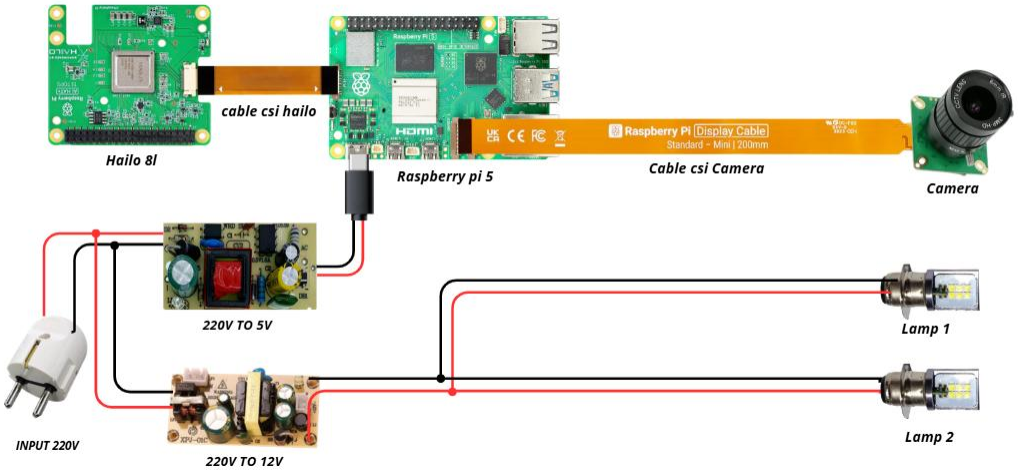
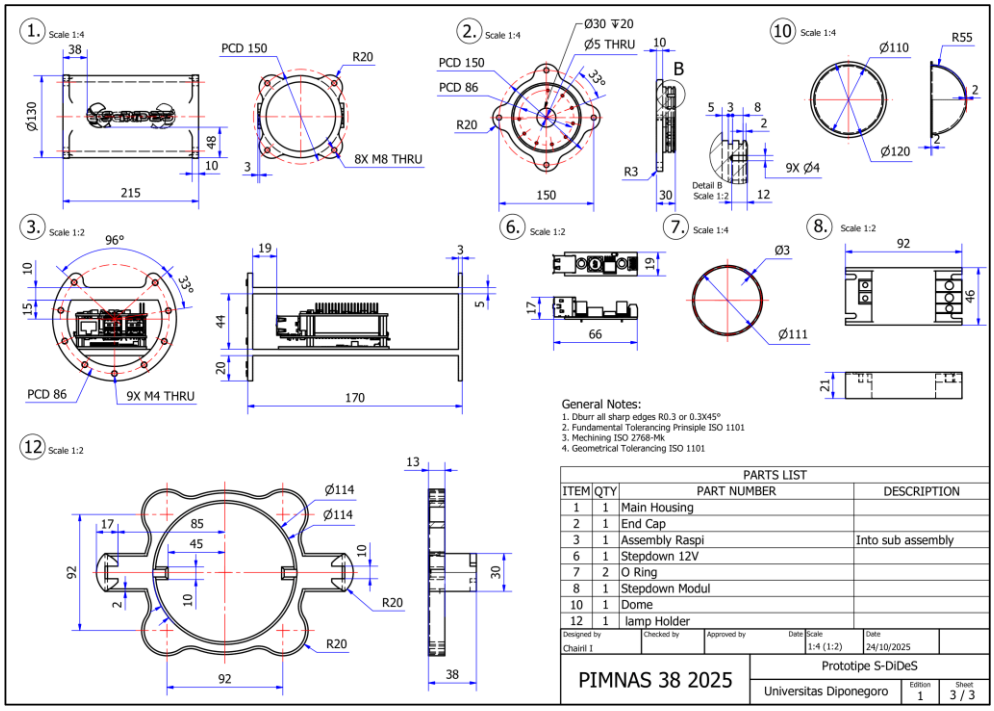
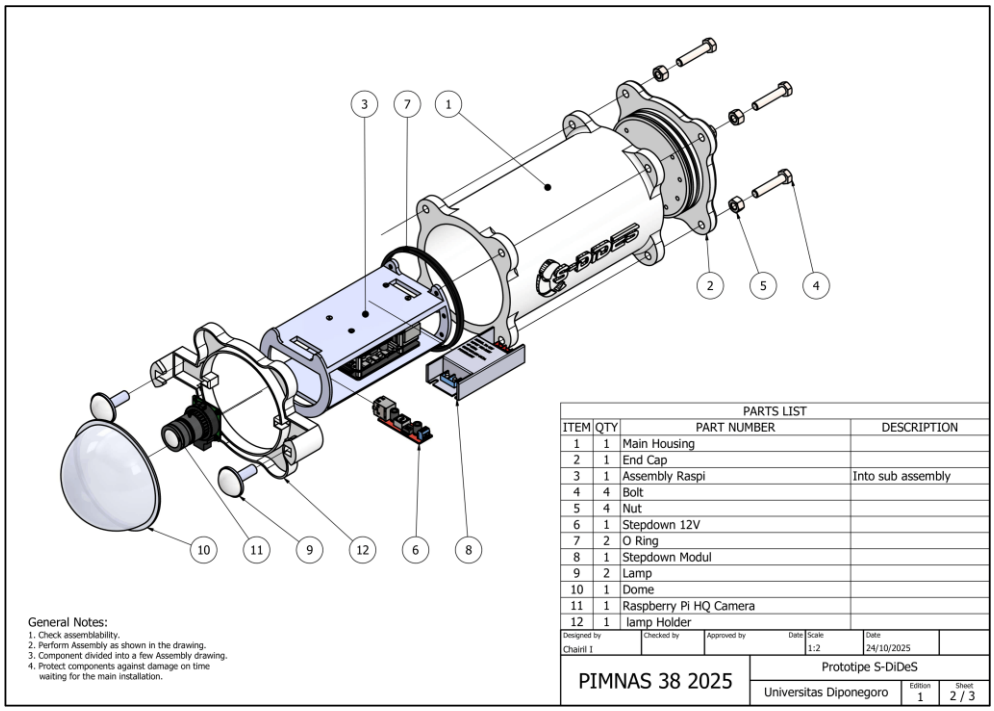
LAMPIRAN

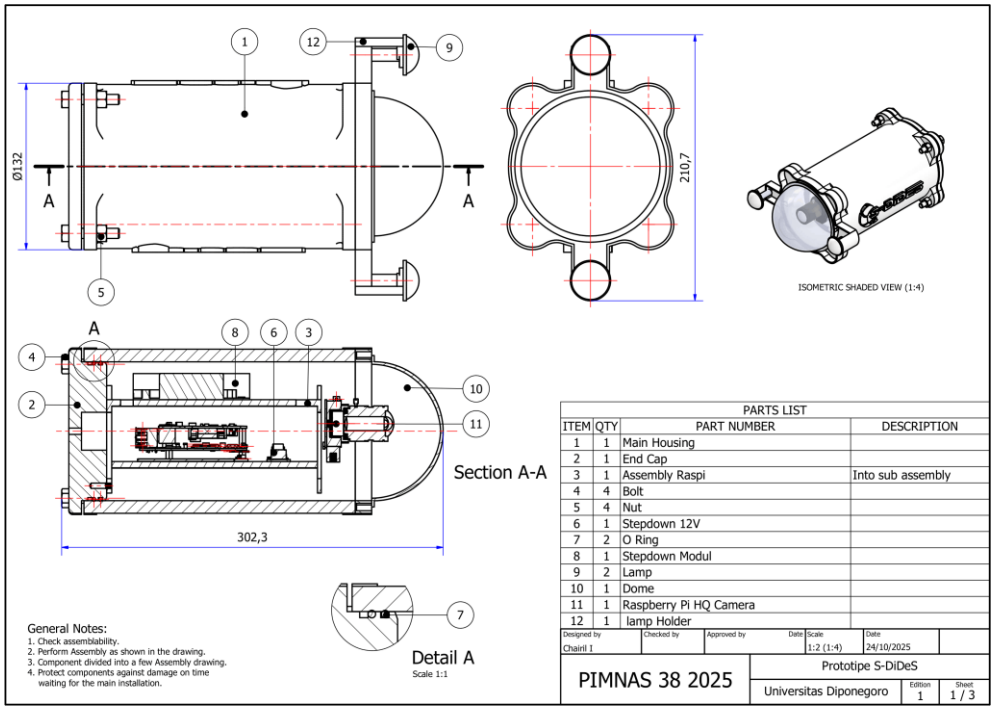
Lampiran 1 Skema Rangkaian Alat



Lampiran 2 Rancangan Prototipe







Lampiran 3 Program Disesase Detail Aplikasi S-dides

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

class DiseaseDetailScreen extends StatelessWidget {
```

```
  final String diseaseName;
  final String diseaseCode;
  final String status;
  final Color statusColor;
  final IconData icon;
  final Color iconColor;
  final Color iconBgColor;

  const DiseaseDetailScreen({
    super.key,
    required this.diseaseName,
    required this.diseaseCode,
    required this.status,
    required this.statusColor,
    required this.icon,
    required this.iconColor,
    required this.iconBgColor,
  });

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      appBar: AppBar(
        backgroundColor: const Color(0xFF00BFA5),
        elevation: 0,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white),
          onPressed: () => Navigator.pop(context),
        ),
        title: const Text(
          'Detail Penyakit',
          style: TextStyle(color: Colors.white, fontWeight:
FontWeight.bold),
        ),
      ),
      body: SingleChildScrollView(
        child: Column(
          children: [
            // Header Card
            Container(
              width: double.infinity,
              padding: const EdgeInsets.all(24),
              decoration: const BoxDecoration(
                gradient: LinearGradient(
```

```
            colors: [Color(0xFF00BFA5), Color(0xFF00897B)],
            begin: Alignment.topLeft,
            end: Alignment.bottomRight,
          ),
        ),
        child: Column(
          children: [
            Container(
              padding: const EdgeInsets.all(20),
              decoration: BoxDecoration(
                color: iconBgColor,
                borderRadius: BorderRadius.circular(20),
                boxShadow: [
                  BoxShadow(
                    color: Colors.black.withAlpha(25),
                    blurRadius: 15,
                    offset: const Offset(0, 5),
                  ),
                ],
              ),
              child: Icon(
                icon,
                color: iconColor,
                size: 48,
              ),
            ),
            const SizedBox(height: 16),
            Text(
              diseaseName,
              textAlign: TextAlign.center,
              style: const TextStyle(
                fontSize: 20,
                fontWeight: FontWeight.bold,
                color: Colors.white,
              ),
            ),
            const SizedBox(height: 8),
            Text(
              diseaseCode,
              style: TextStyle(
                fontSize: 14,
                color: Colors.white.withAlpha(200),
              ),
            ),
            const SizedBox(height: 12),
```

```
                Container(
                  padding: const EdgeInsets.symmetric(horizontal:
16, vertical: 8),
                  decoration: BoxDecoration(
                    color: statusColor,
                    borderRadius: BorderRadius.circular(20),
                  ),
                  child: Text(
                    'Status: $status',
                    style: const TextStyle(
                      color: Colors.white,
                      fontSize: 14,
                      fontWeight: FontWeight.bold,
                    ),
                  ),
                ),
              ],
            ),
          ),

          // Information Sections
          Padding(
            padding: const EdgeInsets.all(16),
            child: Column(
              children: [
                // Deskripsi
                _buildInfoCard(
                  title: 'Deskripsi',
                  icon: Icons.description,
                  content: _getDescription(diseaseCode),
                ),
                const SizedBox(height: 12),

                // Gejala
                _buildInfoCard(
                  title: 'Gejala Klinis',
                  icon: Icons.medical_information,
                  content: _getSymptoms(diseaseCode),
                ),
                const SizedBox(height: 12),

                // Penyebab
                _buildInfoCard(
                  title: 'Penyebab',
                  icon: Icons.coronavirus,
```

```
                  content: _getCauses(diseaseCode),
                ),
                const SizedBox(height: 12),

                // Pencegahan
                _buildInfoCard(
                  title: 'Pencegahan',
                  icon: Icons.shield,
                  content: _getPrevention(diseaseCode),
                ),
                const SizedBox(height: 12),

                // Penanganan
                _buildInfoCard(
                  title: 'Penanganan',
                  icon: Icons.healing,
                  content: _getTreatment(diseaseCode),
                ),
                const SizedBox(height: 12),

                // Statistics Card
                Container(
                  padding: const EdgeInsets.all(20),
                  decoration: BoxDecoration(
                    color: Colors.white,
                    borderRadius: BorderRadius.circular(16),
                    boxShadow: [
                      BoxShadow(
                        color: Colors.black.withAlpha(13),
                        blurRadius: 10,
                        offset: const Offset(0, 2),
                      ),
                    ],
                  ),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      Row(
                        children: [
                          Container(
                            padding: const EdgeInsets.all(8),
                            decoration: BoxDecoration(
                              color: const
Color(0xFF00BFA5).withAlpha(25),
```

```
                                borderRadius:
BorderRadius.circular(8),
                              ),
                              child: const Icon(
                                Icons.bar_chart,
                                color: Color(0xFF00BFA5),
                                size: 20,
                              ),
                            ),
                            const SizedBox(width: 12),
                            const Text(
                              'Statistik',
                              style: TextStyle(
                                fontSize: 16,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 16),
                        _buildStatRow('Tingkat Keparahan',
_getSeverityLevel(diseaseCode)),
                        const Divider(height: 24),
                        _buildStatRow('Tingkat Penyebaran',
_getSpreadRate(diseaseCode)),
                        const Divider(height: 24),
                        _buildStatRow('Tingkat Mortalitas',
_getMortalityRate(diseaseCode)),
                      ],
                    ),
                  ),
                ],
              ),
            ),
          ],
        ),
      ),
    );
  }

  Widget _buildInfoCard({
    required String title,
    required IconData icon,
    required String content,
  }) {
```

```
    return Container(
      width: double.infinity,
      padding: const EdgeInsets.all(20),
      decoration: BoxDecoration(
        color: Colors.white,
        borderRadius: BorderRadius.circular(16),
        boxShadow: [
          BoxShadow(
            color: Colors.black.withAlpha(13),
            blurRadius: 10,
            offset: const Offset(0, 2),
          ),
        ],
      ),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Row(
            children: [
              Container(
                padding: const EdgeInsets.all(8),
                decoration: BoxDecoration(
                  color: const Color(0xFF00BFA5).withAlpha(25),
                  borderRadius: BorderRadius.circular(8),
                ),
                child: Icon(
                  icon,
                  color: const Color(0xFF00BFA5),
                  size: 20,
                ),
              ),
              const SizedBox(width: 12),
              Text(
                title,
                style: const TextStyle(
                  fontSize: 16,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
          const SizedBox(height: 12),
          Text(
            content,
            style: TextStyle(
```

```
              fontSize: 14,
              color: Colors.grey[700],
              height: 1.6,
            ),
          ),
        ],
      ),
    );
  }

  Widget _buildStatRow(String label, String value) {
    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Text(
          label,
          style: TextStyle(
            fontSize: 14,
            color: Colors.grey[600],
          ),
        ),
        Text(
          value,
          style: const TextStyle(
            fontSize: 14,
            fontWeight: FontWeight.bold,
            color: Color(0xFF00BFA5),
          ),
        ),
      ],
    );
  }

  // Data methods
  String _getDescription(String code) {
    switch (code) {
      case '(WSSV)':
        return 'White Spot Syndrome Virus (WSSV) adalah virus DNA
yang sangat menular dan mematikan yang menyerang udang. Virus ini
dapat menyebabkan kematian massal hingga 100% dalam waktu 3-10 hari
setelah gejala pertama muncul.';
      case '(TSV)':
        return 'Taura Syndrome Virus (TSV) adalah penyakit virus
yang menyerang udang, terutama udang putih Pasifik. Penyakit ini
```

```
pertama kali ditemukan di dekat Sungai Taura, Ekuador pada tahun
1992.';
      case '(WFD)':
        return 'White Feces Disease (WFD) adalah penyakit yang
ditandai dengan kotoran berwarna putih mengambang di permukaan air.
Penyakit ini dapat menyebabkan pertumbuhan lambat dan kematian pada
udang.';
      case '(IMNV)':
        return 'Infectious Myonecrosis Virus (IMNV) adalah virus
yang menyebabkan nekrosis pada otot udang. Penyakit ini pertama kali
ditemukan di Brasil dan dapat menyebabkan kerugian ekonomi yang
signifikan.';
      default:
        return 'Informasi tidak tersedia.';
    }
  }

  String _getSymptoms(String code) {
    switch (code) {
      case '(WSSV)':
        return '• Bintik putih pada karapas dan kulit\n• Udang
berenang ke permukaan air\n• Nafsu makan menurun drastis\n• Warna
tubuh menjadi kemerahan\n• Karapas lunak dan mudah dilepas\n•
Kematian mendadak dalam jumlah besar';
      case '(TSV)':
        return '• Kulit lunak (soft shell)\n• Ekor berwarna merah\n•
Perubahan warna pada hepatopankreas\n• Udang letargi dan lemah\n•
Melanisasi atau bintik hitam pada kulit\n• Penurunan nafsu makan';
      case '(WFD)':
        return '• Kotoran berwarna putih mengambang\n•
Hepatopankreas pucat dan mengecil\n• Pertumbuhan lambat\n• Usus
kosong dan berwarna putih\n• Nafsu makan berkurang\n• Tingkat
kelangsungan hidup rendah';
      case '(IMNV)':
        return '• Nekrosis fokal pada otot\n• Otot berwarna putih
keruh\n• Udang lemah dan letargi\n• Kesulitan berenang\n• Nekrosis
pada ekor\n• Penurunan bobot tubuh';
      default:
        return 'Informasi gejala tidak tersedia.';
    }
  }

  String _getCauses(String code) {
    switch (code) {
      case '(WSSV)':
```

```
        return 'Disebabkan oleh White Spot Syndrome Virus (genus
Whispovirus). Penularan terjadi melalui air, kontak langsung,
kanibalisme, dan vektor seperti zooplankton. Faktor pemicu: kualitas
air buruk, stres, kepadatan tinggi.';
      case '(TSV)':
        return 'Disebabkan oleh Taura Syndrome Virus (genus
Aparavirus). Penularan melalui air dan kontak horizontal antar
udang. Faktor pemicu: stres lingkungan, perubahan salinitas
mendadak, kualitas benur.';
      case '(WFD)':
        return 'Penyebab pasti belum diketahui secara pasti, diduga
kombinasi infeksi mikroba (Vibrio, Microsporidium) dan kondisi
lingkungan. Faktor pemicu: kualitas pakan, kualitas air, manajemen
kolam.';
      case '(IMNV)':
        return 'Disebabkan oleh Infectious Myonecrosis Virus (genus
Totiviridae). Penularan vertikal (induk ke benur) dan horizontal.
Faktor risiko: suhu air rendah, kualitas benur, manajemen tambak.';
      default:
        return 'Informasi penyebab tidak tersedia.';
    }
  }

  String _getPrevention(String code) {
    switch (code) {
      case '(WSSV)':
        return '• Gunakan benur SPF (Specific Pathogen Free)\n•
Terapkan biosekuriti ketat\n• Jaga kualitas air optimal\n• Hindari
kepadatan tinggi\n• Disinfeksi peralatan secara rutin\n• Screening
benur sebelum tebar';
      case '(TSV)':
        return '• Gunakan benur SPR (Specific Pathogen Resistant)\n•
Aklimatisasi benur dengan baik\n• Jaga stabilitas salinitas\n•
Monitoring kesehatan rutin\n• Karantina udang baru\n• Manajemen
stres yang baik';
      case '(WFD)':
        return '• Jaga kualitas pakan (probiotik)\n• Manajemen
kualitas air optimal\n• Hindari overfeeding\n• Gunakan aerasi yang
cukup\n• Monitoring plankton\n• Sanitasi kolam yang baik';
      case '(IMNV)':
        return '• Seleksi induk bebas IMNV\n• Kontrol suhu air
(hindari < 18°C)\n• Terapkan biosekuriti\n• Monitoring PCR secara
berkala\n• Manajemen nutrisi baik\n• Hindari stres lingkungan';
      default:
        return 'Informasi pencegahan tidak tersedia.';
```

```
    }
  }

  String _getTreatment(String code) {
    switch (code) {
      case '(WSSV)':
        return '• Tidak ada pengobatan spesifik untuk virus\n• Panen
awal (emergency harvest)\n• Isolasi kolam terinfeksi\n• Disinfeksi
total setelah panen\n• Pengeringan kolam minimal 2 minggu\n• Buang
udang mati segera';
      case '(TSV)':
        return '• Tidak ada pengobatan antivirus\n• Kurangi feeding
rate\n• Tingkatkan aerasi\n• Jaga kualitas air optimal\n• Pemberian
imunostimulan\n• Partial harvest jika perlu';
      case '(WFD)':
        return '• Aplikasi probiotik melalui pakan dan air\n•
Perbaiki kualitas air\n• Kurangi jumlah pakan bertahap\n• Tingkatkan
aerasi\n• Siphon untuk buang kotoran\n• Pemberian suplemen vitamin
C';
      case '(IMNV)':
        return '• Tidak ada pengobatan spesifik\n• Jaga suhu air
stabil > 20°C\n• Perbaiki nutrisi pakan\n• Kurangi kepadatan\n•
Tingkatkan kualitas air\n• Panen selektif udang sakit';
      default:
        return 'Informasi penanganan tidak tersedia.';
    }
  }

  String _getSeverityLevel(String code) {
    switch (code) {
      case '(WSSV)':
        return 'Sangat Tinggi';
      case '(TSV)':
        return 'Tinggi';
      case '(WFD)':
        return 'Sedang-Tinggi';
      case '(IMNV)':
        return 'Sangat Tinggi';
      default:
        return 'N/A';
    }
  }

  String _getSpreadRate(String code) {
    switch (code) {
```

```
      case '(WSSV)':
        return 'Sangat Cepat';
      case '(TSV)':
        return 'Cepat';
      case '(WFD)':
        return 'Sedang';
      case '(IMNV)':
        return 'Sedang-Cepat';
      default:
        return 'N/A';
    }
  }

  String _getMortalityRate(String code) {
    switch (code) {
      case '(WSSV)':
        return '80-100%';
      case '(TSV)':
        return '40-90%';
      case '(WFD)':
        return '20-60%';
      case '(IMNV)':
        return '40-70%';
      default:
        return 'N/A';
    }
  }
}
```

Lampiran 4 Program Monitor Aplikasi S-dides

```
import 'package:flutter/material.dart';
import 'package:camera/camera.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';
import 'dart:io';
import 'settings_screen.dart';

class MonitorScreen extends StatefulWidget {
  const MonitorScreen({super.key});

  @override
  State<MonitorScreen> createState() => _MonitorScreenState();
}
```

```
class _MonitorScreenState extends State<MonitorScreen> {
  CameraController? _cameraController;
  List<CameraDescription>? _cameras;
  bool _isCameraInitialized = false;
  bool _isDetecting = false;
  String _serverUrl = '';

  @override
  void initState() {
    super.initState();
    _loadServerUrl();
    _initializeCamera();
  }

  Future<void> _loadServerUrl() async {
    final prefs = await SharedPreferences.getInstance();
    final url = prefs.getString('server_url') ??
'http://192.168.1.100';
    final port = prefs.getString('server_port') ?? '5000';
    setState(() {
      _serverUrl = '$url:$port';
    });
  }

  Future<void> _initializeCamera() async {
    try {
      _cameras = await availableCameras();
      if (_cameras != null && _cameras!.isNotEmpty) {
        _cameraController = CameraController(
          _cameras![0],
          ResolutionPreset.high,
          enableAudio: false,
        );

        await _cameraController!.initialize();
        if (mounted) {
          setState(() {
            _isCameraInitialized = true;
          });
        }
      }
    } catch (e) {
      debugPrint('Error initializing camera: $e');
    }
```

```
  }

  Future<void> _startDetection() async {
    if (_cameraController == null ||
!_cameraController!.value.isInitialized) {
      _showSnackBar('Kamera belum siap', Colors.red);
      return;
    }

    if (_serverUrl.isEmpty) {
      _showSnackBar('URL server belum dikonfigurasi',
Colors.orange);
      return;
    }

    setState(() => _isDetecting = true);

    try {
      // Capture image from camera
      final XFile imageFile = await
_cameraController!.takePicture();

      // Send to server for detection
      await _sendImageToServer(imageFile);

    } catch (e) {
      _showSnackBar('Gagal mengambil gambar: $e', Colors.red);
    } finally {
      setState(() => _isDetecting = false);
    }
  }

  Future<void> _sendImageToServer(XFile imageFile) async {
    try {
      final uri = Uri.parse('$_serverUrl/detect');
      final request = http.MultipartRequest('POST', uri);

      // Add image file to request
      final file = await http.MultipartFile.fromPath(
        'image',
        imageFile.path,
      );
      request.files.add(file);

      _showSnackBar('Mengirim ke server...', Colors.blue);
```

```
      final streamedResponse = await request.send().timeout(
        const Duration(seconds: 30),
      );

      final response = await
http.Response.fromStream(streamedResponse);

      if (response.statusCode == 200) {
        final result = json.decode(response.body);
        _showDetectionResult(result);
      } else {
        _showSnackBar('Server error: ${response.statusCode}',
Colors.red);
      }

    } catch (e) {
      _showSnackBar('Gagal mengirim ke server: $e', Colors.red);
    }
  }

  void _showDetectionResult(Map<String, dynamic> result) {
    // Parse detection result
    final detected = result['detected'] ?? false;
    final disease = result['disease'] ?? 'Unknown';
    final confidence = result['confidence'] ?? 0.0;

    showDialog(
      context: context,
      builder: (context) => AlertDialog(
        title: Row(
          children: [
            Icon(
              detected ? Icons.warning_amber : Icons.check_circle,
              color: detected ? Colors.red : Colors.green,
            ),
            const SizedBox(width: 8),
            const Text('Hasil Deteksi'),
          ],
        ),
        content: Column(
          mainAxisSize: MainAxisSize.min,
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
```

```
            Text('Status: ${detected ? "Terdeteksi Penyakit" :
"Sehat"}'),
            if (detected) ...[
              const SizedBox(height: 8),
              Text('Penyakit: $disease'),
              Text('Confidence: ${(confidence *
100).toStringAsFixed(1)}%'),
            ],
          ],
        ),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('OK'),
          ),
        ],
      ),
    );
  }

  void _showSnackBar(String message, Color color) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: Text(message),
        backgroundColor: color,
        behavior: SnackBarBehavior.floating,
        duration: const Duration(seconds: 2),
      ),
    );
  }

  @override
  void dispose() {
    _cameraController?.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      body: SingleChildScrollView(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.start,
          children: [
```

```
            // Live Camera Section
            Container(
              margin: const EdgeInsets.all(16),
              padding: const EdgeInsets.all(20),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.circular(16),
                boxShadow: [
                  BoxShadow(
                    color: Colors.black.withAlpha(13),
                    blurRadius: 10,
                    offset: const Offset(0, 2),
                  ),
                ],
              ),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Row(
                    children: [
                      Container(
                        padding: const EdgeInsets.all(8),
                        decoration: BoxDecoration(
                          color: const Color(0xFF00BFA5),
                          borderRadius: BorderRadius.circular(8),
                        ),
                        child: const Icon(
                          Icons.camera_alt,
                          color: Colors.white,
                          size: 20,
                        ),
                      ),
                      const SizedBox(width: 12),
                      const Expanded(
                        child: Column(
                          crossAxisAlignment:
CrossAxisAlignment.start,
                          children: [
                            Text(
                              'Live Camera',
                              style: TextStyle(
                                fontSize: 18,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
```

```
                          Text(
                            'AI-Powered Detection',
                            style: TextStyle(
                              fontSize: 12,
                              color: Colors.grey,
                            ),
                          ),
                        ],
                      ),
                    ),
                    // Settings Button
                    IconButton(
                      onPressed: () async {
                        final result = await Navigator.push(
                          context,
                          MaterialPageRoute(
                            builder: (context) => const
SettingsScreen(),
                          ),
                        );
                        if (result != null) {
                          await _loadServerUrl();
                        }
                      },
                      icon: const Icon(Icons.settings),
                      color: const Color(0xFF00BFA5),
                      tooltip: 'Pengaturan Server',
                    ),
                  ],
                ),
                const SizedBox(height: 16),
                // Camera Preview
                Container(
                  height: 200,
                  decoration: BoxDecoration(
                    color: const Color(0xFF1A1F2E),
                    borderRadius: BorderRadius.circular(12),
                  ),
                  child: ClipRRect(
                    borderRadius: BorderRadius.circular(12),
                    child: _isCameraInitialized &&
_cameraController != null
                        ? Stack(
                            children: [
                              SizedBox.expand(
```

```
                                  child: FittedBox(
                                    fit: BoxFit.cover,
                                    child: SizedBox(
                                      width:
_cameraController!.value.previewSize?.height ?? 1,
                                      height:
_cameraController!.value.previewSize?.width ?? 1,
                                      child:
CameraPreview(_cameraController!),
                                    ),
                                  ),
                                ),
                                Positioned(
                                  top: 12,
                                  left: 12,
                                  child: Container(
                                    padding: const
EdgeInsets.symmetric(
                                      horizontal: 8,
                                      vertical: 4,
                                    ),
                                    decoration: BoxDecoration(
                                      color: Colors.red,
                                      borderRadius:
BorderRadius.circular(4),
                                    ),
                                    child: const Row(
                                      children: [
                                        Icon(
                                          Icons.circle,
                                          color: Colors.white,
                                          size: 8,
                                        ),
                                        SizedBox(width: 4),
                                        Text(
                                          'LIVE',
                                          style: TextStyle(
                                            color: Colors.white,
                                            fontSize: 10,
                                            fontWeight:
FontWeight.bold,
                                          ),
                                        ),
                                      ],
                                    ),
```

```
                                ),
                              ),
                              Positioned(
                                top: 12,
                                right: 12,
                                child: Container(
                                  padding: const
EdgeInsets.symmetric(
                                    horizontal: 8,
                                    vertical: 4,
                                  ),
                                  decoration: BoxDecoration(
                                    color:
Colors.black.withAlpha(127),
                                    borderRadius:
BorderRadius.circular(4),
                                  ),
                                  child: const Text(
                                    '4K',
                                    style: TextStyle(
                                      color: Colors.white,
                                      fontSize: 10,
                                      fontWeight: FontWeight.bold,
                                    ),
                                  ),
                                ),
                              ),
                            ],
                          )
                        : Center(
                            child: Column(
                              mainAxisAlignment:
MainAxisAlignment.center,
                              children: [
                                const CircularProgressIndicator(
                                  color: Color(0xFF00BFA5),
                                ),
                                const SizedBox(height: 12),
                                Text(
                                  'Memuat kamera...',
                                  style: TextStyle(
                                    color:
Colors.white.withAlpha(178),
                                    fontSize: 14,
                                  ),
```

```
                                ),
                              ],
                            ),
                          ),
                    ),
                  ),
                  const SizedBox(height: 16),
                  // Start Detection Button
                  SizedBox(
                    width: double.infinity,
                    child: ElevatedButton(
                      onPressed: _isDetecting ? null :
_startDetection,
                      style: ElevatedButton.styleFrom(
                        backgroundColor: const Color(0xFF00BFA5),
                        padding: const
EdgeInsets.symmetric(vertical: 16),
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(12),
                        ),
                        disabledBackgroundColor: Colors.grey,
                      ),
                      child: _isDetecting
                          ? const SizedBox(
                              height: 20,
                              width: 20,
                              child: CircularProgressIndicator(
                                color: Colors.white,
                                strokeWidth: 2,
                              ),
                            )
                          : const Row(
                              mainAxisAlignment:
MainAxisAlignment.center,
                              children: [
                                Icon(Icons.play_arrow, color:
Colors.white),
                                SizedBox(width: 8),
                                Text(
                                  'Start Detection',
                                  style: TextStyle(
                                    fontSize: 16,
                                    fontWeight: FontWeight.bold,
                                    color: Colors.white,
                                  ),
```

```
                              ),
                            ],
                          ),
                      ),
                    ),
                  ],
                ),
              ),

              // Stats Cards
              Padding(
                padding: const EdgeInsets.symmetric(horizontal: 16),
                child: Row(
                  children: [
                    Expanded(
                      child: Container(
                        padding: const EdgeInsets.all(16),
                        decoration: BoxDecoration(
                          color: const Color(0xFFE8F5E9),
                          borderRadius: BorderRadius.circular(12),
                        ),
                        child: Column(
                          crossAxisAlignment:
CrossAxisAlignment.start,
                          children: [
                            Row(
                              children: [
                                Container(
                                  padding: const EdgeInsets.all(6),
                                  decoration: BoxDecoration(
                                    color: const Color(0xFF00BFA5),
                                    borderRadius:
BorderRadius.circular(6),
                                  ),
                                  child: const Icon(
                                    Icons.check_circle,
                                    color: Colors.white,
                                    size: 16,
                                  ),
                                ),
                                const SizedBox(width: 8),
                                const Text(
                                  'Sehat Hari Ini',
                                  style: TextStyle(
                                    fontSize: 12,
```

```
                                color: Color(0xFF00BFA5),
                                fontWeight: FontWeight.w500,
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 8),
                        const Text(
                          '0',
                          style: TextStyle(
                            fontSize: 32,
                            fontWeight: FontWeight.bold,
                            color: Color(0xFF00BFA5),
                          ),
                        ),
                        const Text(
                          '↑ 0% dari kemarin',
                          style: TextStyle(
                            fontSize: 10,
                            color: Color(0xFF00BFA5),
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
                const SizedBox(width: 12),
                Expanded(
                  child: Container(
                    padding: const EdgeInsets.all(16),
                    decoration: BoxDecoration(
                      color: const Color(0xFFFFEBEE),
                      borderRadius: BorderRadius.circular(12),
                    ),
                    child: Column(
                      crossAxisAlignment:
CrossAxisAlignment.start,
                      children: [
                        Row(
                          children: [
                            Container(
                              padding: const EdgeInsets.all(6),
                              decoration: BoxDecoration(
                                color: Colors.red,
```

```
                                borderRadius:
BorderRadius.circular(6),
                              ),
                              child: const Icon(
                                Icons.cancel,
                                color: Colors.white,
                                size: 16,
                              ),
                            ),
                            const SizedBox(width: 8),
                            const Flexible(
                              child: Text(
                                'Terdeteksi Sakit',
                                style: TextStyle(
                                  fontSize: 12,
                                  color: Colors.red,
                                  fontWeight: FontWeight.w500,
                                ),
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 8),
                        const Text(
                          '0',
                          style: TextStyle(
                            fontSize: 32,
                            fontWeight: FontWeight.bold,
                            color: Colors.red,
                          ),
                        ),
                        const Text(
                          'Butuh perhatian',
                          style: TextStyle(
                            fontSize: 10,
                            color: Colors.red,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
              ],
            ),
          ),
        ),
```

```
          const SizedBox(height: 16),

          // Recent Detection Section
          Container(
            margin: const EdgeInsets.all(16),
            padding: const EdgeInsets.all(20),
            decoration: BoxDecoration(
              color: Colors.white,
              borderRadius: BorderRadius.circular(16),
              boxShadow: [
                BoxShadow(
                  color: Colors.black.withAlpha(13),
                  blurRadius: 10,
                  offset: const Offset(0, 2),
                ),
              ],
            ),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Row(
                  children: [
                    Container(
                      padding: const EdgeInsets.all(8),
                      decoration: BoxDecoration(
                        color: const Color(0xFF00BFA5),
                        borderRadius: BorderRadius.circular(8),
                      ),
                      child: const Icon(
                        Icons.auto_graph,
                        color: Colors.white,
                        size: 20,
                      ),
                    ),
                    const SizedBox(width: 12),
                    const Text(
                      'Deteksi Terbaru',
                      style: TextStyle(
                        fontSize: 18,
                        fontWeight: FontWeight.bold,
                      ),
                    ),
                  ],
                ),
```

```
                const SizedBox(height: 16),
                Center(
                  child: Column(
                    children: [
                      Icon(
                        Icons.show_chart,
                        size: 48,
                        color: Colors.grey[300],
                      ),
                      const SizedBox(height: 8),
                      Text(
                        'Belum ada deteksi hari ini',
                        style: TextStyle(
                          color: Colors.grey[400],
                          fontSize: 14,
                        ),
                      ),
                    ],
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    ),
  );
  }
}
```

Lampiran 5 Program Penyakitl Aplikasi S-dides

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

class PenyakitScreen extends StatelessWidget {
  const PenyakitScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      body: SingleChildScrollView(
        child: Column(
          children: [
```

```
            // Database Penyakit Section
            Container(
              margin: const EdgeInsets.all(16),
              padding: const EdgeInsets.all(20),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.circular(16),
                boxShadow: [
                  BoxShadow(
                    color: Colors.black.withAlpha(13),
                    blurRadius: 10,
                    offset: const Offset(0, 2),
                  ),
                ],
              ),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Row(
                    children: [
                      Icon(
                        Icons.warning_amber_rounded,
                        color: Colors.red[700],
                        size: 24,
                      ),
                      const SizedBox(width: 12),
                      const Text(
                        'Database Penyakit',
                        style: TextStyle(
                          fontSize: 18,
                          fontWeight: FontWeight.bold,
                        ),
                      ),
                    ],
                  ),
                  const SizedBox(height: 20),

                  // Disease List
                  _buildDiseaseItem(
                    context,
                    icon: Icons.coronavirus,
                    iconColor: const Color(0xFF4CAF50),
                    iconBgColor: const Color(0xFFE8F5E9),
                    title: 'White Spot Syndrome Virus',
                    subtitle: '(WSSV)',
```

```
                  status: 'Kritis',
                  statusColor: Colors.red,
                ),
                const SizedBox(height: 12),
                _buildDiseaseItem(
                  context,
                  icon: Icons.warning,
                  iconColor: Colors.orange,
                  iconBgColor: Colors.orange.shade50,
                  title: 'Taura Syndrome Virus',
                  subtitle: '(TSV)',
                  status: 'Tinggi',
                  statusColor: Colors.orange,
                ),
                const SizedBox(height: 12),
                _buildDiseaseItem(
                  context,
                  icon: Icons.search,
                  iconColor: Colors.blue,
                  iconBgColor: Colors.blue.shade50,
                  title: 'White Feces Disease',
                  subtitle: '(WFD)',
                  status: 'Sedang',
                  statusColor: Colors.amber[700]!,
                ),
                const SizedBox(height: 12),
                _buildDiseaseItem(
                  context,
                  icon: Icons.local_hospital,
                  iconColor: Colors.red,
                  iconBgColor: Colors.red.shade50,
                  title: 'Infectious Myonecrosis Virus',
                  subtitle: '(IMNV)',
                  status: 'Kritis',
                  statusColor: Colors.red,
                ),
              ],
            ),
          ),
        ],
      ),
    ),
  );
}
```

```
Widget _buildDiseaseItem(
  BuildContext context, {
  required IconData icon,
  required Color iconColor,
  required Color iconBgColor,
  required String title,
  required String subtitle,
  required String status,
  required Color statusColor,
}) {
  return InkWell(
    onTap: () {
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) => DiseaseDetailScreen(
            diseaseName: title,
            diseaseCode: subtitle,
            status: status,
            statusColor: statusColor,
            icon: icon,
            iconColor: iconColor,
            iconBgColor: iconBgColor,
          ),
        ),
      );
    },
    borderRadius: BorderRadius.circular(12),
    child: Container(
      padding: const EdgeInsets.all(16),
      decoration: BoxDecoration(
        color: Colors.grey[50],
        borderRadius: BorderRadius.circular(12),
        border: Border.all(color: Colors.grey.shade200),
      ),
      child: Row(
        children: [
          Container(
            padding: const EdgeInsets.all(10),
            decoration: BoxDecoration(
              color: iconBgColor,
              borderRadius: BorderRadius.circular(10),
            ),
            child: Icon(
              icon,
```

```
              color: iconColor,
              size: 24,
            ),
          ),
          const SizedBox(width: 12),
          Expanded(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text(
                  title,
                  style: const TextStyle(
                    fontSize: 14,
                    fontWeight: FontWeight.w600,
                  ),
                ),
                const SizedBox(height: 2),
                Text(
                  subtitle,
                  style: TextStyle(
                    fontSize: 12,
                    color: Colors.grey[600],
                  ),
                ),
              ],
            ),
          ),
          Container(
            padding: const EdgeInsets.symmetric(horizontal: 12,
vertical: 6),
            decoration: BoxDecoration(
              color: statusColor,
              borderRadius: BorderRadius.circular(20),
            ),
            child: Text(
              status,
              style: const TextStyle(
                color: Colors.white,
                fontSize: 12,
                fontWeight: FontWeight.bold,
              ),
            ),
          ),
          const SizedBox(width: 8),
          Icon(
```

```
              Icons.arrow_forward_ios,
              size: 16,
              color: Colors.grey[400],
            ),
          ],
        ),
      ),
    );
  }
}
```

Lampiran 6 Program Riwayat Aplikasi S-dides

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

class RiwayatScreen extends StatefulWidget {
  const RiwayatScreen({super.key});

  @override
  State<RiwayatScreen> createState() => _RiwayatScreenState();
}

class _RiwayatScreenState extends State<RiwayatScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      body: SingleChildScrollView(
        child: Column(
          children: [
            Container(
              margin: const EdgeInsets.all(16),
              padding: const EdgeInsets.all(20),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.circular(16),
                boxShadow: [
                  BoxShadow(
                    color: Colors.black.withAlpha(13),
                    blurRadius: 10,
                    offset: const Offset(0, 2),
                  ),
                ],
              ),
              width: double.infinity,
```

```
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                Row(
                  children: [
                    Container(
                      padding: const EdgeInsets.all(8),
                      child: const Icon(
                        Icons.schedule,
                        color: Colors.blue,
                        size: 26,
                      ),
                    ),
                    const SizedBox(width: 5),
                    const Text(
                      'Riwayat Deteksi',
                      style: TextStyle(
                        fontSize: 18,
                        fontWeight: FontWeight.bold,
                      ),
                    ),
                  ],
                ),
                const SizedBox(height: 20),
                // Riwayat List
                _buildEmptyState(),
              ],
            ),
          ),
        ],
      ),
    ),
  );
}

Widget _buildEmptyState() {
  return Container(
    padding: const EdgeInsets.all(40),
    child: Column(
      children: [
        Icon(Icons.schedule, size: 80, color: Colors.grey[300]),
        const SizedBox(height: 16),
        Text(
          'Belum ada riwayat',
          style: TextStyle(
```

```
              fontSize: 16,
              fontWeight: FontWeight.w500,
              color: Colors.grey[600],
            ),
          ),
          const SizedBox(height: 8),
          Text(
            'Mulai deteksi untuk melihat riwayat',
            style: TextStyle(fontSize: 14, color: Colors.grey[500]),
          ),
        ],
      ),
    );
  }
}
```

Lampiran 7 Program Pengaturan Aplikasi S-dides

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

class SettingsScreen extends StatefulWidget {
  const SettingsScreen({super.key});

  @override
  State<SettingsScreen> createState() => _SettingsScreenState();
}

class _SettingsScreenState extends State<SettingsScreen> {
  final TextEditingController _serverUrlController =
TextEditingController();
  final TextEditingController _portController =
TextEditingController();
  bool _isLoading = true;
  bool _isSaving = false;
  String _connectionStatus = 'Belum diperiksa';
  Color _statusColor = Colors.grey;

  @override
  void initState() {
    super.initState();
    _loadSettings();
  }
```

```
  Future<void> _loadSettings() async {
    setState(() => _isLoading = true);
    try {
      final prefs = await SharedPreferences.getInstance();
      _serverUrlController.text = prefs.getString('server_url') ??
'http://192.168.1.100';
      _portController.text = prefs.getString('server_port') ??
'5000';
    } catch (e) {
      _showSnackBar('Gagal memuat pengaturan: $e', Colors.red);
    } finally {
      setState(() => _isLoading = false);
    }
  }

  Future<void> _saveSettings() async {
    if (_serverUrlController.text.isEmpty ||
_portController.text.isEmpty) {
      _showSnackBar('URL Server dan Port tidak boleh kosong',
Colors.orange);
      return;
    }

    setState(() => _isSaving = true);
    try {
      final prefs = await SharedPreferences.getInstance();
      await prefs.setString('server_url',
_serverUrlController.text);
      await prefs.setString('server_port', _portController.text);
      _showSnackBar('Pengaturan berhasil disimpan', const
Color(0xFF00BFA5));

      // Return the new URL to the previous screen
      if (mounted) {
        Navigator.pop(context, getFullServerUrl());
      }
    } catch (e) {
      _showSnackBar('Gagal menyimpan pengaturan: $e', Colors.red);
    } finally {
      setState(() => _isSaving = false);
    }
  }

  String getFullServerUrl() {
```

```
    return '${_serverUrlController.text}:${_portController.text}';
  }

  Future<void> _testConnection() async {
    if (_serverUrlController.text.isEmpty ||
_portController.text.isEmpty) {
      _showSnackBar('Masukkan URL Server dan Port terlebih dahulu',
Colors.orange);
      return;
    }

    setState(() {
      _connectionStatus = 'Memeriksa koneksi...';
      _statusColor = Colors.orange;
    });

    try {
      // Attempt to connect to server with timeout
      final url = Uri.parse('${getFullServerUrl()}');

      final response = await http.get(url).timeout(
        const Duration(seconds: 5),
        onTimeout: () {
          throw TimeoutException('Koneksi timeout');
        },
      );

      if (response.statusCode == 200) {
        setState(() {
          _connectionStatus = 'Terhubung';
          _statusColor = const Color(0xFF00BFA5);
        });
        _showSnackBar('Koneksi berhasil! Server merespons dengan
baik.', const Color(0xFF00BFA5));
      } else {
        setState(() {
          _connectionStatus = 'Gagal (Status:
${response.statusCode})';
          _statusColor = Colors.red;
        });
        _showSnackBar('Server merespons dengan error:
${response.statusCode}', Colors.red);
      }
    } on TimeoutException {
      setState(() {
```

```
        _connectionStatus = 'Timeout';
        _statusColor = Colors.red;
      });
      _showSnackBar('Koneksi timeout. Pastikan server aktif dan URL
benar.', Colors.red);
    } catch (e) {
      setState(() {
        _connectionStatus = 'Gagal terhubung';
        _statusColor = Colors.red;
      });
      _showSnackBar('Gagal terhubung: ${e.toString()}', Colors.red);
    }
  }

  void _showSnackBar(String message, Color color) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: Text(message),
        backgroundColor: color,
        behavior: SnackBarBehavior.floating,
        margin: const EdgeInsets.all(16),
        shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(8)),
      ),
    );
  }

  void _resetToDefault() {
    setState(() {
      _serverUrlController.text = 'http://192.168.1.100';
      _portController.text = '5000';
      _connectionStatus = 'Belum diperiksa';
      _statusColor = Colors.grey;
    });
    _showSnackBar('Pengaturan direset ke default', Colors.blue);
  }

  @override
  void dispose() {
    _serverUrlController.dispose();
    _portController.dispose();
    super.dispose();
  }

  @override
```

```
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.grey[100],
      appBar: AppBar(
        backgroundColor: const Color(0xFF00BFA5),
        elevation: 0,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.white),
          onPressed: () => Navigator.pop(context),
        ),
        title: const Text(
          'Pengaturan Server',
          style: TextStyle(color: Colors.white, fontWeight:
FontWeight.bold),
        ),
        actions: [
          IconButton(
            icon: const Icon(Icons.refresh, color: Colors.white),
            onPressed: _resetToDefault,
            tooltip: 'Reset ke Default',
          ),
        ],
      ),
      body: _isLoading
          ? const Center(child: CircularProgressIndicator())
          : SingleChildScrollView(
              padding: const EdgeInsets.all(16),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  // Info Card
                  Container(
                    padding: const EdgeInsets.all(16),
                    decoration: BoxDecoration(
                      color: const Color(0xFFE3F2FD),
                      borderRadius: BorderRadius.circular(12),
                      border: Border.all(color:
Colors.blue.shade200),
                    ),
                    child: Row(
                      children: [
                        Icon(Icons.info_outline, color:
Colors.blue[700]),
                        const SizedBox(width: 12),
                        const Expanded(
```

```
                          child: Text(
                            'Konfigurasikan URL server untuk deteksi
penyakit udang',
                            style: TextStyle(fontSize: 12, height:
1.4),
                          ),
                        ),
                      ],
                    ),
                  ),
                  const SizedBox(height: 24),

                  // Server Configuration Card
                  Container(
                    padding: const EdgeInsets.all(20),
                    decoration: BoxDecoration(
                      color: Colors.white,
                      borderRadius: BorderRadius.circular(16),
                      boxShadow: [
                        BoxShadow(
                          color: Colors.black.withAlpha(13),
                          blurRadius: 10,
                          offset: const Offset(0, 2),
                        ),
                      ],
                    ),
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Row(
                          children: [
                            Container(
                              padding: const EdgeInsets.all(8),
                              decoration: BoxDecoration(
                                color: const
Color(0xFF00BFA5).withAlpha(25),
                                borderRadius:
BorderRadius.circular(8),
                              ),
                              child: const Icon(
                                Icons.dns,
                                color: Color(0xFF00BFA5),
                                size: 20,
                              ),
                            ),
```

```
                            const SizedBox(width: 12),
                            const Text(
                              'Konfigurasi Server',
                              style: TextStyle(
                                fontSize: 16,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 20),

                        // Server URL Input
                        const Text(
                          'URL Server',
                          style: TextStyle(
                            fontSize: 14,
                            fontWeight: FontWeight.w600,
                          ),
                        ),
                        const SizedBox(height: 8),
                        TextField(
                          controller: _serverUrlController,
                          decoration: InputDecoration(
                            hintText: 'http://192.168.1.100',
                            hintStyle: TextStyle(color:
Colors.black.withAlpha(80)),
                            prefixIcon: const Icon(Icons.link,
color: Color(0xFF00BFA5)),
                            filled: true,
                            fillColor: Colors.grey[50],
                            border: OutlineInputBorder(
                              borderRadius:
BorderRadius.circular(12),
                              borderSide: BorderSide(color:
Colors.grey.shade300),
                            ),
                            enabledBorder: OutlineInputBorder(
                              borderRadius:
BorderRadius.circular(12),
                              borderSide: BorderSide(color:
Colors.grey.shade300),
                            ),
                            focusedBorder: OutlineInputBorder(
```

```
                              borderRadius:
BorderRadius.circular(12),
                              borderSide: const BorderSide(color:
Color(0xFF00BFA5), width: 2),
                            ),
                          ),
                        ),
                        const SizedBox(height: 16),

                        // Port Input
                        const Text(
                          'Port',
                          style: TextStyle(
                            fontSize: 14,
                            fontWeight: FontWeight.w600,
                          ),
                        ),
                        const SizedBox(height: 8),
                        TextField(
                          controller: _portController,
                          keyboardType: TextInputType.number,
                          decoration: InputDecoration(
                            hintText: '5000',
                            hintStyle: TextStyle(color:
Colors.black.withAlpha(80)),
                            prefixIcon: const
Icon(Icons.settings_ethernet, color: Color(0xFF00BFA5)),
                            filled: true,
                            fillColor: Colors.grey[50],
                            border: OutlineInputBorder(
                              borderRadius:
BorderRadius.circular(12),
                              borderSide: BorderSide(color:
Colors.grey.shade300),
                            ),
                            enabledBorder: OutlineInputBorder(
                              borderRadius:
BorderRadius.circular(12),
                              borderSide: BorderSide(color:
Colors.grey.shade300),
                            ),
                            focusedBorder: OutlineInputBorder(
                              borderRadius:
BorderRadius.circular(12),
```

```
                              borderSide: const BorderSide(color:
Color(0xFF00BFA5), width: 2),
                            ),
                          ),
                        ),
                        const SizedBox(height: 20),

                        // Full URL Preview
                        Container(
                          padding: const EdgeInsets.all(12),
                          decoration: BoxDecoration(
                            color: Colors.grey[100],
                            borderRadius: BorderRadius.circular(8),
                            border: Border.all(color:
Colors.grey.shade300),
                          ),
                          child: Row(
                            children: [
                              Icon(Icons.check_circle, size: 16,
color: Colors.grey[600]),
                              const SizedBox(width: 8),
                              Expanded(
                                child: Text(
                                  'Full URL: ${getFullServerUrl()}',
                                  style: TextStyle(
                                    fontSize: 12,
                                    color: Colors.grey[700],
                                    fontFamily: 'monospace',
                                  ),
                                ),
                              ),
                            ],
                          ),
                        ),
                      ],
                    ),
                  ),
                  const SizedBox(height: 16),

                  // Connection Status Card
                  Container(
                    padding: const EdgeInsets.all(20),
                    decoration: BoxDecoration(
                      color: Colors.white,
                      borderRadius: BorderRadius.circular(16),
```

```
                      boxShadow: [
                        BoxShadow(
                          color: Colors.black.withAlpha(13),
                          blurRadius: 10,
                          offset: const Offset(0, 2),
                        ),
                      ],
                    ),
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Row(
                          children: [
                            Container(
                              padding: const EdgeInsets.all(8),
                              decoration: BoxDecoration(
                                color: _statusColor.withAlpha(25),
                                borderRadius:
BorderRadius.circular(8),
                              ),
                              child: Icon(
                                Icons.wifi_tethering,
                                color: _statusColor,
                                size: 20,
                              ),
                            ),
                            const SizedBox(width: 12),
                            const Text(
                              'Status Koneksi',
                              style: TextStyle(
                                fontSize: 16,
                                fontWeight: FontWeight.bold,
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 16),
                        Row(
                          mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                          children: [
                            Text(
                              _connectionStatus,
                              style: TextStyle(
                                fontSize: 14,
```

```
                              color: _statusColor,
                              fontWeight: FontWeight.w600,
                            ),
                          ),
                          Container(
                            width: 12,
                            height: 12,
                            decoration: BoxDecoration(
                              color: _statusColor,
                              shape: BoxShape.circle,
                            ),
                          ),
                        ],
                      ),
                      const SizedBox(height: 16),
                      SizedBox(
                        width: double.infinity,
                        child: OutlinedButton.icon(
                          onPressed: _testConnection,
                          icon: const Icon(Icons.wifi_find),
                          label: const Text('Test Koneksi'),
                          style: OutlinedButton.styleFrom(
                            foregroundColor: const
Color(0xFF00BFA5),
                            side: const BorderSide(color:
Color(0xFF00BFA5)),
                            padding: const
EdgeInsets.symmetric(vertical: 12),
                            shape: RoundedRectangleBorder(
                              borderRadius:
BorderRadius.circular(12),
                            ),
                          ),
                        ),
                      ),
                    ],
                  ),
                ),
                const SizedBox(height: 24),

                // Save Button
                SizedBox(
                  width: double.infinity,
                  child: ElevatedButton(
                    onPressed: _isSaving ? null : _saveSettings,
```

```
                      style: ElevatedButton.styleFrom(
                        backgroundColor: const Color(0xFF00BFA5),
                        padding: const
EdgeInsets.symmetric(vertical: 16),
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(12),
                        ),
                        disabledBackgroundColor: Colors.grey,
                      ),
                      child: _isSaving
                          ? const SizedBox(
                              height: 20,
                              width: 20,
                              child: CircularProgressIndicator(
                                color: Colors.white,
                                strokeWidth: 2,
                              ),
                            )
                          : const Row(
                              mainAxisAlignment:
MainAxisAlignment.center,
                              children: [
                                Icon(Icons.save, color:
Colors.white),
                                SizedBox(width: 8),
                                Text(
                                  'Simpan Pengaturan',
                                  style: TextStyle(
                                    fontSize: 16,
                                    fontWeight: FontWeight.bold,
                                    color: Colors.white,
                                  ),
                                ),
                              ],
                            ),
                    ),
                  ),
                ],
              ),
            ),
    );
  }
}
```

Lampiran 8 Program Maindata Aplikasi S-dides

```
import 'package:flutter/material.dart';
import 'screens/monitor_screen.dart';
import 'screens/penyakit_screen.dart';
import 'screens/riwayat_screen.dart';
import 'widgets/app_header.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'S-DIDES Monitor',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: const
Color(0xFF00BFA5)),
        useMaterial3: true,
      ),
      home: const MainScreen(),
    );
  }
}

class MainScreen extends StatefulWidget {
  const MainScreen({super.key});

  @override
  State<MainScreen> createState() => _MainScreenState();
}

class _MainScreenState extends State<MainScreen> {
  int _selectedIndex = 0;

  final List<Widget> _screens = [
    const MonitorScreen(),
    const PenyakitScreen(),
    const RiwayatScreen(),
  ];
```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.grey[100],
    body: Column(
      children: [
        const AppHeader(),
        // Tab Navigation
        Container(
          margin: const EdgeInsets.all(16),
          padding: const EdgeInsets.all(4),
          decoration: BoxDecoration(
            color: Colors.white,
            borderRadius: BorderRadius.circular(12),
            boxShadow: [
              BoxShadow(
                color: Colors.black.withOpacity(0.05),
                blurRadius: 10,
                offset: const Offset(0, 2),
              ),
            ],
          ),
          child: Row(
            children: [
              Expanded(
                child: _buildTabButton(
                  index: 0,
                  icon: Icons.monitor,
                  label: 'Monitor',
                ),
              ),
              Expanded(
                child: _buildTabButton(
                  index: 1,
                  icon: Icons.warning_amber_rounded,
                  label: 'Penyakit',
                ),
              ),
              Expanded(
                child: _buildTabButton(
                  index: 2,
                  icon: Icons.schedule,
                  label: 'Riwayat',
                ),
```

```
                ),
              ],
            ),
          ),
          Expanded(
            child: _screens[_selectedIndex],
          ),
        ],
      ),
    );
  }

  Widget _buildTabButton({
    required int index,
    required IconData icon,
    required String label,
  }) {
    final isSelected = _selectedIndex == index;

    List<Color> gradientColors;
    switch (index) {
      case 0:
        gradientColors = [const Color(0xFF00BFA5), const
Color(0xFF00897B)];
        break;
      case 1:
        gradientColors = [const Color(0xFFEF5350), const
Color(0xFFE53935)];
        break;
      case 2:
        gradientColors = [const Color(0xFF2196F3), const
Color(0xFF1976D2)];
        break;
      default:
        gradientColors = [const Color(0xFF00BFA5), const
Color(0xFF00897B)];
    }

    return GestureDetector(
      onTap: () {
        setState(() {
          _selectedIndex = index;
        });
      },
      child: Container(
```

```
        padding: const EdgeInsets.symmetric(vertical: 12),
        decoration: BoxDecoration(
          gradient: isSelected
              ? LinearGradient(
                  colors: gradientColors,
                  begin: Alignment.bottomLeft,
                  end: Alignment.topRight,
                )
              : null,
          color: isSelected ? null : Colors.transparent,
          borderRadius: BorderRadius.circular(8),
        ),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Icon(
              icon,
              color: isSelected ? Colors.white : Colors.grey[600],
              size: 18,
            ),
            const SizedBox(width: 6),
            Text(
              label,
              style: TextStyle(
                color: isSelected ? Colors.white : Colors.grey[600],
                fontWeight: isSelected ? FontWeight.bold :
FontWeight.normal,
                fontSize: 14,
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Lampiran 9 Program Traing data Model Yolov11n

```
!pip install ultralytics roboflow -q

# 2. Import libraries
from ultralytics import YOLO
from roboflow import Roboflow
import torch
```

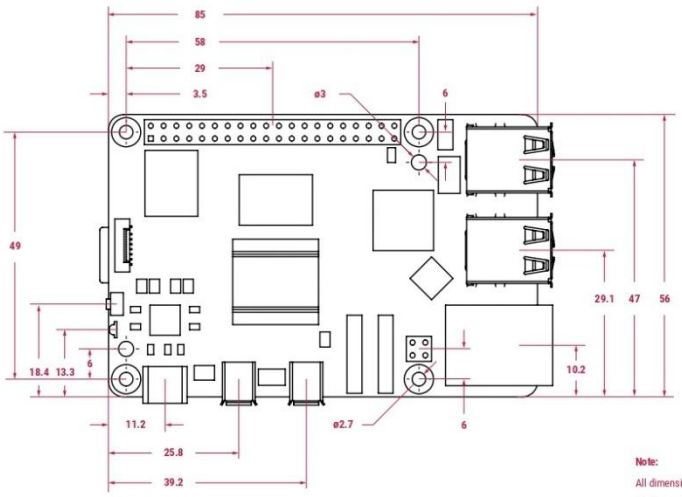
```
import os
from pathlib import Path
print(f"PyTorch version: {torch.__version__}")
print(f"CUDA available: {torch.cuda.is_available()}")
if torch.cuda.is_available():
    print(f"GPU: {torch.cuda.get_device_name(0)}")
    print(f"GPU Memory:
{torch.cuda.get_device_properties(0).total_memory / 1024**3:.2f}
GB")

print("\n📥 Downloading dataset from Roboflow...")
rf = Roboflow(api_key="ShMdmvpZMZ7N50ToLx34")
project = rf.workspace("aaa-fmxpv").project("s-dides-2-1-46jzf")
version = project.version(10)
dataset = version.download("yolov11")

# Get dataset path
dataset_path = dataset.location
data_yaml = f"{dataset_path}/data.yaml"
print(f"☑ Dataset downloaded to: {dataset_path}")’
print("\n🚀 Loading YOLOv11n model...")
model = YOLO('yolo11n.pt')  # Load pretrained YOLOv11n

# 6. Training dengan konfigurasi optimal
print("\n🎯 Starting training with optimized parameters...")
results = model.train(
    data=data_yaml,

    # Model & Training Configuration
    epochs=100,              # Jumlah epoch (sesuaikan dengan
dataset Anda)
    imgsz=640,              # Image size (640 adalah standar
optimal)
    batch=16,               # Batch size (sesuaikan dengan VRAM GPU)

    # Optimizer Settings
    optimizer='AdamW',      # AdamW optimizer untuk hasil lebih baik
    lr0=0.001,              # Initial learning rate
    lrf=0.01,               # Final learning rate (lr0 * lrf)
    momentum=0.937,         # Momentum
    weight_decay=0.0005,    # Weight decay untuk regularisasi

    # Augmentation (penting untuk hasil bagus!)
    hsv_h=0.015,           # HSV-Hue augmentation
    hsv_s=0.7,             # HSV-Saturation augmentation
```

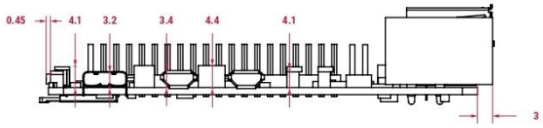
```
    hsv_v=0.4,              # HSV-Value augmentation
    degrees=0.0,            # Rotation augmentation
    translate=0.1,          # Translation augmentation
    scale=0.5,              # Scale augmentation
    shear=0.0,              # Shear augmentation
    perspective=0.0,        # Perspective augmentation
    flipud=0.0,             # Flip up-down probability
    fliplr=0.5,             # Flip left-right probability
    mosaic=1.0,             # Mosaic augmentation probability
    mixup=0.0,              # Mixup augmentation probability
    copy_paste=0.0,         # Copy-paste augmentation

    # Training Settings
    patience=50,            # Early stopping patience (epochs)
    save=True,              # Save checkpoints
    save_period=10,         # Save checkpoint every N epochs
    cache=True,             # Cache images untuk training lebih cepat
    device=0,               # GPU device (0 untuk GPU pertama, 'cpu'
untuk CPU)
    workers=8,              # Number of worker threads

    # Validation
    val=True,               # Validate during training

    # Output
    project='yolo_training',    # Project name
    name='yolov11n_run',        # Run name
    exist_ok=True,              # Overwrite existing project
    pretrained=True,            # Use pretrained weights
    verbose=True,               # Verbose output

    # Performance optimization
    amp=True,               # Automatic Mixed Precision training
(lebih cepat)
    fraction=1.0,           # Dataset fraction to use
    profile=False,          # Profile ONNX model

    # Advanced
    cos_lr=True,            # Cosine learning rate scheduler
    close_mosaic=10,        # Disable mosaic N epochs before end
    resume=False,           # Resume training dari last checkpoint
    overlap_mask=True,      # Overlap mask untuk segmentation (jika
ada)
    mask_ratio=4,           # Mask downsample ratio
    dropout=0.0,            # Dropout regularization
```

```
    # Label settings
    label_smoothing=0.0,   # Label smoothing epsilon
    nbs=64,                # Nominal batch size
    single_cls=False,      # Train as single-class dataset
)

print("\n📊 Training completed!")
print(f"Results saved to: {results.save_dir}")

# 8. Validate model
print("\n☑️ Validating model on test set...")
metrics = model.val()

# Print metrics
print("\n📈 Validation Metrics:")
print(f"mAP50: {metrics.box.map50:.4f}")
print(f"mAP50-95: {metrics.box.map:.4f}")
print(f"Precision: {metrics.box.mp:.4f}")
print(f"Recall: {metrics.box.mr:.4f}")

# 9. Export model (opsional)
print("\n💾 Exporting model...")
# Uncomment format yang Anda butuhkan:
model.export(format='onnx')      # ONNX format
```

Lampiran 10 Datasheet Raspberry Pi 5

Physical specification





Note:

All dimensions in mm

All dimensions are approximate and for reference purposes only. The dimensions shown should not be used for producing production data

The dimensions are subject to part and manufacturing tolerances

Not all of the board components are shown. Please reference a physical board for representation of componentry

Dimensions may be subject to change

WARNINGS

- This product should be operated in a well ventilated environment, and if used inside a case, the case should not be covered.
- While in use, this product should be firmly secured or should be placed on a stable, flat, non-conductive surface, and should not be contacted by conductive items.
- The connection of incompatible devices to Raspberry Pi 5 may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met.

SAFETY INSTRUCTIONS

To avoid malfunction or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface while in operation.
- Do not expose to heat from any source; Raspberry Pi 5 is designed for reliable operation at normal ambient temperatures.
- Store in a cool, dry location.
- Take care while handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- While it is powered, avoid handling the printed circuit board, or handle it only by the edges, to minimise the risk of electrostatic discharge damage.

Specification

Processor:	Broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76 CPU, with Cryptographic Extension, 512KB per-core L2 caches, and a 2MB shared L3 cache
Features:	• VideoCore VII GPU, supporting OpenGL ES 3.1, Vulkan 1.2 • Dual 4Kp60 HDMI® display output with HDR support • 4Kp60 HEVC decoder • LPDDR4X-4267 SDRAM (options for 1GB, 2GB, 4GB, 8GB and 16GB) • Dual-band 802.11ac Wi-Fi® • Bluetooth 5.0 / Bluetooth Low Energy (BLE) • microSD card slot, with support for high-speed SDR104 mode • 2 × USB 3.0 ports, supporting simultaneous 5Gbps operation • 2 × USB 2.0 ports • Gigabit Ethernet, with PoE+ support (requires separate PoE+ HAT) • 2 × 4-lane MIPI camera/display transceivers • PCIe 2.0 x1 interface for fast peripherals (requires separate M.2 HAT or other adapter) • 5V/5A DC power via USB-C, with Power Delivery support • Raspberry Pi standard 40-pin header • Real-time clock (RTC), powered from external battery • Power button
MTBF[1] Ground Benign:	93 800 hours
Operating temperature:	0°C to 70°C
Production lifetime:	Raspberry Pi 5 will remain in production until at least January 2036
Compliance:	For a full list of local and regional product approvals, please visit pip.raspberrypi.com
List price:	1GB **$45** 2GB **$65** 4GB **$110** 8GB **$175** 16GB **$305**

[1] Mean Time Between Failure

Overview

Welcome to the latest generation of Raspberry Pi: the everything computer.

Featuring a 64-bit quad-core Arm Cortex-A76 processor running at 2.4GHz, Raspberry Pi 5 delivers a 2–3× increase in CPU performance relative to Raspberry Pi 4. Alongside a substantial uplift in graphics performance from an 800MHz VideoCore VII GPU; dual 4Kp60 display output over HDMI; and state-of-the-art camera support from a rearchitected Raspberry Pi Image Signal Processor, it provides a smooth desktop experience for consumers, and opens the door to new applications for industrial customers.

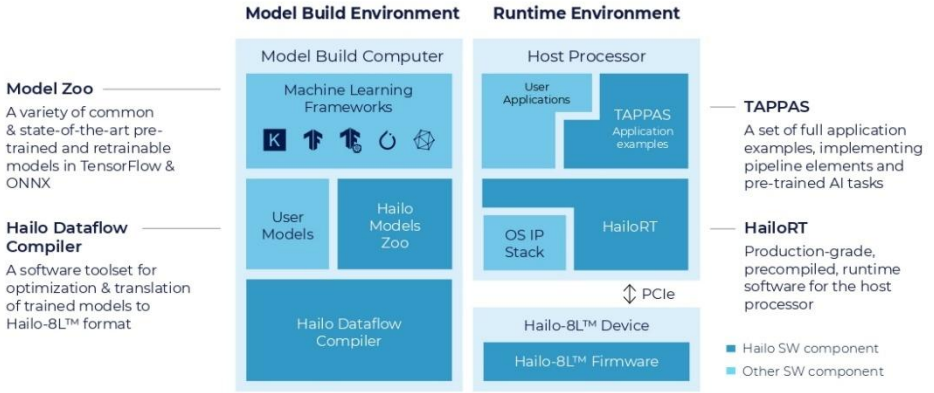
For the first time, this is a full-size Raspberry Pi computer using silicon built in-house at Raspberry Pi. The RP1 "southbridge" provides the bulk of the I/O capabilities for Raspberry Pi 5, and delivers a step change in peripheral performance and functionality. Aggregate USB bandwidth is more than doubled, yielding faster transfer speeds to external UAS drives and other high-speed peripherals; the dedicated two-lane 1Gbps MIPI camera and display interfaces present on earlier models have been replaced by a pair of four-lane 1.5Gbps MIPI transceivers, tripling total bandwidth, and supporting any combination of up to two cameras or displays; peak SD card performance is doubled, through support for the SDR104 high-speed mode; and for the first time the platform exposes a single-lane PCI Express 2.0 interface, providing support for high-bandwidth peripherals.

Lampiran 11 Datasheet Hailo 8l

Hailo-8L™ Entry-Level M.2 AI Acceleration Modules

Comprehensive Software Suite

AI software suite which seamlessly integrates with existing deep learning development frameworks to allow smooth and easy integration in existing development ecosystems. The Hailo AI software suite is future proof, supporting all Hailo-8™ product lines, and includes:



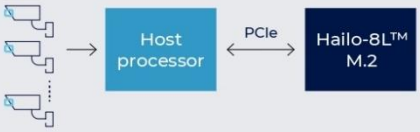
Hailo Ecosystem

Hailo collaborates with partners globally to offer products for edge solutions — from small and fanless devices, to ruggedized industrial appliances, high capacity video analytics platforms, embedded computing platforms for automotive & robotics and more.

To select your Hailo-based AI platform click here

System Usage

M.2 module connected to the various types of host processors via PCIe.



Industries

Industrial Automation

Security

Retail

Click here for more information & product inquiry

HAILO

HAILO

Hailo-8L™ Entry-Level M.2 AI Acceleration Modules

Delivering data center class performance to edge devices

Key Features & Benefits

Powered by 13 Tera-Operations Per Second (TOPS) Hailo-8L™ AI processor

Enabling real-time, low latency, and high-efficiency AI inferencing on edge devices

Fast time to market using a standard M.2 form factor module, with key B+M & key A+E

Best-in-class power efficiency

High cost-efficiency (TOPS/$) compared with existing solutions

Supporting extended temperature range of -40°C to 85°C

Robust software suite supports state-of-the-art deep learning models & applications out-of-the-box

Scalable:

- Enabling simultaneous processing of multi-streams & multi-models
- Future-ready — software compatibility when migrating to Hailo-8™ for more powerful AI

Technical Specifications

Form factor options
M.2 key B+M, key A+E

Dimensions
- Key B+M: 22×42 mm with breakable extensions to 22×60 mm & 22×80 mm
- Key A+E: 22×30 mm

Interface
PCIe Gen-3.0, 2-lanes

Supported host architectures
X86 or ARM based

Supported OS
Linux, Windows

Supported AI frameworks
TensorFlow, TensorFlow Lite, Keras, PyTorch & ONNX

Hailo-8L™ M.2 AI Acceleration Modules

Compatible with M.2 form factor, the Hailo-8L™ based modules can be plugged into an existing edge device with an M.2 socket to execute deep neural network inferencing in real-time utilizing low power for a broad range of market segments.

M.2 Key B+M
(2242/2260/2280)

M.2 Key A+E
(2230)

Featuring the Hailo-8L™ AI Processor

The Hailo-8L™ AI processor, delivering up to 13 tera-operations per second (TOPS), significantly outperforms all other edge AI processors. Its area and power efficiency are far superior to other leading solutions by an order of magnitude.

Hailo-8L™ unique, powerful and scalable structure-driven dataflow architecture takes advantage of the core properties of neural networks. It enables edge devices to run deep learning applications at full scale more efficiently, effectively, and substantially than traditional solutions, while significantly lowering costs.

www.hailo.ai

Lampiran 12 Datasheet HQ Camera

WARNINGS

- This product should be operated in a well ventilated environment, and if used inside a case, the case should not be covered.
- Whilst in use, this product should be firmly secured or should be placed on a stable, flat, non-conductive surface, and should not be contacted by conductive items.
- The connection of incompatible devices to the Raspberry Pi High Quality Camera may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met.

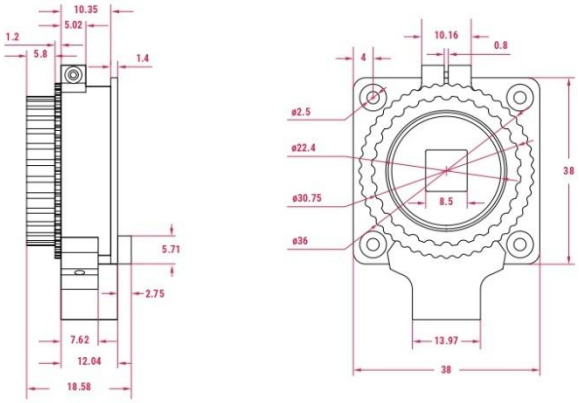
SAFETY INSTRUCTIONS

To avoid malfunction or damage to this product, please observe the following:

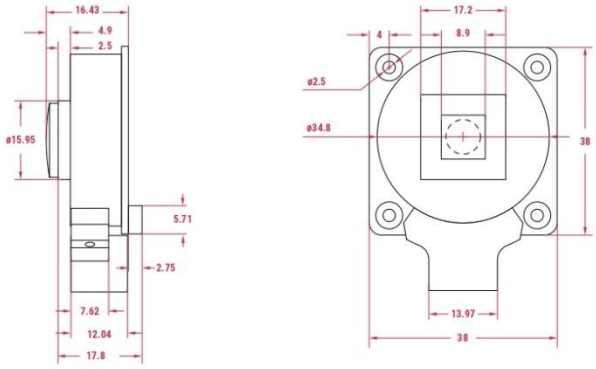
- Important: Before connecting this device, shut down your Raspberry Pi computer and disconnect it from external power.
- If the cable becomes detached, first pull forward the locking mechanism on the connector, then insert the ribbon cable ensuring that the metal contacts face towards the circuit board, and finally push the locking mechanism back into place.
- This device should be operated in a dry environment at 0–50°C.
- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi High Quality Camera is designed for reliable operation at normal ambient temperatures.
- Store in a cool, dry location.
- Avoid rapid changes of temperature, which can cause moisture to build up in the device, affecting image quality.
- Take care not to fold or strain the ribbon cable.
- Take care when screwing in parts or fitting a tripod. A cross-thread can cause irreparable damage and void the warranty.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or handle it only by the edges, to minimise the risk of electrostatic discharge damage.

Physical specification

CS Mount



M12 Mount



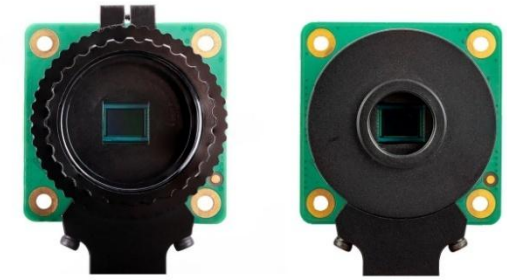
Note: all dimensions in mm

Specification

Sensor:	Sony IMX477R stacked, back-illuminated sensor
Resolution:	12.3 megapixels
Sensor size:	7.9mm sensor diagonal
Pixel size:	1.55µm × 1.55µm
Output:	RAW12/10/8, COMP8
Back focus length of lens:	2.6mm–11.8mm (M12 Mount variant) 12.5mm–22.4mm (CS Mount variant)
Lens sensor format:	1/2.3" (7.9mm) or larger
IR cut filter:	Integrated [2]
Ribbon cable length:	200mm
Tripod mount:	1/4"-20
Compliance:	FCC 47 CFR Part 15, Subpart B, Class B Digital Device Electromagnetic Compatibility Directive (EMC) 2014/30/EU Restriction of Hazardous Substances (RoHS) Directive 2011/65/EU
Production lifetime:	The Raspberry Pi High Quality Camera will remain in production until at least January 2030

[2] Can be removed to enable IR sensitivity. Modification is irreversible.

Overview



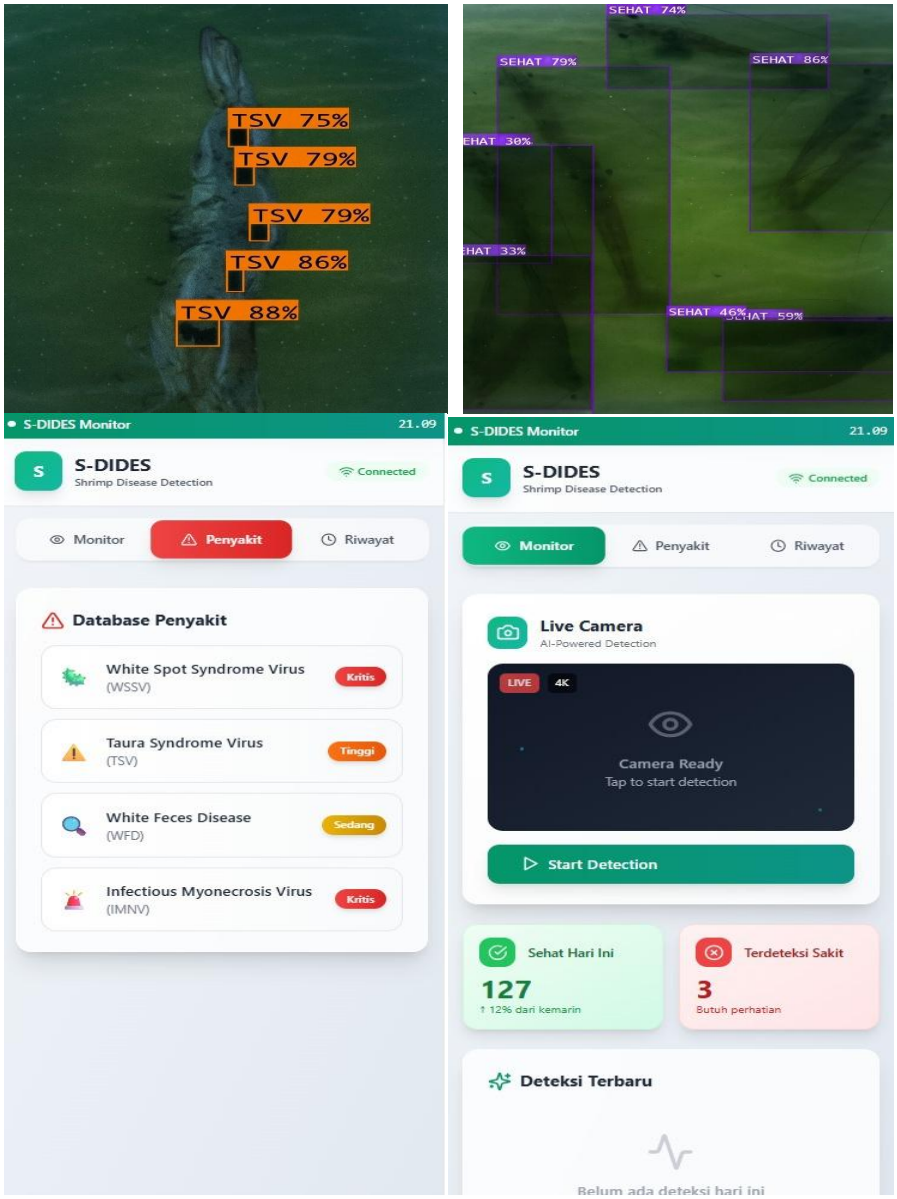
The Raspberry Pi High Quality Camera is an affordable high-quality camera from Raspberry Pi. It offers 12-megapixel resolution and a 7.9mm-diagonal sensor for impressive low-light performance. The M12 Mount variant is designed to work with most interchangeable M12 lenses, and the CS Mount variant is designed to work with interchangeable lenses in both CS- and C-mount form factors (C-mount lenses require the use of the C-CS adapter included with this variant). Other lens form factors can be accommodated using third-party lens adapters.

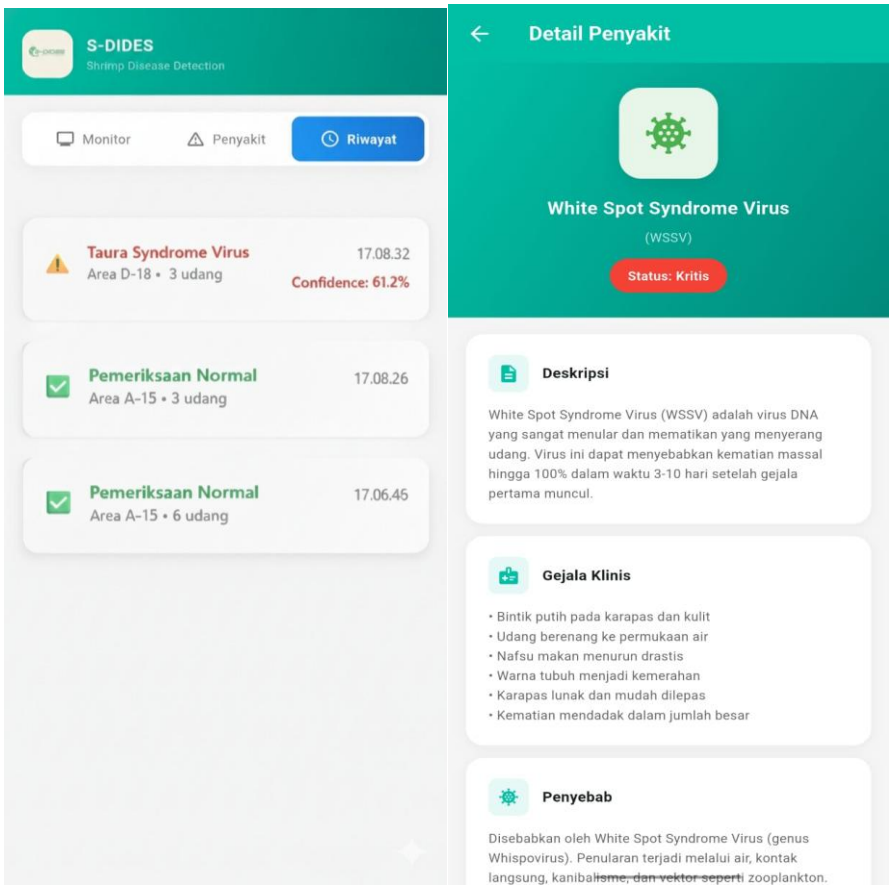
The High Quality Camera is well suited to industrial and consumer applications, including security cameras, which require the highest levels of visual fidelity and/or integration with specialist optics. It is compatible with all models of Raspberry Pi computer from Raspberry Pi 1 Model B onwards, using the latest software release from raspberrypi.com.[1]

The package comprises a circuit board carrying a Sony IMX477 sensor, an FPC cable for connection to a Raspberry Pi computer, and a milled aluminium lens mount with integrated tripod mount. The CS Mount variant lens mount features a focus adjustment ring, and this variant ships with a C- to CS-mount adapter; the M12 Mount variant ships with three lens locking rings (one required plus two spare).

[1] Excluding early Raspberry Pi Zero models, which lack the necessary FPC connector. Later Raspberry Pi Zero models require an adapter FPC, sold separately.

Lampiran 13 Prototipe yang Diciptakan




S-DIDES
Shrimp Disease Detection
Monitor
Penyakit
Riwayat
Taura Syndrome Virus
Area D-18 • 3 udang
17.08.32
Confidence: 61.2%
Pemeriksaan Normal
Area A-15 • 3 udang
17.08.26
Pemeriksaan Normal
Area A-15 • 6 udang
17.06.45
Detail Penyakit
White Spot Syndrome Virus
(WSSV)
Status: Kritis
Deskripsi
White Spot Syndrome Virus (WSSV) adalah virus DNA yang sangat menular dan mematikan yang menyerang udang. Virus ini dapat menyebabkan kematian massal hingga 100% dalam waktu 3-10 hari setelah gejala pertama muncul.
Gejala Klinis
• Bintik putih pada karapas dan kulit
• Udang berenang ke permukaan air
• Nafsu makan menurun drastis
• Warna tubuh menjadi kemerahan
• Karapas lunak dan mudah dilepas
• Kematian mendadak dalam jumlah besar
Penyebab
Disebabkan oleh White Spot Syndrome Virus (genus Whispovirus). Penularan terjadi melalui air, kontak langsung, kanibalisme, dan vektor seperti zooplankton.