) {
}else{
    bottle.add(bottleNew);
}
}

// ITEM NEED
if (Object=="ItemNeed") {
    var quantityFinal="1";

bool isFind=false;
for (int i = 0; i < itemneed.length; i++) {
    if (itemneed[i]['key']==index) {

        setState(() {
            isFind=true;
            itemneed[i]['quantity']=value.toString();
            quantityFinal=value.toString();
        });

        break;
    }

}

if (isFind) {
}else{
    // itemneed.add(itemneedNew);
}
}

    },
    items: options.map((String value) {
        return DropdownMenuItem<String>(
            value: value,
            child: Text(value),
        );
    }).toList(),
),
),

```

```

    ),
  ),
);
}

Widget _buildMealSection(String mealType, int index) {
  return Column(
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
      Padding(
        padding: const EdgeInsets.only(top: 10, bottom: 5),
        child: Text(
          mealType,
          style: TextStyle(
            fontSize: 16,
            fontWeight: FontWeight.bold,
            color: Colors.black87,
          ),
        ),
      ),
      _buildTextField(_mealFoodControllers[index], 'Food',
Icons.restaurant_menu, index, false, "Meals"),
      _buildDropdownField(index,
        'Quantity',
        Icons.restaurant,
        ['None', 'Some', 'Lots'],
        _mealQuantities[index],
        (newValue) {
          setState(() {
            Map<String, dynamic> Item={
'key':index,
'Food':"none",
'Comment':"none",
'Quantity':'none',
};
            if (meals.length<=0) {
              meals.add(Item);
            }
            bool isFind=false;
            for (var i = 0; i < meals.length; i++) {
              if (meals[i]['key']==index) {
                meals[i]['Quantity']=newValue;
              } else {
                // meals.add(Item);
              }
            }
          }
        },
        _mealQuantities[index] = newValue!;
      ),
      _buildTextField(_mealCommentControllers[index], 'Comment',
Icons.comment, index, false, "Meals"),
      SizedBox(height: 10),
    ],
  );
}

```

```

}

Widget _buildToiletEntry(int index) {
  final ToiletEntry toiletEntry = _toiletEntries[index];
  return Column(
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
      Padding(
        padding: const EdgeInsets.only(top: 10, bottom: 5),
        child: Text(
          'Toilet Entry ${index + 1}',
          style: TextStyle(
            fontSize: 16,
            fontWeight: FontWeight.bold,
            color: Colors.black87,
          ),
        ),
      ),
      _buildTimeField(toiletEntry.timeController, 'Time',
Icons.access_time,index,"Toilet"),
      _buildDropdownField(index,
        'Type',
        Icons.wc,
        ['Diaper', 'Potty'],
        toiletEntry.type,
        (newValue) {
          setState(() {
            toiletEntry.type = newValue!;
          });
        }, "Toilet"
      ),
      _buildDropdownField(index,
        'Condition',
        Icons.wc,
        ['DR', 'Wet', 'BM'],
        toiletEntry.condition,
        (newValue) {
          setState(() {
            toiletEntry.condition = newValue!;
          });
        }, "Toilet"
      ),
      _buildTextField(toiletEntry.notesController, 'Notes',
Icons.notes,index,false,"Toilet"),
      SizedBox(height: 10),
    ],
  );
}

Widget _buildBottleEntry(int index) {
  final BottleEntry bottleEntry = _bottleEntries[index];
  return Column(
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
      Padding(
        padding: const EdgeInsets.only(top: 10, bottom: 5),
        child: Text(
          'Bottle Entry ${index + 1}',

```

```

        style: TextStyle(
          fontSize: 16,
          fontWeight: FontWeight.bold,
          color: Colors.black87,
        ),
      ),
    ),
    _buildTimeField(bottleEntry.timeController, 'Time',
Icons.access_time,index,"Bottle"),
    _buildTextField(bottleEntry.mlController, 'ML',
Icons.local_drink,index,false,'Bottle'),
    _buildDropdownField(index,
      'Type',
      Icons.local_drink,
      ['Breast', 'Formula'],
      bottleEntry.type,
      (newValue) {
        setState(() {
          bottleEntry.type = newValue!;
        });
      }, "Bottle"
    ),
    SizedBox(height: 10),
  ],
);
}

Widget _buildVitaminEntry(int index) {
  final VitaminEntry vitaminEntry = _vitaminEntries[index];
  return Column(
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
      Padding(
        padding: const EdgeInsets.only(top: 10, bottom: 5),
        child: Text(
          'Vitamin Entry ${index + 1}',
          style: TextStyle(
            fontSize: 16,
            fontWeight: FontWeight.bold,
            color: Colors.black87,
          ),
        ),
      ),
      _buildTextField(vitaminEntry.nameController, 'Vitamin
Name', Icons.local_pharmacy,index,false,'Vitamin'),
      _buildTextField(vitaminEntry.amountController, 'Amount',
Icons.local_offer,index,false,'Vitamin'),
      _buildTimeField(vitaminEntry.timeController, 'Time',
Icons.access_time,index,"Vitamin"),
      SizedBox(height: 10),
    ],
  );
}
}

```

B. data_anak.dart

```

import 'dart:async';
import 'dart:convert';
import 'dart:ffi';

import 'package:daycare_app/input_data_anak.dart';
import 'package:flutter/material.dart';

import 'activity_input.dart'; // Pastikan file activity_input.dart
diimpor
import 'package:auto_spacing/auto_spacing.dart';
import 'package:http/http.dart' as http;
class DataAnakScreen extends StatefulWidget {
  final List<dynamic>user;
  DataAnakScreen({required this.user});

  @override
  _DataAnakScreenState createState() =>
  _DataAnakScreenState(anak:this.user);
}

class _DataAnakScreenState extends State<DataAnakScreen> {
List<dynamic> anak;
  _DataAnakScreenState({required this.anak});
  // List<String> childrenNames = ['Anak 1', 'Anak 2', 'Anak 3'];
Future<void>selectDataPengasuh()async{
  var response=await
http.post(Uri.parse('http://10.0.2.2/daycare_api/get_Data_anak_peng
asuh.php'),body: {
  'id_pengasuh':anak[0]['id'],
});
  try {
    if (response.statusCode==200||response.statusCode==201) {
      setState(() {

        anak=jsonDecode(response.body);
      });

      print(anak);
    }
    else{
      print("gaada");
      print(jsonDecode(response.body));
    }
  } catch (e) {
    print(e);
  }
}
Future<void> getDataAnak()async{
  final $response=await
http.get(Uri.parse('http://10.0.2.2/daycare_api/get_Data_anak.php')
);
  setState(() {

    anak=jsonDecode($response.body);
  });
}

```

```

@override

Widget build(BuildContext context) {

return Scaffold(
  appBar: AppBar(
    title: Text(
      'Data Anak Daycare',
      style: TextStyle(
        fontSize: 24,
        fontWeight: FontWeight.bold,
        color: Colors.white,
      ),
    ),
    backgroundColor: Colors.orange,
  ),
  body: Center(child: Column(mainAxisAlignment:
MainAxisAlignment.center, children: [
    Padding(
      padding: const EdgeInsets.all(8.0),
      child: TextButton(style:
TextButton.styleFrom(backgroundColor: Colors.orange), onPressed: ()
async{
        await getDataAnak();
        print("cekanak");
        // print(anak);
        if (anak.length<0) {

        }
        else{

        Navigator.push(context,MaterialPageRoute(builder:
(context)=>ListAnak(anak: anak)));
        }
        }, child: const Text("Lihat Semua Anak",style:
TextStyle(color: Colors.white,fontSize:20 ))),
      ),
      // Padding(
      //   padding: const EdgeInsets.all(10.0),
      //   child:
      //     TextButton(style: TextButton.styleFrom(backgroundColor:
Colors.orange), onPressed: () async{
      //       await selectDataPengasuh();
      //       print("cekanak");
      //       // print(anak);
      //       if (anak.length<0) {

      //       }
      //       else{

      //       Navigator.push(context,MaterialPageRoute(builder:
(context)=>ListAnak(anak: anak)));
      //       }
      //     }, child: const Text("",style: TextStyle(color:
Colors.white,fontSize:20 ))),
      //   ),
      ],)),

```

```

        // Button gausah dipake karna hanya orang tua saja yang bisa
        daftar anak
        // floatingActionButton: FloatingActionButton(
        //   onPressed: getDataAnak,
        //   backgroundColor: Colors.orange,
        //   child: Icon(Icons.add),
        // ),
      );
    }
  }

class ListAnak extends StatefulWidget {
  ListAnak({
    required this.anak,
  });

  final List<dynamic> anak;

  @override
  State<ListAnak> createState() => _ListAnakState();
}

class _ListAnakState extends State<ListAnak> {

  @override

  @override
  Widget build(BuildContext context) {
    if (widget.anak.length>0) {

      return Scaffold(
        appBar:
        AppBar(
          backgroundColor: Colors.orange,
          foregroundColor: Colors.white,
          title:
          Text('Lihat Semua Data'),),
        body: ListView.builder(
          padding: const EdgeInsets.all(8.0),
          itemCount: widget.anak.length,
          itemBuilder: (context, index) {
            // getDataAnak();

            return Card(
              elevation: 4.0,
              margin: EdgeInsets.symmetric(vertical: 8.0),
              child: ListTile(
                subtitle: Text(widget.anak[index]['updated_at']),
                leading: CircleAvatar(
                  backgroundColor: Colors.orange,
                  child: Icon(Icons.child_care, color: Colors.white),
                ),
                title: Text(
                  widget.anak[index]['nama_lengkap'],
                  style: TextStyle(

```

```

        fontSize: 18,
      ),
    ),
    trailing: IconButton(
      icon: Icon(Icons.edit, color: Colors.orange),
      onPressed: ()
    {
      // print(widget.anak[index]['id']);
// print(widget.anak[index]);
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) => ActivityInputScreen(
            id_anak: widget.anak[index]['id'], nama_anak:
            widget.anak[index]['nama_lengkap'], id_pengasuh:
            widget.anak[index]['id_pengasuh']
          )),
        ),
      ),
    );
  },
),);
} else {
  return Scaffold(

    appBar:
    AppBar(
      backgroundColor: Colors.orange,
      foregroundColor: Colors.white,
      title:
      Text('Lihat Semua Data'),
    ),
    body:
    Center(child: Column(mainAxisAlignment:
    MainAxisAlignment.center, children: [Icon(Icons.close, size:
    75, color: Colors.red), Text('404 Data Not Found...')],),
    );
  }
}
}

```

C. input_data_anak.dart

```

import 'dart:convert';

import 'package:daycare_app/login.dart';
import 'package:daycare_app/menu.dart';
import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;
class InputDataAnakScreen extends StatefulWidget {
  final List user;
  InputDataAnakScreen({required this.user});
  @override

```



```

    _InputDataAnakScreenState createState() =>
    _InputDataAnakScreenState();
}

class _InputDataAnakScreenState extends State<InputDataAnakScreen>
{
    final TextEditingController _namaController =
    TextEditingController();
    final TextEditingController _tanggalLahirController =
    TextEditingController();
    final TextEditingController _alamatController =
    TextEditingController();
    final TextEditingController _kontakNamaController =
    TextEditingController();
    final TextEditingController _hubunganController =
    TextEditingController();
    final TextEditingController _nomorTeleponController =
    TextEditingController();
    final TextEditingController _riwayatMedisController =
    TextEditingController();
    String? _selectedJenisKelamin;

    // void _submitData(BuildContext context) {
    //     // Logika untuk menyimpan data anak
    //     ScaffoldMessenger.of(context).showSnackBar(
    //         SnackBar(content: Text('Data anak berhasil disimpan')),
    //     );

    //     // Navigasi ke halaman laporan aktivitas anak
    //     Navigator.push(
    //         context,
    //         MaterialPageRoute(builder: (context) =>
    LaporanActivityAnakScreen()),
    //     );
    // }

    late bool status=false;
    Future<void> InputAnak()async{

    // if (_namaController.text==' ' ||
    _alamatController.text==' ' || _hubunganController.text==' ' || _kontakNa
    maController.text==' ' || _tanggalLahirController.text==' ' ) {
    if (_namaController.text==' ' ) {
        print("Isi Semua Kolom...");
    } else {
        try {
            final String id="1";
            final response = await
http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_anak.php
'),body: {
"nama_lengkap": _namaController.text,
"tanggal_lahir": _tanggalLahirController.text,
"jenis_kelamin": _selectedJenisKelamin,
"alamat_rumah": _alamatController.text,
"nama_darurat": _kontakNamaController.text,
"hubungan_darurat": _hubunganController.text,

```

```

"nomor_darurat":_nomorTeleponController.text,
"riwayat_medis":_riwayatMedisController.text,
"id_orangtua":widget.user[0]['id'].toString(),
"id_pengasuh":"'2',
    });

    print(response.body);
    var $result=jsonDecode(response.body);
    //
print("-----
----");
    if($result["status"]=="Berhasil"){
print("MASUKKK");
setState(() {

    status=true;
});
    }else{
    print("GAK MASUKKKK");
    setState(() {

    status=false;
    });
    }

    } catch (e) {
    print("GA MASUKK");
    print(e);
    }
}

}

Future<void> _selectDate(BuildContext context) async {
    DateTime? pickedDate = await showDatePicker(
        context: context,
        initialDate: DateTime.now(),
        firstDate: DateTime(2000),
        lastDate: DateTime(2101),
        builder: (BuildContext context, Widget? child) {
            return Theme(
                data: ThemeData.light().copyWith(
                    primaryColor: Colors.orange,
                    colorScheme: ColorScheme.light(primary: Colors.orange,
secondary: Colors.orange),
                    buttonTheme: ButtonThemeData(textTheme:
ButtonTextTheme.primary),
                ),
                child: child!,
            );
        },
    );
    if (pickedDate != null) {
        setState(() {
            _tanggalLahirController.text =
"$${pickedDate.toLocal()}"
.split(' ')[0];

```

```

        print(_tanggalLahirController.text);
    });
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(
        'Input Data Anak',
        style: TextStyle(
          fontSize: 24,
          fontWeight: FontWeight.bold,
          color: Colors.white,
        ),
      ),
    ),
    backgroundColor: Colors.orange,
  ),
  body: Padding(
    padding: const EdgeInsets.all(16.0),
    child: SingleChildScrollView(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: <Widget>[
          _buildSectionTitle('Informasi Pribadi Anak'),
          _buildTextField(_namaController, 'Nama lengkap',
Icons.person),
          _buildDateField(context),
          _buildDropdownJenisKelamin(),
          _buildTextField(_alamatController, 'Alamat rumah',
Icons.home),
          SizedBox(height: 20),
          _buildSectionTitle('Informasi Kontak Darurat'),
          Row(
            children: <Widget>[
              Expanded(
                child: _buildTextField(_kontakNamaController,
'Nama kontak', Icons.person),
              ),
              SizedBox(width: 10),
              Expanded(
                child: _buildTextField(_hubunganController,
'Hubungan', Icons.family_restroom),
              ),
            ],
          ),
          _buildTextField(_nomorTeleponController, 'Nomor
telepon darurat', Icons.phone),
          SizedBox(height: 20),
          _buildSectionTitle('Riwayat Medis'),
          _buildTextField(_riwayatMedisController, 'Riwayat
medis anak', Icons.medical_services),
          SizedBox(height: 30),
          ElevatedButton(
            onPressed: () async{
              // print( _namaController.text );

```

```

// print( _hubunganController.text );
// print( _nomorTeleponController.text );
// print( _riwayatMedisController.text );
// print(_selectedJenisKelamin);
        await InputAnak();
        if (status) {
            Navigator.pop(context);
            print("BERRHASSSIIILLLLLLLL");
        }
        else{
            print("SGGGAAGGAAALLLLL");
        }
    },
    style: ElevatedButton.styleFrom(
        padding: EdgeInsets.symmetric(vertical: 15),
        shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(20),
        ),
        backgroundColor: Colors.orange,
    ),
    child: Text(
        'Simpan',
        style: TextStyle(fontSize: 18, color:
Colors.white),
    ),
),
),
),
),
),
);
}

Widget _buildSectionTitle(String title) {
    return Padding(
        padding: const EdgeInsets.only(bottom: 10),
        child: Text(
            title,
            style: TextStyle(
                fontSize: 18,
                fontWeight: FontWeight.bold,
                color: Colors.orange,
            ),
        ),
    );
}

Widget _buildTextField(TextEditingController controller, String
label, IconData icon) {
    return Padding(
        padding: const EdgeInsets.only(bottom: 10),
        child: TextField(
            controller: controller,
            decoration: InputDecoration(
                labelText: label,
                prefixIcon: Icon(icon),
                filled: true,
                fillColor: Colors.white,
            ),
        ),
    );
}

```

```

        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(10),
        ),
      ),
      style: TextStyle(fontSize: 16), // Add this line to ensure
consistency
    ),
  );
}

```

```

Widget _buildDropDownJenisKelamin() {
  return Padding(
    padding: const EdgeInsets.only(bottom: 10),
    child: DropdownButtonFormField<String>(
      value: _selectedJenisKelamin,
      decoration: InputDecoration(
        labelText: 'Jenis kelamin',
        prefixIcon: Icon(Icons.transgender),
        filled: true,
        fillColor: Colors.white,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(10),
        ),
      ),
      items: ['Laki-laki', 'Perempuan']
        .map((label) => DropdownMenuItem(
          child: Text(label, style: TextStyle(fontSize: 16, color:
Colors.black)),
          value: label,
        ))
        .toList(),
      onChanged: (value) {
        setState(() {
          _selectedJenisKelamin = value;
        });
      },
      style: TextStyle(fontSize: 16, color: Colors.black),
    ),
  );
}

```

```

Widget _buildDateField(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.only(bottom: 10),
    child: TextField(
      controller: _tanggalLahirController,
      decoration: InputDecoration(
        labelText: 'Tanggal lahir',
        prefixIcon: Icon(Icons.calendar_today),
        filled: true,
        fillColor: Colors.white,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(10),
        ),
      ),
      readOnly: true,
      onTap: () => _selectDate(context),
      style: TextStyle(fontSize: 16), // Add this line to ensure

```

```

consistency
    ),
  );
}
}

```

D. laporan_activity_anak.dart

```

import 'dart:convert';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';
import 'package:pie_chart/pie_chart.dart';

class LaporanActivityAnakScreen extends StatefulWidget {
  final String id_anak;

  LaporanActivityAnakScreen({required this.id_anak});

  @override
  State<LaporanActivityAnakScreen> createState() =>
    _LaporanActivityAnakScreenState();
}

class _LaporanActivityAnakScreenState extends
  State<LaporanActivityAnakScreen> {
  late Map<String, dynamic> jsonData = {};

  Future<void> selectDataLaporan() async {
    var response = await http.post(
      Uri.parse('http://10.0.2.2/daycare_api/get_Data_laporan_pengasuh.php'),
      body: {'id_anak': widget.id_anak},
    );
    try {
      if (response.statusCode == 200 || response.statusCode == 201)
      {
        setState(() {
          jsonData = jsonDecode(response.body);
        });
      } else {
        print("Data not found");
      }
    } catch (e) {
      print(e);
    }
  }

  @override
  void initState() {
    super.initState();
    selectDataLaporan();
  }
}

```

```

@override
Widget build(BuildContext context) {
  if (jsonData.isEmpty) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Laporan Aktivitas Anak'),
        backgroundColor: Colors.orange,
      ),
      body: Center(child: CircularProgressIndicator()),
    );
  }

  final dataMap = <String, double>{
    "Meals": jsonData['meal'].length.toDouble() ,
    "Toilet": jsonData['toilet'].length.toDouble() ,
    "Rest": jsonData['rest'].length.toDouble() ,
    "Bottle": jsonData['rest'].length.toDouble() ,
    "Vitamin": jsonData['vitamin'].length.toDouble() ,
    "Item Needs": jsonData['item'].length.toDouble() ,
  };

  final colorList = <Color>[
    Colors.greenAccent,
    const Color.fromARGB(255, 30, 94, 63),
    Color.fromARGB(255, 70, 2, 196),
    Color.fromARGB(255, 112, 7, 87),
    Color.fromARGB(205, 157, 31, 31),
    Color.fromARGB(151, 9, 106, 141),
  ];

  return Scaffold(
    appBar: AppBar(
      title: Text('Laporan Aktivitas Anak'),
      backgroundColor: Colors.orange,
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: SingleChildScrollView(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Container(
              padding: EdgeInsets.symmetric(horizontal: 16),
              child: PieChart(
                dataMap: dataMap,
                chartType: ChartType.ring,
                baseChartColor: Colors.grey[300]!,
                colorList: colorList,
              ),
            ),
            _buildInfoHeader(),
            SizedBox(height: 20),
            _buildMealsSection(),
            SizedBox(height: 20),
            _buildToiletSection(),
            SizedBox(height: 20),
            _buildRestSection(),
            SizedBox(height: 20),
          ],
        ),
      ),
    ),
  );
}

```

```

        _buildBottleSection(),
        SizedBox(height: 20),
        _buildShowerSection(),
        SizedBox(height: 20),
        _buildVitaminSection(),
        SizedBox(height: 20),
        _buildItemNeedsSection(),
      ],
    ),
  ),
);
}

Widget _buildInfoHeader() {
  return Container(
    padding: EdgeInsets.all(16.0),
    decoration: BoxDecoration(
      color: Colors.white,
      borderRadius: BorderRadius.circular(10),
      boxShadow: [
        BoxShadow(
          color: Colors.grey.withOpacity(0.5),
          spreadRadius: 1,
          blurRadius: 5,
          offset: Offset(0, 3),
        ),
      ],
    ),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        _buildInfoRow('Nama', jsonData['laporan'][0]['id_anak']
        ?? ''),
        _buildInfoRow('Arrival',
        jsonData['laporan'][0]['arrival_time'] ?? ''),
        _buildInfoRow('Kondisi',
        jsonData['laporan'][0]['kondisi'] ?? ''),
        _buildInfoRow('Tanggal',
        jsonData['laporan'][0]['timestamp'] ?? ''),
        _buildInfoRow('Suhu Tubuh',
        '${jsonData['laporan'][0]['temperature'] ?? ''}°C'),
      ],
    ),
  );
}

Widget _buildInfoRow(String title, String content) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 4.0),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Text(title, style: TextStyle(fontWeight:
        FontWeight.bold)),
        Text(content),
      ],
    ),
  );
}

```



```

    );
}

Widget _buildMealsSection() {
  return _buildSectionWithTable(
    'Makanan',
    jsonData['meal'] ?? [],
    ['comment', 'Food', 'Quantity'],
    (meal) => [
      DataCell(Text(meal['comment'] ?? '')),
      DataCell(Text(meal['food'] ?? '')),
      DataCell(Text(meal['quantity'] ?? '')),
    ],
  );
}

Widget _buildToiletSection() {
  return _buildSectionWithTable(
    'Toilet',
    jsonData['toilet'] ?? [],
    ['Waktu', 'Jenis', 'Kondisi'],
    (entry) => [
      DataCell(Text(entry['waktu'] ?? '')),
      DataCell(Text(entry['type'] ?? '')),
      DataCell(Text(entry['kondisi'] ?? '')),
    ],
  );
}

Widget _buildRestSection() {
  return _buildSectionWithTable(
    'Istirahat',
    jsonData['rest'] ?? [],
    ['Mulai Istirahat', 'Akhir Istirahat'],
    (entry) => [
      DataCell(Text(entry['start_time'] ?? '')),
      DataCell(Text(entry['end_time'] ?? '')),
    ],
  );
}

Widget _buildBottleSection() {
  return _buildSectionWithTable(
    'Botol',
    jsonData['bottle'] ?? [],
    ['Waktu', 'ML', 'Tipe'],
    (entry) => [
      DataCell(Text(entry['time'] ?? '')),
      DataCell(Text(entry['ML'] ?? '')),
      DataCell(Text(entry['type'] ?? '')),
    ],
  );
}

Widget _buildVitaminSection() {
  return _buildSectionWithTable(
    'Vitamin',
    jsonData['vitamin'] ?? [],

```

```

        ['Nama', 'Jumlah', 'Waktu'],
        (entry) => [
            DataCell(Text(entry['vitamin_name'] ?? '')),
            DataCell(Text(entry['amount'] ?? '')),
            DataCell(Text(entry['time'] ?? '')),
        ],
    );
}

Widget _buildShowerSection() {
    if (jsonData['shower'] != null && jsonData['shower'].isNotEmpty)
    {
        final showerData = jsonData['shower'][0]; // Assuming you want
        the first entry in the list

        return Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
                _buildSectionTitle('Mandi'),
                Padding(
                    padding: const EdgeInsets.symmetric(vertical: 4.0),
                    child: Row(
                        mainAxisAlignment: MainAxisAlignment.spaceBetween,
                        children: [
                            Text('Pagi', style: TextStyle(fontWeight:
FontWeight.bold)),
                            Text(showerData['morning_shower'] ?? '-', style:
TextStyle(fontSize: 16)),
                        ],
                    ),
                ),
                Padding(
                    padding: const EdgeInsets.symmetric(vertical: 4.0),
                    child: Row(
                        mainAxisAlignment: MainAxisAlignment.spaceBetween,
                        children: [
                            Text('Siang', style: TextStyle(fontWeight:
FontWeight.bold)),
                            Text(showerData['evening_shower'] ?? '-', style:
TextStyle(fontSize: 16)),
                        ],
                    ),
                ),
            ],
        );
    } else {
        return _buildSectionTitle('Mandi: Data not available');
    }
}

Widget _buildItemNeedsSection() {
    return Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
            _buildSectionTitle('Catatan Untuk Orang Tua'),
            for (var item in jsonData['item'] ?? [])
            Padding(
                padding: const EdgeInsets.symmetric(vertical: 4.0),
                child: Text('${item['key']} ?? ': ${item['quantity']}

```

```

?? '}',),
    ),
  ],
);
}

Widget _buildSectionWithTable(
  String title, List<dynamic> data, List<String> columns,
  Function(dynamic) buildCells) {
  return Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      _buildSectionTitle(title),
      DataTable(
        columns: columns.map((col) => DataColumn(label: Text(col,
style: TextStyle(fontWeight: FontWeight.bold)))).toList(),
        rows: data.map((entry) => DataRow(cells:
buildCells(entry))).toList(),
        headingRowColor:
MaterialStateProperty.resolveWith((states) => Colors.orange[100]),
        dataRowColor: MaterialStateProperty.resolveWith(
          (states) => states.contains(MaterialState.selected) ?
Colors.orange[50] : Colors.white,
        ),
        dividerThickness: 1,
        columnSpacing: 20,
      ),
    ],
  );
}

Widget _buildSectionTitle(String title) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 10.0),
    child: Text(
      title,
      style: TextStyle(
        fontSize: 18,
        fontWeight: FontWeight.bold,
        color: Colors.orange,
      ),
    ),
  );
}
}

```

E. listanak_orangtua.dart

```

import 'dart:convert';
import 'package:daycare_app/data_anak.dart';
import 'package:daycare_app/laporan_activity_anak.dart';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';

class ListanakOrangtua extends StatefulWidget {

```

```

final List<dynamic> user;

ListanakOrangtua({required this.user});

@override
State<ListanakOrangtua> createState() =>
_ListanakOrangtuaState();
}

class _ListanakOrangtuaState extends State<ListanakOrangtua> {
  late List anak = [];
  late bool statusGetAnak = false;

  Future<void> postToilet() async {
    try {
      final response = await http.post(
Uri.parse('http://10.0.2.2/daycare_api/get_Data_anak_orangtua.php')
,
        body: {
          "id_orangtua": widget.user[0]['id'],
          // "id_orangtua": widget.user[0]['id'],
        },
      );

      print(response.body);

      var result = jsonDecode(response.body);

      if (response.statusCode==200) {
        print("API berhasil digunakan...");
        setState(() {
          anak=result; // Menambahkan data anak ke dalam
List<Map<String, dynamic>>
          // print(anak[0]['nama_lengkap']);
          statusGetAnak = true;
        });
      } else {
        print("API gagal digunakan...");
        setState(() {
          statusGetAnak = false;
        });
      }
    } catch (e) {
      print("Terjadi error: $e");
      setState(() {
        statusGetAnak = false;
      });
    }
  }

  @override
  void initState() {
    postToilet();
    print('refresh');
    super.initState();
  }
}

```

```

@override
Widget build(BuildContext context) {
  // print(widget.user);
  return Scaffold(
    appBar: AppBar(
      title: Text('Daftar Anak'),
    ),
    body: ListView.builder(itemBuilder: (context,index){
      return Card(
        elevation: 4.0,
        margin: EdgeInsets.symmetric(vertical: 8.0),
        child: ListTile(
          subtitle: Text(anak[index]['updated_at']),
          leading: CircleAvatar(
            backgroundColor: Colors.orange,
            child: Icon(Icons.child_care, color: Colors.white),
          ),
          title: Text(
            anak[index]['nama_lengkap'],
            style: TextStyle(
              fontSize: 18,
            ),
          ),
          trailing: IconButton(
            icon: Icon(Icons.remove_red_eye, color:
Color.fromARGB(255, 5, 239, 36)),
            onPressed: () async{
              null;
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) =>
LaporanActivityAnakScreen(id_anak:anak[index]['id']),
                  // builder: (context) =>
LaporanActivityAnakScreen(),
                ),
              );
            },
          ),
        ),
      );
    },itemCount: anak.length,));
}
}

```

F. login.dart

```

import 'dart:convert';

import 'package:flutter/material.dart';
import 'register.dart';
import 'package:http/http.dart' as http;
import 'menu.dart'; // Import halaman MenuScreen

```

```

class LoginScreen extends StatefulWidget {

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final TextEditingController _usernameController =
    TextEditingController();

  final TextEditingController _passwordController =
    TextEditingController();

  late bool statusLogin=false;
  late List<dynamic> user;

  Future<void> Login()async{

    if (_usernameController.text!='' || _passwordController.text!='' )
    {
      print("Isi Semua Kolom...");
    } else {
      try {
        final response = await
http.post(Uri.parse('http://10.0.2.2/daycare_api/login.php'),body:
{
  "username":_usernameController.text,
  "password":_passwordController.text,
});
        // print(response.body);
        user=jsonDecode(response.body);

        print(user[0]['username']);
        if(user[0]['username']!=[]){
setState(() {

          statusLogin=true;
});
        }else{
          setState(() {

            statusLogin=false;
          });
        }

      } catch (e) {
        print(e);
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(

```

```

appBar: AppBar(
  title: Text(
    'Login',
    style: TextStyle(
      fontSize: 24,
      fontWeight: FontWeight.bold,
      color: Colors.white,
    ),
  ),
  backgroundColor: Colors.orange,
),
body: SingleChildScrollView(
  child: Padding(
    padding: const EdgeInsets.all(18.0),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: <Widget>[
        SizedBox(height: 60), // Memberi ruang antara AppBar
dan konten
        Image.asset(
          'images/gambar2.png',
          height: 150, // Adjust the height as needed
        ),
        SizedBox(height: 20),
        Text(
          'Login to continue',
          style: TextStyle(
            fontSize: 20,
            color: Colors.orange[700],
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 20),
        TextField(
          controller: _usernameController,
          decoration: InputDecoration(
            labelText: 'Username',
            prefixIcon: Icon(Icons.person),
          ),
        ),
        SizedBox(height: 20),
        TextField(
          controller: _passwordController,
          decoration: InputDecoration(
            labelText: 'Password',
            prefixIcon: Icon(Icons.lock),
          ),
          obscureText: true,
        ),
        SizedBox(height: 20),
        ElevatedButton(
          onPressed: () async {
            await Login(); // Panggil fungsi Login saat
tombol ditekan
            print("STATUS LOGIN USER : ");
            print(statusLogin);
            if (statusLogin) {

```


StatefulWidget yang menampilkan antarmuka login untuk pengguna. Menggunakan TextEditingController untuk input teks pada kolom username dan password.

2. Class _LoginScreenState

Memeriksa apakah input kosong dan melakukan panggilan HTTP POST ke server API lokal untuk verifikasi kredensial pengguna. Menggunakan jsonDecode untuk decode respons dari server dan memeriksa status login.

3. Lainnya :

- ElevatedButton: Tombol untuk mengirimkan kredensial pengguna dan memanggil fungsi login.
- Navigasi: Jika login berhasil, pengguna diarahkan ke MenuScreen dengan data pengguna. Jika tidak akan menampilkan pesan error.
- GestureDetector: Menyediakan button redirect untuk pengguna yang belum memiliki akun untuk mendaftar di RegisterScreen.

G. menu.dart

```
import 'dart:async';
import 'dart:convert';

import 'package:daycare_app/data_anak.dart';
import 'package:daycare_app/listanak_orangtua.dart';
import 'package:flutter/material.dart';
import 'input_data_anak.dart'; // Import halaman
InputDataAnakScreen
import 'package:http/http.dart' as http;
class MenuScreen extends StatefulWidget {
  final List<dynamic>user;

  MenuScreen({required this.user});
  @override
  State<MenuScreen> createState() => _MenuScreenState();
}

class _MenuScreenState extends State<MenuScreen> {

  @override
  Widget build(BuildContext context) {
    if (widget.user[0]['isAdmin'] == '0') {
      // ORANG TUA
```

```

        return TampilanOrangTua(widget: widget);
    } else{
// ADMIN
return DataAnakScreen(user: widget.user);
    }
}
}

class TampilanOrangTua extends StatefulWidget {
    late List anak;
    TampilanOrangTua({
        super.key,

        required this.widget,
    });

    final MenuScreen widget;

    @override
    State<TampilanOrangTua> createState() =>
    _TampilanOrangTuaState();
}

class _TampilanOrangTuaState extends State<TampilanOrangTua> {
    bool statusSelect=false;
    late List<dynamic>resultanak;

    @override
    Widget build(BuildContext context) {

        return Scaffold(
            appBar: AppBar(
                title: Text(
                    'Menu',
                    style: TextStyle(
                        fontSize: 24,
                        fontWeight: FontWeight.bold,
                        color: Colors.white,
                    ),
                ),
            backgroundColor: Colors.orange,
        ),
        body: SingleChildScrollView(
            child: Padding(
                padding: const EdgeInsets.all(18.0),
                child: Column(
                    crossAxisAlignment: CrossAxisAlignment.center, //
Center the content horizontally
                    children: <Widget>[
                        SizedBox(height: 150), // Memberi ruang antara AppBar
dan konten
                        Center(
                            child: Image.asset(
                                'images/gambar2.png',
                                height: 200, // Adjust the height as needed
                            ),
                        ),
                    ],
                ),
            ),
        ),
    ),
);

```

```

    ),
    SizedBox(height: 20),
    Center(child: Text("Welcome
    ${widget.user[0]['username']}"),),
    SizedBox(height: 20),
    ElevatedButton(
      onPressed: () {

        Navigator.push(
          context,
          MaterialPageRoute(builder: (context) =>
InputDataAnakScreen(user: widget.user)),
        );
      },
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.orange,
        foregroundColor: Colors.white,
        minimumSize: Size(double.infinity, 50), // Make
button stretch horizontally
      ),
      child: Container(
        width: MediaQuery.of(context).size.width * 0.8,
// Adjust the width
        padding: EdgeInsets.all(16),
        child: Text(
          'Daftarkan Anak Anda',
          textAlign: TextAlign.center,
          style: TextStyle(
            fontSize: 18,
          ),
        ),
      ),
    ),
    SizedBox(height: 20),
    ElevatedButton(
      onPressed: () {
        Navigator.push(
          context,
          MaterialPageRoute(builder: (context) =>
ListanakOrangtua(user: widget.user)),
          // MaterialPageRoute(builder: (context) =>
ListanakOrangtua(user: widget.user)),
        );
      },
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.orange,
        foregroundColor: Colors.white,
        minimumSize: Size(double.infinity, 50), // Make
button stretch horizontally
      ),
      child: Container(
        width: MediaQuery.of(context).size.width * 0.8,
// Adjust the width
        padding: EdgeInsets.all(16),
        child: Text(
          'Lihat Daftar Anak Anda',
          textAlign: TextAlign.center,
          style: TextStyle(

```

```

        ),
      ),
    ),
  ],
),
),
),
);
}
}

```

H. register.dart

```

import 'dart:convert';
import 'package:daycare_app/login.dart';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';

class RegisterScreen extends StatefulWidget {
  @override
  State<RegisterScreen> createState() => _RegisterScreenState();
}

class _RegisterScreenState extends State<RegisterScreen> {
  final TextEditingController _usernameController =
    TextEditingController();

  final TextEditingController _passwordController =
    TextEditingController();

  final TextEditingController _confirmPasswordController =
    TextEditingController();

  bool isRegister=false;

  void _register(BuildContext context) {
    final username = _usernameController.text;
    final password = _passwordController.text;
    final confirmPassword = _confirmPasswordController.text;

  }

  Future<void> SubmitRegister()async{

if (_usernameController.text==" " || _passwordController.text==" " ||
_confirmPasswordController.text!=_passwordController.text) {
  print("Isi Semua Kolom...");
} else {
  try {
    final response = await
http.post(Uri.parse('http://10.0.2.2/daycare_api/post_Data_user.php

```

```

'),body: {
  "username":_usernameController.text,
  "password":_passwordController.text,
  });

  var $result=jsonDecode(response.body);
  if($result["status"]=="Berhasil"){
    setState(() {
      isRegister=true;
    });
  }

  } catch (e) {
    print(e);
  }
}

}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(
        'Register',
        style: TextStyle(
          fontSize: 24,
          fontWeight: FontWeight.bold,
          color: Colors.white,
        ),
      ),
      backgroundColor: Colors.orange,
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          TextField(
            controller: _usernameController,
            decoration: InputDecoration(
              labelText: 'Email',
              prefixIcon: Icon(Icons.email),
            ),
          ),
          SizedBox(height: 20),
          TextField(
            controller: _passwordController,
            decoration: InputDecoration(
              labelText: 'Password',
              prefixIcon: Icon(Icons.lock),
            ),
            obscureText: true,
          ),
          SizedBox(height: 20),
          TextField(

```

```

        controller: _confirmPasswordController,
        decoration: InputDecoration(
          labelText: 'Confirm Password',
          prefixIcon: Icon(Icons.lock),
        ),
        obscureText: true,
      ),
      SizedBox(height: 30),
      ElevatedButton(
        onPressed: ()async{

          await SubmitRegister();
          if (isRegister==true) {
            print('berhasil regis');
            setState(() {

              isRegister==false;
            });
            Navigator.push(context,
MaterialPageRoute(builder: (context)=>LoginScreen()));

          }
        },
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.orange,
          foregroundColor: Colors.white,
        ),
        child: Container(
          width: double.infinity,
          child: Text(
            'Register',
            textAlign: TextAlign.center,
            style: TextStyle(
              fontSize: 18,
            ),
          ),
        ),
      ),
    ],
  ),
);
}
}

```

Penjelasan:

1. Class RegisterScreen

StatefulWidget yang menampilkan antarmuka pendaftaran. Menggunakan TextEditingController untuk input teks pada kolom email, password, dan konfirmasi password.

2. SubmitRegister()

Fungsi asinkron yang memvalidasi input pengguna, mengirimkan permintaan POST ke server API lokal, dan memproses respons. Memeriksa apakah semua bidang terisi dan apakah password cocok dengan konfirmasi password. Jika respons dari server menunjukkan berhasil, status pendaftaran diatur menjadi true dan pengguna diarahkan ke halaman login.

3. Lainnya:

ElevatedButton: Tombol untuk mengirimkan data pendaftaran. Memanggil SubmitRegister() saat ditekan. Jika pendaftaran berhasil, pengguna diarahkan kembali ke halaman login.

I. main.dart

```
import 'package:daycare_app/register.dart';
import 'package:flutter/material.dart';
import 'login.dart';
void main() {
  runApp(MaterialApp(
    home: HomeScreen(),
  ));
}

class HomeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        color: Colors.amber[50], // Light grey background
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: [
            Image.asset(
              'images/gambar2.png', // Replace with your image path
              height: 250, // Adjust the height as needed
              fit: BoxFit.contain,
            ),
            SizedBox(height: 20),
            Text(
              'Welcome to Kids Daycare!',
              style: TextStyle(
                fontSize: 28,
                fontWeight: FontWeight.bold,
                color: Colors.brown[800], // Dark brown text color
              ),
              textAlign: TextAlign.center,
            ),
            SizedBox(height: 80),
            Center(
              child: ElevatedButton(
```

```

        onPressed: () {
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) =>
LoginScreen()),
          );
        },
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.orange,
          foregroundColor: Colors.white,
          minimumSize: Size(300, 60), // Set minimum width
and height
        ),
        child: Container(
          padding: EdgeInsets.all(16),
          child: Text(
            'Login',
            textAlign: TextAlign.center,
            style: TextStyle(
              fontSize: 18,
            ),
          ),
        ),
      ),
    ),
    SizedBox(height: 20),
    Center(
      child: ElevatedButton(
        onPressed: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => RegisterScreen()),
          );
        },
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.orange,
          foregroundColor: Colors.white,
          minimumSize: Size(300, 60), // Set minimum width
and height
        ),
        child: Container(
          padding: EdgeInsets.all(10),
          child: Text(
            'Register',
            textAlign: TextAlign.center,
            style: TextStyle(
              fontSize: 18,
            ),
          ),
        ),
      ),
    ),
  ],
),
);
}

```



```
}
```

Penjelasan:

4. Fungsi main()

Fungsi utama yang memulai aplikasi. Menggunakan runApp() untuk menjalankan aplikasi dengan MaterialApp yang menampilkan LoginScreen sebagai halaman awal.

5. Class HomeScreen

StatelessWidget yang menampilkan screen selamat datang untuk aplikasi.

6. Lainnya:

ElevatedButton: Dua tombol di bagian bawah, satu untuk navigasi ke halaman login (LoginScreen) dan satu lagi untuk halaman pendaftaran (RegisterScreen).

J. pubspec.yaml

```
dependencies:
  flutter:
    sdk: flutter
  http: ^1.2.1
  auto_spacing: ^0.0.7

flutter:
  uses-material-design: true
  assets:
    - images/
```

V. Source Code API

A. dbConnection.php

```
<?php
function connectdaycareDB()
{
    $con=mysqli_connect('localhost','root','','daycare_app');

    return $con;
}??>
```

B. delete_Data.php

```
<?php
function connectdaycareDB()
{
    $con=mysqli_connect('localhost','root','','daycare_app');

    return $con;
}??>
```

C. get_Data_anak.php

```
<?php
include ('dbConnection.php');
$con=connectdaycareDB();
$query='Select * from `anak`';
$exe=mysqli_query($con,$query);

$arr=[];
while ($row = mysqli_fetch_array($exe)) {
    $arr[]=$row;
}
header('Content-Type: application/json');
echo json_encode($arr,JSON_PRETTY_PRINT);
mysqli_close($con);
?>
```

D. get_Data_anak_orangtua.php

```
<?php
// Menggunakan file dbConnection.php untuk koneksi ke database
include('dbConnection.php');

// Fungsi untuk terhubung ke database
$con = connectdaycareDB();

// Memeriksa apakah parameter 'id_orangtua' ada dalam request POST
if (isset($_POST['id_orangtua'])) {
    $id_orangtua = $_POST['id_orangtua'];

    // Query SQL untuk mengambil data anak berdasarkan id_orangtua
    $query = "SELECT * FROM anak WHERE id_orangtua = 
'$id_orangtua'";
    $exe = mysqli_query($con, $query);

    // Array untuk menyimpan hasil query
    $arr = [];

    // Mengambil setiap baris hasil query dan menyimpannya dalam
```

```

array
    while ($row = mysqli_fetch_array($exe)) {
        $arr[] = $row;
    }

    // Mengirimkan respons dalam format JSON
    if (empty($arr)) {
        header('Content-Type: application/json');
        echo json_encode(["status" => "Gagal", "message" => "Data
anak tidak ditemukan"]);
    } else {
        header('Content-Type: application/json');
        echo json_encode($arr, JSON_PRETTY_PRINT);
    }
} else {
    // Jika parameter 'id_orangtua' tidak ada dalam request POST
    header('Content-Type: application/json');
    echo json_encode(["status" => "Gagal", "message" => "Parameter
'id_orangtua' tidak tersedia"]);
}

// Menutup koneksi ke database
mysqli_close($con);
?>

```

E. get_Data_anak_pengasuh.php

```

<?php
include ('dbConnection.php');
$con=connectdaycareDB();

if (isset($_POST['id_pengasuh'])) {
    $id_pengasuh=$_POST['id_pengasuh'];
    # code...
}else{
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}

if ($id_pengasuh!=null||$id_pengasuh!="") {
    # code...
    $query="Select * from `anak` where id_pengasuh='$id_pengasuh'";
    $exe=mysqli_query($con,$query);

    $arr=[];
    while ($row = mysqli_fetch_array($exe)) {
        $arr[]=$row;
    }
    header('Content-Type: application/json');
    echo json_encode($arr,JSON_PRETTY_PRINT);
}

mysqli_close($con);
?>

```

F. get_Data_laporan_pengasuh.php

```
<?php
// Menggunakan file dbConnection.php untuk koneksi ke database
include('dbConnection.php');

// Fungsi untuk terhubung ke database
$con = connectdaycareDB();
$result=[];
// Memeriksa apakah parameter 'id_orangtua' ada dalam request POST
if (1==1) {
    // $id_anak = $_POST['id_anak'];
    $querylaporan = "SELECT * FROM laporan_anak WHERE id_anak =
'25' ORDER BY id DESC LIMIT 1 " ;
    $exelaporan = mysqli_query($con, $querylaporan);
    $laporan=[];
    while ($row = mysqli_fetch_array($exelaporan)) {
        $laporan[] = $row;
        // print $row[];
    }
    // echo json_encode($laporan[0]['id']);
    $result+=['laporan'=> $laporan];
    // echo json_encode($result);
    $get_id= $laporan[0]["id"];
    // $get_id=11;

    // ----- MEALS
    $querymeals = "SELECT * FROM meals WHERE id_laporan ='$get_id'"
;
    $exemeals = mysqli_query($con, $querymeals);
    $meal=[];
    while ($row = mysqli_fetch_array($exemeals)) {
        $meal[] = $row;
        // print $row[];
    }
    // print("ADA");
    $result+=['meal'=> $meal];
    // echo json_encode($meal);

    // echo $meal[0]['id'];

    // ----- Toilet
    $querytoilet = "SELECT * FROM toilet WHERE id_laporan
='$get_id'" ;
    $exetoilet = mysqli_query($con, $querytoilet);
    $toilet=[];
    while ($row = mysqli_fetch_array($exetoilet)) {
        $toilet[] = $row;
        // print $row[];
    }
    // print("ADA");
    $result+=['toilet'=> $toilet];
    // echo json_encode($toilet);
}
```

```

// ----- rest
$queryrest = "SELECT * FROM rest WHERE id_laporan ='$get_id' " ;
$exerest = mysqli_query($con, $queryrest);
$rest=[];
while ($row = mysqli_fetch_array($exerest)) {
    $rest[] = $row;
    // print $row[];
    // print("ADA");
}
$result+=[ 'rest'=> $rest];
// echo json_encode($rest);

// ----- bottle
$querybottle = "SELECT * FROM bottle WHERE id_laporan
='$get_id' " ;
$exeibottle = mysqli_query($con, $querybottle);
$bottle=[];
while ($row = mysqli_fetch_array($exeibottle)) {
    $bottle[] = $row;
    // print $row[];
    // print("ADA");
}
$result+=[ 'bottle'=> $bottle];
// echo json_encode($bottle);

// echo $bottle[0]['id'];

// ----- vitamin
$queryvitamin = "SELECT * FROM vitamin WHERE id_laporan
='$get_id' " ;
$exevitamin = mysqli_query($con, $queryvitamin);
$vitamin=[];
while ($row = mysqli_fetch_array($exevitamin)) {
    $vitamin[] = $row;
    // print $row[];
    // print("ADA");
}
$result+=[ 'vitamin'=> $vitamin];
// echo json_encode($vitamin);

// echo $vitamin[0]['id'];

// ----- item
$queryitem = "SELECT * FROM item WHERE id_laporan ='$get_id' " ;
$exeitem = mysqli_query($con, $queryitem);
$item=[];
while ($row = mysqli_fetch_array($exeitem)) {
    $item[] = $row;
    // print $row[];
    // print("ADA");
}

```

```

        $result+=[ 'item'=> $item];
        // echo json_encode($item);

        // echo $item[0]['id'];

        header('Content-Type: application/json');

        echo json_encode($result,JSON_PRETTY_PRINT);

    }
    // Menutup koneksi ke database
    mysqli_close($con);
?>

```

G. get_Data_user.php

```

<?php
include ('dbConnection.php');
$con=connectdaycareDB();
$query='Select * from `user`';
$exe=mysqli_query($con,$query);

$arr=[];
while ($row = mysqli_fetch_array($exe)) {
    $arr[]=$row;
}
header('Content-Type: application/json');
echo json_encode($arr,JSON_PRETTY_PRINT);
mysqli_close($con);
?>

```

H. login.php

```

<?php
include('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['username'])) {
    $username = $_POST['username'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Username
tidak tersedia"]));
}
if (isset($_POST['password'])) {
    $password = $_POST['password'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Password
tidak tersedia"]));
}

```

```

}

// Query SQL untuk select data dari tabel user berdasarkan username
dan password
$query = "SELECT * FROM `user` WHERE `username`='$username' AND
`password`='$password'";
$result = mysqli_query($con, $query);
$exe=mysqli_query($con,$query);

// Memeriksa apakah query berhasil dijalankan atau tidak
$arr=[];
while ($row = mysqli_fetch_array($exe)) {
    $arr[]=$row;
}
header('Content-Type: application/json');
echo json_encode($arr,JSON_PRETTY_PRINT);

if ($exe==false) {
    echo json_encode($result);
    # code...
}

// Menutup koneksi ke database
mysqli_close($con);
?>

```

I. post_Data_anak.php

```

<?php
include ('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['nama_lengkap'])) {
    $nama_lengkap = $_POST['nama_lengkap'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama
lengkap tidak tersedia"]));
}
if (isset($_POST['tanggal_lahir'])) {
    $tanggal_lahir = $_POST['tanggal_lahir'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Tanggal
lahir tidak tersedia"]));
}
if (isset($_POST['jenis_kelamin'])) {
    $jenis_kelamin = $_POST['jenis_kelamin'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Jenis
kelamin tidak tersedia"]));
}

```

```

if (isset($_POST['alamat_rumah'])) {
    $alamat_rumah = $_POST['alamat_rumah'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Alamat
rumah tidak tersedia"]));
}
if (isset($_POST['nama_darurat'])) {
    $nama_darurat = $_POST['nama_darurat'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama kontak
darurat tidak tersedia"]));
}
if (isset($_POST['hubungan_darurat'])) {
    $hubungan_darurat = $_POST['hubungan_darurat'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Hubungan
dengan darurat tidak tersedia"]));
}
if (isset($_POST['nomor_darurat'])) {
    $nomor_darurat = $_POST['nomor_darurat'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nomor
telepon darurat tidak tersedia"]));
}
if (isset($_POST['riwayat_medis'])) {
    $riwayat_medis = $_POST['riwayat_medis'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Riwayat
medis tidak tersedia"]));
}
if (isset($_POST['id_orangtua'])) {
    $id_orangtua = $_POST['id_orangtua'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "ID orang
tua tidak tersedia"]));
}
if (isset($_POST['id_pengasuh'])) {
    $id_pengasuh = $_POST['id_pengasuh'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "ID orang
tua tidak tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel anak
$query = "INSERT INTO `anak` (`nama_lengkap`, `tanggal_lahir`,
`jenis_kelamin`, `alamat_rumah`, `nama_darurat`,
`hubungan_darurat`, `nomor_darurat`, `riwayat_medis`,
`id_orangtua`, `id_pengasuh`)
VALUES ('$nama_lengkap', '$tanggal_lahir',
'$jenis_kelamin', '$alamat_rumah', '$nama_darurat',
'$hubungan_darurat', '$nomor_darurat', '$riwayat_medis',
'$id_orangtua', '$id_pengasuh')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil disimpan"];
}

```



```

} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

J. post_Data_bottle.php

```

<?php
include('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['time'])) {
    $time = $_POST['time'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['ML'])) {
    $ML = $_POST['ML'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['type'])) {
    $type = $_POST['type'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel bottle
$query = "INSERT INTO `bottle` (`time`, `ML`, `type`, `id_laporan`)
VALUES ('$time', '$ML', '$type', '$id_laporan')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data bottle
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan

```

```

data bottle: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

K. post_Data_item.php

```

<?php
include('dbConnection.php'); // Sesuaikan dengan file
dbConnection.php Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['key'])) {
    $key = $_POST['key'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['quantity'])) {
    $quantity = $_POST['quantity'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel item
$query = "INSERT INTO `item` (`key`, `quantity`, `id_laporan`)
VALUES ('$key', '$quantity', '$id_laporan')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data item
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data item: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database

```

```
mysqli_close($con);  
?>
```

L. post_Data_laporananak.php

```
<?php  
include ('dbConnection.php'); // Pastikan file dbConnection.php  
sesuai dengan kebutuhan Anda  
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database  
  
// Memeriksa dan mengambil nilai dari $_POST  
if (isset($_POST['id_anak'])) {  
    $id_anak = $_POST['id_anak'];  
} else {  
    die(json_encode(["status" => "Gagal", "message" => "Id anak  
tidak tersedia"]));  
}  
if (isset($_POST['arrival_time'])) {  
    $arrival_time = $_POST['arrival_time'];  
} else {  
    die(json_encode(["status" => "Gagal", "message" => "arrival  
tidak tersedia"]));  
}  
if (isset($_POST['temperature'])) {  
    $temperature = $_POST['temperature'];  
} else {  
    die(json_encode(["status" => "Gagal", "message" => "temperatur  
tidak tersedia"]));  
}  
if (isset($_POST['kondisi'])) {  
    $kondisi = $_POST['kondisi'];  
} else {  
    die(json_encode(["status" => "Gagal", "message" => "kondisi  
tidak tersedia"]));  
}  
  
// Query SQL untuk menyisipkan data ke dalam tabel laporan_anak  
$query = "INSERT INTO `laporan_anak` (`id_anak`, `arrival_time`,  
`temperature`, `kondisi`)  
VALUES ('$id_anak', '$arrival_time', '$temperature',  
'$kondisi')";  
  
// Menjalankan query  
$exe = mysqli_query($con, $query);  
  
// Memeriksa apakah query berhasil dijalankan atau tidak  
if ($exe) {  
    // Mengambil id_anak yang baru saja dimasukkan  
    $new_id = mysqli_insert_id($con);  
  
    $response = ["status" => "Berhasil", "message" => "Data  
berhasil disimpan", "new_id" => $new_id];  
} else {  
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan  
data: " . mysqli_error($con)];  
}
```

```

}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

M. post_Data_meals.php

```

<?php
include ('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Periksa dan mengambil nilai dari $_POST
if (isset($_POST['food'])) {
    $food = $_POST['food'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['quantity'])) {
    $quantity = $_POST['quantity'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['comment'])) {
    $comment = $_POST['comment'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Item tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel user
$query = "INSERT INTO `meals` (`food`,
`quantity`, `comment`, `id_laporan`) VALUES ('$food', '$quantity',
'$comment', '$id_laporan')";
$exe = mysqli_query($con, $query);

// Periksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data: " . mysqli_error($con)];
}

```

```

}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

N. post_Data_rest.php

```

<?php
include ('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Periksa dan mengambil nilai dari $_POST
if (isset($_POST['start_time'])) {
    $start_time = $_POST['start_time'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Username
tidak tersedia"]));
}
if (isset($_POST['end_time'])) {
    $end_time = $_POST['end_time'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel user
$query = "INSERT INTO `rest` (`start_time`,`end_time`,`id_laporan`)
VALUES ('$start_time', '$end_time', '$id_laporan')";
$exe = mysqli_query($con, $query);

// Periksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);

```

```
?>
```

O. post_Data_shower.php

```
<?php
include('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Periksa dan mengambil nilai dari $_POST
if (isset($_POST['morning_shower'])) {
    $morning_shower = $_POST['morning_shower'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Mandi pagi
tidak tersedia"]));
}
if (isset($_POST['evening_shower'])) {
    $evening_shower = $_POST['evening_shower'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Mandi malam
tidak tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Mandi malam
tidak tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel shower
$query = "INSERT INTO `shower` (`morning_shower`, `evening_shower`,
`id_laporan`) VALUES ('$morning_shower', '$evening_shower',
'$id_laporan')";
$exe = mysqli_query($con, $query);

// Periksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data shower
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data shower: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>
```

P. post_Data_toilet.php

```

<?php
include ('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['waktu'])) {
    $waktu = $_POST['waktu'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "item tidak
tersedia"]));
}
if (isset($_POST['type'])) {
    $type = $_POST['type'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "item tidak
tersedia"]));
}
if (isset($_POST['kondisi'])) {
    $kondisi = $_POST['kondisi'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "item tidak
tersedia"]));
}
if (isset($_POST['catatan'])) {
    $catatan = $_POST['catatan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "item tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "item tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel user
$query = "INSERT INTO `toilet` (`waktu`,
`type`,`kondisi`,`catatan`,`id_laporan`) VALUES ('$waktu', '$type',
'$kondisi','$catatan','$id_laporan')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

Q. post_Data_user.php

```
<?php
include ('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['username'])) {
    $username = $_POST['username'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Username
tidak tersedia"]));
}
if (isset($_POST['password'])) {
    $password = $_POST['password'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel user
$query = "INSERT INTO `user` (`username`, `password`) VALUES
('$username', '$password')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>
```

R. post_Data_vitamin.php

```
<?php
include('dbConnection.php'); // Sesuaikan dengan file
dbConnection.php Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['vitamin_name'])) {
    $vitamin_name = $_POST['vitamin_name'];
}
```



```

} else {
    die(json_encode(["status" => "Gagal", "message" => "Nama
vitamin tidak tersedia"]));
}
if (isset($_POST['amount'])) {
    $amount = $_POST['amount'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Jumlah
tidak tersedia"]));
}
if (isset($_POST['time'])) {
    $time = $_POST['time'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Waktu tidak
tersedia"]));
}
if (isset($_POST['id_laporan'])) {
    $id_laporan = $_POST['id_laporan'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Waktu tidak
tersedia"]));
}

// Query SQL untuk menyisipkan data ke dalam tabel vitamin
$query = "INSERT INTO `vitamin` (`vitamin_name`, `amount`, `time`,
`id_laporan`) VALUES ('$vitamin_name', '$amount', '$time',
'$id_laporan')";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data vitamin
berhasil disimpan"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal menyimpan
data vitamin: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

S. put_Data_anak.php

```

<?php
include ('dbConnection.php');
$con=connectdaycareDB();
if (isset($_POST['id_anak'])) {
    $id_anak=$_POST['id_anak'];
}else{
    die(json_encode(["status" => "Gagal", "message" => "Item tidak

```

```

tersedia" ]));
    // $id_anak=null;
    // $id_anak=2;

}
if ($id_anak) {
    # code...
    $query="Select * from `anak` where `id`='$id_anak'";
    $exe=mysqli_query($con,$query);

    if ($exe ) {
        # code...
        $arr=[];
        while ($row = mysqli_fetch_array($exe)) {
            $arr[]=$row;
        }
        header('Content-Type: application/json');
        echo json_encode($arr,JSON_PRETTY_PRINT);
    } else {

        $arr=["status" => "Gagal", "message" => "Data ditemukan"];
        echo json_encode($arr,JSON_PRETTY_PRINT);
        // echo json_encode($arr,JSON_PRETTY_PRINT);
        # code...
    }
}
else {
    $arr=["status" => "Gagal", "message" => "Data tidak lengkap"];
    // $arr=['Status']['Gagal'];
    // $arr[[]]=['Gagal']
    echo json_encode($arr,JSON_PRETTY_PRINT);
}
mysqli_close($con);

?>

```

T. update_Data.php

```

<?php
include('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['username']) && isset($_POST['nama']) &&
isset($_POST['password'])) {
    $username = $_POST['username'];
    $nama = $_POST['nama'];
    $password = $_POST['password'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Data tidak
lengkap"]));
}

```

```

// Query SQL untuk melakukan update data di dalam tabel user
berdasarkan username
$query = "UPDATE `user` SET `nama`='$nama', `password`='$password'
WHERE `username`='$username'";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil diperbarui"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal
memperbarui data: " . mysqli_error($con)];
}

// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>

```

U. update_Data_laporananak.php

```

<?php
include('dbConnection.php'); // Pastikan file dbConnection.php
sesuai dengan kebutuhan Anda
$con = connectdaycareDB(); // Fungsi untuk koneksi ke database

// Memeriksa dan mengambil nilai dari $_POST
if (isset($_POST['username']) && isset($_POST['nama']) &&
isset($_POST['password'])) {
    $username = $_POST['username'];
    $nama = $_POST['nama'];
    $password = $_POST['password'];
} else {
    die(json_encode(["status" => "Gagal", "message" => "Data tidak
lengkap"]));
}

// Query SQL untuk melakukan update data di dalam tabel user
berdasarkan username
$query = "UPDATE `user` SET `nama`='$nama', `password`='$password'
WHERE `username`='$username'";
$exe = mysqli_query($con, $query);

// Memeriksa apakah query berhasil dijalankan atau tidak
if ($exe) {
    $response = ["status" => "Berhasil", "message" => "Data
berhasil diperbarui"];
} else {
    $response = ["status" => "Gagal", "message" => "Gagal
memperbarui data: " . mysqli_error($con)];
}

```

```
// Mengembalikan respons dalam format JSON
echo json_encode($response);

// Menutup koneksi ke database
mysqli_close($con);
?>
```

VI. Assets

Assets yang digunakan yaitu gambar2.png sebagai berikut :

```
Image.asset(
    'images/gambar2.png',
    height: 200, // Adjust the height as needed
),
```

