INTERNET OF THINGS

implementasi IoT pada pemanfaatan di Rumah Sakit dengan skema sistem seperti berikut :

Tujuan Tujuan dari praktikum ini adalah untuk:

- Membuat struktur database di Firebase untuk menyimpan data permintaan bantuan dari pasien.
- Mengimplementasikan sistem menggunakan ESP32 untuk menerima dan mengirim data ke Firebase.
- Membuat aplikasi Flutter untuk mengontrol sistem IoT di rumah sakit, khususnya untuk pasien dan perawat.

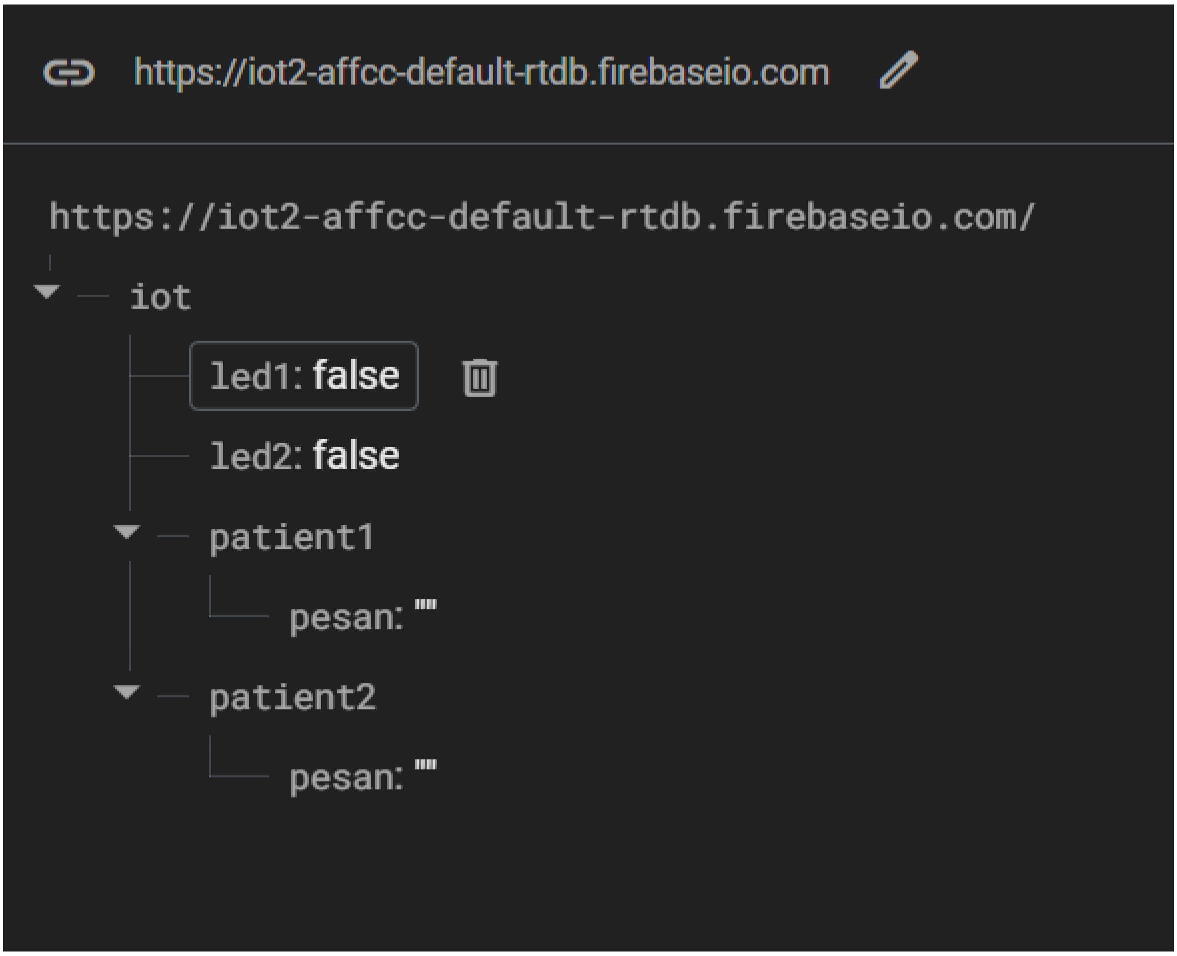
Metodologi Pada praktikum ini, implementasi IoT menggunakan dua komponen utama: perangkat keras (ESP32) dan aplikasi (Flutter). Langkah-langkah yang dilakukan adalah sebagai berikut:

Struktur Database Firebase

Pertama, kita buat struktur database Firebase yang berisi data tentang pasien yang sedang meminta bantuan. Berikut adalah struktur yang digunakan dalam Firebase:

json

```
{
  "iot": {
    "patient1": {
      "pesan": "pasien 1 meminta bantuan",
      "status": "menunggu",
      "led1": true
    },
    "patient2": {
      "pesan": "pasien 2 meminta bantuan",
      "status": "menunggu",
      "led2": true
    }
  }
}
```



https://iot2-affcc-default-rtdb.firebaseio.com

```
https://iot2-affcc-default-rtdb.firebaseio.com/
  iot
    led1: false
    led2: false
    patient1
      pesan: ""
    patient2
      pesan: ""
```

Koding ESP32

Selanjutnya, dilakukan pemrograman menggunakan ESP32 untuk menghubungkan sistem dengan Firebase dan mengontrol LED yang menandakan permintaan bantuan dari pasien. Berikut adalah kode yang digunakan pada ESP32:

```
#include <Arduino.h>

#include <WiFi.h>
#include <FirebaseESP32.h>

// WiFi credentials
#define WIFI_SSID "xrp"
#define WIFI_PASSWORD "mstahulhaq"

// Firebase credentials
#define API_KEY "AIzaSyBz7EAZbKYOCMELCun4rv9Oahvci0DRtos"
#define DATABASE_URL "https://iot2-affcc-default-rtdb.firebaseio.com/"

// LED configuration
#define LED1_PIN 2
```

```
#define LED2_PIN 4

// Define Firebase Data object
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

// Initialize LED states
bool patient1LedState = false;
bool patient2LedState = false;

void setup() {
 Serial.begin(115200);

 // Initialize LED pins
 pinMode(LED1_PIN, OUTPUT);
 pinMode(LED2_PIN, OUTPUT);

 // Connect to WiFi
 WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
 while (WiFi.status() != WL_CONNECTED) {
   delay(300);
 }
 Serial.println("Connected to WiFi");

 // Firebase configuration
 config.api_key = API_KEY;
 config.database_url = DATABASE_URL;
 auth.user.email = "ulhaq@gmail.com";
 auth.user.password = "ulhaq123";

 // Initialize Firebase
 Firebase.begin(&config, &auth);

 // Set SSL buffer size
 fbdo.setBSSLBufferSize(4096, 1024);

 // Initialize Firebase data (if not already initialized)
 FirebaseJson json;
 json.set("patient1/pesan", "pasien 1 meminta bantuan");
 json.set("patient2/pesan", "pasien 2 meminta bantuan");
 json.set("led1", false);
 json.set("led2", false);
 Firebase.setJSON(fbdo, "/iot", json);
}

void loop() {
 if (Firebase.ready()) {
   // Check patient 1 request from Firebase
   if (Firebase.getString(fbdo, "/iot/patient1/pesan")) {
     String message = fbdo.stringData();
     if (message == "pasien 1 meminta bantuan" && !patient1LedState) {
       digitalWrite(LED1_PIN, HIGH);  // Turn on LED 1
       patient1LedState = true;      // Update LED state
     } else if (message != "pasien 1 meminta bantuan" && patient1LedState) {
       digitalWrite(LED1_PIN, LOW);   // Turn off LED 1
       patient1LedState = false;     // Update LED state
     }
   }

   // Check patient 2 request from Firebase
   if (Firebase.getString(fbdo, "/iot/patient2/pesan")) {
     String message = fbdo.stringData();
     if (message == "pasien 2 meminta bantuan" && !patient2LedState) {
       digitalWrite(LED2_PIN, HIGH);  // Turn on LED 2
       patient2LedState = true;      // Update LED state
     } else if (message != "pasien 2 meminta bantuan" && patient2LedState) {
       digitalWrite(LED2_PIN, LOW);   // Turn off LED 2
       patient2LedState = false;     // Update LED state
     }
   }

   delay(500);  // Wait for 500 ms before checking again
 }
}
```

Aplikasi Flutter

Setelah perangkat keras selesai diprogram, aplikasi Flutter dibuat untuk memungkinkan pasien dan perawat berinteraksi dengan sistem. Pada aplikasi ini, terdapat dua role: **Perawat** dan **Pasien**. Perawat dapat melihat permintaan bantuan dari pasien, sementara pasien dapat mengirimkan permintaan bantuan.

MainScreen.dart Pada screen utama, pengguna dapat memilih apakah ingin menjadi perawat atau pasien.

```
import 'package:flutter/material.dart';
import 'nurse_screen.dart';
import 'patient_screen.dart';

class MainScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          'Choose Your Role',
          style: TextStyle(
            fontWeight: FontWeight.w700,
            color: Colors.white,
            fontSize: 22,
            fontFamily: 'Poppins',
          ),
        ),
        backgroundColor: Colors.blueAccent.shade700,
        elevation: 0,
        centerTitle: true,
      ),
      body: Container(
        decoration: const BoxDecoration(
          gradient: LinearGradient(
            colors: [Color(0xFFB3E5FC), Color(0xFFE1BEE7)],
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
          ),
        ),
        child: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              _buildRoleButton(
                context,
                title: 'Nurse Dashboard',
                icon: Icons.medical_services,
                onPressed: () {
                  Navigator.push(
                    context,
                    PageRouteBuilder(
                      pageBuilder: (context, animation, secondaryAnimation) =>
                          NurseScreen(),
                      transitionsBuilder:
                          (context, animation, secondaryAnimation, child) {
                        const begin = Offset(1.0, 0.0);
                        const end = Offset.zero;
                        const curve = Curves.easeInOut;
                        var tween = Tween(begin: begin, end: end)
                            .chain(CurveTween(curve: curve));
                        var offsetAnimation = animation.drive(tween);
                        return SlideTransition(
                            position: offsetAnimation, child: child);
                      },
                    ),
                  );
                },
              ),
              const SizedBox(height: 25),
              _buildRoleButton(
                context,
                title: 'Patient 1 Profile',
                icon: Icons.person,
                onPressed: () {
                  Navigator.push(
                    context,
                    PageRouteBuilder(
                      pageBuilder: (context, animation, secondaryAnimation) =>
                          PatientScreen(patientId: 1),
                      transitionsBuilder:
                          (context, animation, secondaryAnimation, child) {
                        const begin = Offset(1.0, 0.0);
                        const end = Offset.zero;
                        const curve = Curves.easeInOut;
                        var tween = Tween(begin: begin, end: end)
                            .chain(CurveTween(curve: curve));
                        var offsetAnimation = animation.drive(tween);
                        return SlideTransition(
                            position: offsetAnimation, child: child);
                      },
                    ),
                  );
                },
              ),
              const SizedBox(height: 25),
              _buildRoleButton(
                context,
                title: 'Patient 2 Profile',
                icon: Icons.person_outline,
                onPressed: () {
                  Navigator.push(
                    context,
                    PageRouteBuilder(
                      pageBuilder: (context, animation, secondaryAnimation) =>
                          PatientScreen(patientId: 2),
                      transitionsBuilder:
                          (context, animation, secondaryAnimation, child) {
                        const begin = Offset(1.0, 0.0);
                        const end = Offset.zero;
                        const curve = Curves.easeInOut;
                        var tween = Tween(begin: begin, end: end)
                            .chain(CurveTween(curve: curve));
                        var offsetAnimation = animation.drive(tween);
                        return SlideTransition(
                            position: offsetAnimation, child: child);
                      },
                    ),
                );
              },
            ),
          ],
        ),
      ),
    ),
  );
}

Widget _buildRoleButton(BuildContext context,
    {required String title,
    required IconData icon,
    required VoidCallback onPressed}) {
  return Card(
    elevation: 4,
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(20),
    ),
    child: InkWell(
      borderRadius: BorderRadius.circular(20),
      onTap: onPressed,
      child: Container(
        width: 320,
        padding: const EdgeInsets.symmetric(vertical: 18, horizontal: 20),
        decoration: BoxDecoration(
          color: Colors.white,
          borderRadius: BorderRadius.circular(20),
        ),
        child: Row(
          children: [
            Icon(
              icon,
              size: 32,
              color: Colors.blueAccent.shade700,
            ),
            const SizedBox(width: 15),
            Expanded(
              child: Text(
                title,
                style: const TextStyle(
                  fontSize: 18,
                  fontWeight: FontWeight.w600,
                  color: Colors.black87,
                  fontFamily: 'Poppins',
                ),
              ),
            ),
            ],
          ),
        ),
      ),
    );
  }
}
```

Nurse_screen.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_database/firebase_database.dart';

class NurseScreen extends StatefulWidget {
  @override
  _NurseScreenState createState() => _NurseScreenState();
}

class _NurseScreenState extends State<NurseScreen> {
  final DatabaseReference _db = FirebaseDatabase.instance.ref();

  // Method to acknowledge the patient's help request
  void acknowledgePatientRequest(int patientId) {
    _db.child('iot/patient$patientId/pesan').set('sudah ditangani');
    _db.child('iot/led$patientId').set(false);
  }

  // Method to refresh data (example action for FAB)
  void refreshData() {
    setState(() {}); // Trigger rebuild to refresh StreamBuilder
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          'Nurse Dashboard',
          style: TextStyle(
            fontWeight: FontWeight.w700,
            color: Colors.white,
            fontSize: 22,
            fontFamily: 'Poppins',
          ),
        ),
        backgroundColor: Colors.blueAccent.shade700,
        elevation: 0,
        centerTitle: true,
      ),
      body: Container(
        decoration: const BoxDecoration(
          gradient: LinearGradient(
            colors: [Color(0xFFB3E5FC), Color(0xFFE1BEE7)],
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
          ),
        ),
        child: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              _buildPatientCard(
                context,
                patientId: 1,
                stream: _db.child('iot/patient1/pesan').onValue,
                onAcknowledge: () => acknowledgePatientRequest(1),
              ),
              const SizedBox(height: 30),
              _buildPatientCard(
                context,
                patientId: 2,
                stream: _db.child('iot/patient2/pesan').onValue,
                onAcknowledge: () => acknowledgePatientRequest(2),
              ),
            ],
          ),
        ),
      ),
      floatingActionButton: FloatingActionButton(
        onPressed: refreshData,
        backgroundColor: Colors.blueAccent.shade700,
        child: const Icon(Icons.refresh, color: Colors.white),
        elevation: 4,
        tooltip: 'Refresh Data',
      ),
    );
  }

  Widget _buildPatientCard(
    BuildContext context, {
    required int patientId,
    required Stream<DatabaseEvent> stream,
    required VoidCallback onAcknowledge,
  }) {
    return Card(
      elevation: 4,
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(20),
      ),
      child: Container(
        width: 360,
        padding: const EdgeInsets.all(20),
        decoration: BoxDecoration(
          color: Colors.white,
          borderRadius: BorderRadius.circular(20),
        ),
        child: StreamBuilder<DatabaseEvent>(
          stream: stream,
          builder: (context, snapshot) {
            String message = 'Loading...';
            if (snapshot.connectionState == ConnectionState.active) {
              message =
                  (snapshot.data?.snapshot.value ?? 'No Request') as String;
            }
            return Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Row(
                  children: [
                    Icon(
                      Icons.person,
                      size: 32,
                      color: Colors.blueAccent.shade700,
                    ),
                    const SizedBox(width: 12),
                    Text(
                      'Patient $patientId',
                      style: const TextStyle(
                        fontSize: 20,
                        fontWeight: FontWeight.w700,
                        color: Colors.black87,
                        fontFamily: 'Poppins',
                      ),
                    ),
                  ],
                ),
                const SizedBox(height: 12),
                Text(
                  'Status: $message',
                  style: TextStyle(
                    fontSize: 16,
                    color: message == 'No Request'
                        ? Colors.grey.shade600
            : Colors.redAccent,
          fontFamily: 'Poppins',
         ),
        ),
        const SizedBox(height: 20),
        Align(
         alignment: Alignment.centerRight,
         child: AnimatedScaleButton(
          onPressed: onAcknowledge,
          child: ElevatedButton.icon(
           onPressed: onAcknowledge,
           icon: const Icon(Icons.check_circle,
             size: 20, color: Colors.white),
           label: const Text(
            'Acknowledge',
            style: TextStyle(
             color: Colors.white,
             fontFamily: 'Poppins',
             fontWeight: FontWeight.w600,
            ),
           ),
           style: ElevatedButton.styleFrom(
            backgroundColor: Colors.blueAccent.shade700,
            shape: RoundedRectangleBorder(
             borderRadius: BorderRadius.circular(15),
            ),
            padding: const EdgeInsets.symmetric(
              horizontal: 20, vertical: 12),
            elevation: 4,
           ),
          ),
         ),
        ),
       ],
      );
     },
    ),
   ),
  );
 }
}

// Custom widget for button with scale animation
class AnimatedScaleButton extends StatefulWidget {
 final VoidCallback onPressed;
 final Widget child;

 const AnimatedScaleButton({required this.onPressed, required this.child});
```

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

class _AnimatedScaleButtonState extends State<AnimatedScaleButton>
    with SingleTickerProviderStateMixin {
  double _scale = 1.0;

  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTapDown: (_) {
        setState(() {
          _scale = 0.95;
        });
      },
      onTapUp: (_) {
        setState(() {
          _scale = 1.0;
        });
        widget.onPressed();
      },
      onTapCancel: () {
        setState(() {
          _scale = 1.0;
        });
      },
      child: Transform.scale(
        scale: _scale,
        child: widget.child,
      ),
    );
  }
}
```

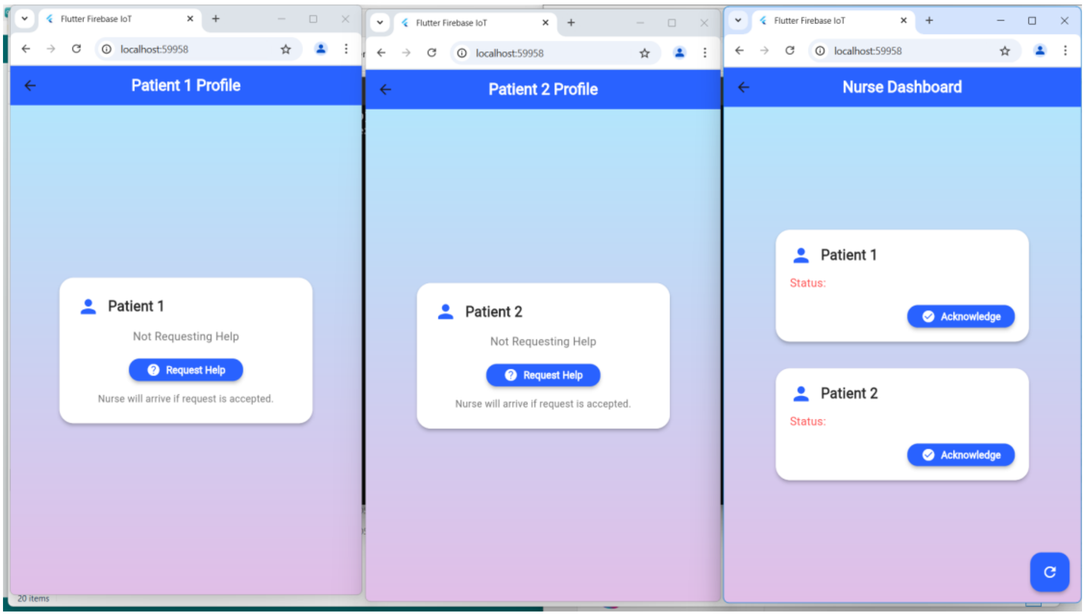
Patient_screen.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_database/firebase_database.dart';

class PatientScreen extends StatefulWidget {
  final int patientId;

  PatientScreen({required this.patientId});
  @override
  _PatientScreenState createState() => _PatientScreenState();
}

class _PatientScreenState extends State<PatientScreen> {
  final DatabaseReference _db = FirebaseDatabase.instance.ref();
  bool isRequestingHelp = false;
  bool isNurseResponding = false;

  // Method to toggle the help request
  void toggleHelpRequest() {
    setState(() {
      isRequestingHelp = !isRequestingHelp;
    });

    if (isRequestingHelp) {
      _db
          .child('iot/patient${widget.patientId}/pesan')
          .set('pasien ${widget.patientId} meminta bantuan');
      _db.child('iot/led${widget.patientId}').set(true);
    } else {
      _db.child('iot/patient${widget.patientId}/pesan').set('');
      _db.child('iot/led${widget.patientId}').set(false);
    }
  }

  // Listen to Firebase for nurse response
  @override
  void initState() {
    super.initState();
    _db.child('iot/patient${widget.patientId}/pesan').onValue.listen((event) {
      final message = event.snapshot.value as String? ?? '';
      if (message == 'sudah ditangani') {
        setState(() {
          isNurseResponding = true;
          isRequestingHelp = false;
        });
        Future.delayed(const Duration(seconds: 5), () {
          setState(() {
            isNurseResponding = false;
          });
          _db.child('iot/patient${widget.patientId}/pesan').set('');
        });
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Patient ${widget.patientId} Profile',
          style: const TextStyle(
            fontWeight: FontWeight.w700,
            color: Colors.white,
            fontSize: 22,
            fontFamily: 'Poppins',
          ),
        ),
        backgroundColor: Colors.blueAccent.shade700,
        elevation: 0,
        centerTitle: true,
      ),
      body: Container(
        decoration: const BoxDecoration(
          gradient: LinearGradient(
            colors: [Color(0xFFB3E5FC), Color(0xFFE1BEE7)],
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
          ),
        ),
        child: Center(
          child: Card(
            elevation: 4,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(20),
            ),
            child: Container(
              width: 360,
              padding: const EdgeInsets.all(25),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.circular(20),
              ),
              child: Column(
                mainAxisSize: MainAxisSize.min,
                children: [
                  Row(
                    children: [
                      Icon(
                        Icons.person,
                        size: 32,
                        color: Colors.blueAccent.shade700,
                      ),
                      const SizedBox(width: 12),
                      Text(
                        'Patient ${widget.patientId}',
                        style: const TextStyle(
                          fontSize: 20,
                          fontWeight: FontWeight.w700,
                          color: Colors.black87,
                          fontFamily: 'Poppins',
                        ),
                      ),
                    ],
                  ),
                  const SizedBox(height: 15),
                  Text(
                    isRequestingHelp
                        ? 'Requesting Help'
                        : 'Not Requesting Help',
                    style: TextStyle(
                      fontSize: 16,
                      color: isRequestingHelp
                          ? Colors.redAccent
                          : Colors.grey.shade600,
                      fontFamily: 'Poppins',
                    ),
                  ),
                  const SizedBox(height: 20),
                  AnimatedScaleButton(
                    onPressed: isNurseResponding ? () {} : toggleHelpRequest,
                    child: ElevatedButton.icon(
                      onPressed: isNurseResponding ? null : toggleHelpRequest,
                      icon: Icon(
                        isRequestingHelp ? Icons.cancel : Icons.help,
                        size: 20,
                        color: Colors.white,
                      ),
                      label: Text(
                        isRequestingHelp ? 'Cancel Request' : 'Request Help',
                        style: const TextStyle(
                          color: Colors.white,
                          fontFamily: 'Poppins',
                          fontWeight: FontWeight.w600,
                        ),
                      ),
                      style: ElevatedButton.styleFrom(
                        backgroundColor: isRequestingHelp
                            ? Colors.redAccent
                            : Colors.blueAccent.shade700,
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(15),
            ),
            padding: const EdgeInsets.symmetric(
              horizontal: 25, vertical: 14),
            elevation: 4,
           ),
          ),
         ),
         const SizedBox(height: 15),
         if (!isRequestingHelp && !isNurseResponding)
          Text(
           'Nurse will arrive if request is accepted.',
           style: TextStyle(
            fontSize: 14,
            color: Colors.grey.shade600,
            fontFamily: 'Poppins',
           ),
           textAlign: TextAlign.center,
          ),
         if (isNurseResponding)
          Text(
           'Nurse is on the way!',
           style: const TextStyle(
            fontSize: 16,
            color: Colors.green,
            fontWeight: FontWeight.w600,
            fontFamily: 'Poppins',
           ),
          ),
        ],
       ),
      ),
     ),
    ),
   ),
  );
 }
}

// Custom widget for button with scale animation
class AnimatedScaleButton extends StatefulWidget {
 final VoidCallback onPressed;
 final Widget child;

 const AnimatedScaleButton({required this.onPressed, required this.child});

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

```
class _AnimatedScaleButtonState extends State<AnimatedScaleButton>
  with SingleTickerProviderStateMixin {
 double _scale = 1.0;

 @override
 Widget build(BuildContext context) {
  return GestureDetector(
   onTapDown: (_) {
    setState(() {
     _scale = 0.95;
    });
   },
   onTapUp: (_) {
    setState(() {
     _scale = 1.0;
    });
    widget.onPressed();
   },
   onTapCancel: () {
    setState(() {
     _scale = 1.0;
    });
   },
   child: Transform.scale(
    scale: _scale,
    child: widget.child,
   ),
  );
 }
}
```

4. Hasil

- Aplikasi ini memungkinkan pasien untuk mengirimkan permintaan bantuan yang ditandai dengan LED menyala.
- Perawat dapat melihat permintaan bantuan tersebut dan menanggapinya melalui aplikasi dengan menekan tombol "Acknowledge", yang akan mengubah status pasien menjadi "Sudah Ditangani".
- Sistem ini dihubungkan menggunakan Firebase untuk memastikan komunikasi real-time antara pasien dan perawat.
- **Link Video:** https://youtube.com/shorts/HM-bPAxvxTQ?si=zljey4_LuXKZh3h8



Flutter Firebase IoT

localhost:59958

Patient 1 Profile

Patient 1

Not Requesting Help

Request Help

Nurse will arrive if request is accepted.

20 items

Flutter Firebase IoT

localhost:59958

Patient 2 Profile

Patient 2

Not Requesting Help

Request Help

Nurse will arrive if request is accepted.

Flutter Firebase IoT

localhost:59958

Nurse Dashboard

Patient 1

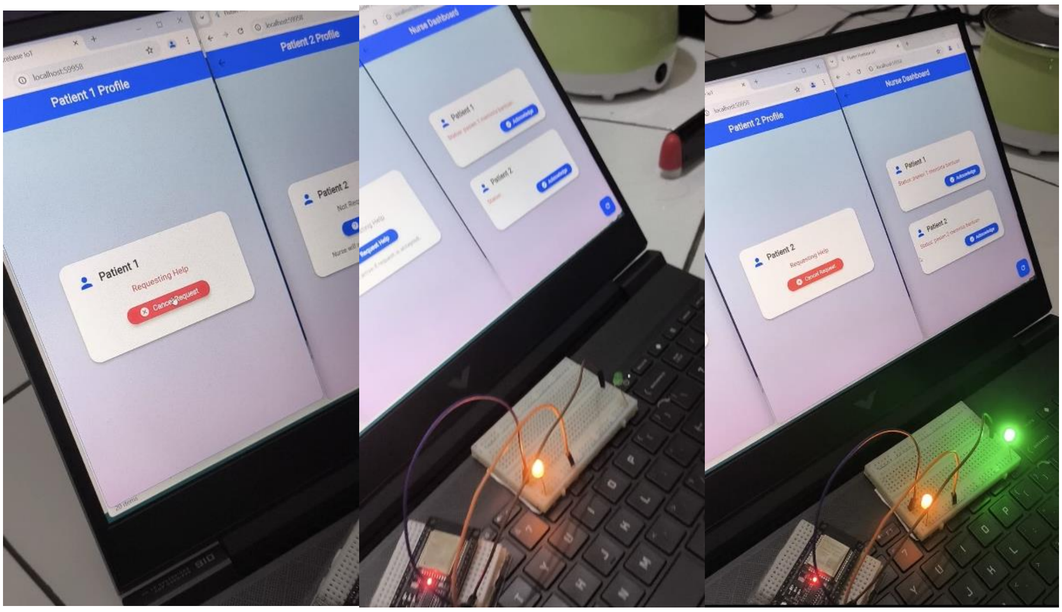
Status:

Acknowledge

Patient 2

Status:

Acknowledge



5. Kesimpulan Implementasi IoT di rumah sakit dapat meningkatkan responsivitas dalam menangani pasien yang membutuhkan bantuan. Melalui penggunaan ESP32 dan Firebase, sistem ini memungkinkan pasien untuk mengirimkan permintaan bantuan secara real-time yang dapat langsung ditindaklanjuti oleh perawat. Dengan aplikasi Flutter yang terintegrasi, komunikasi antara pasien dan perawat menjadi lebih efisien.