

BAB IV PEMBAHASAN

4.1 Gambaran Umum Obyek Penelitian

Digital Imigrasi merupakan sebuah aplikasi yang dirancang untuk Direktorat Keimigrasian dengan tujuan menciptakan versi digital dari dokumen kartu identitas yang digunakan pada layanan keimigrasian. Digital Imigrasi menggunakan sistem OCR untuk pembuatan identitas digital seperti paspor. Penggunaan OCR digunakan untuk mempermudah pengguna dalam pengambilan data untuk pembuatan digital ID.

Pengambilan data paspor memiliki kendala yang membuat pengguna kesusahan dalam penggunaan OCR dikarenakan pengambilan data paspor masih dilakukan secara manual dengan memindai baris MRZ pada paspor. Pengambilan data paspor secara manual memiliki banyak kelemahan di mana kita harus benar-benar mendapatkan baris data MRZ yang di tangkap oleh kotak pemindai yang diberikan. Apabila pengguna tidak dapat mendapatkan data yang sesuai dari data yang ada, digital imigrasi akan menampilkan peringatan untuk mengambil ulang gambar hingga benar-benar mendapatkan semua data dari baris MRZ.

Pemanfaatan OCR yang digabungkan dengan *auto-capture* dan MRZ parse menjadi solusi dalam masalah yang diberikan. Dengan menggunakan *auto-capture* pada sistem OCR yang digunakan mempermudah pengguna, karena dengan menggunakan *auto-capture* pengguna tidak perlu sulit mengambil data secara manual. Pengambilan data akan dilakukan oleh sistem pengecekan langsung dari OCR dengan memanfaatkan *auto-capture* apabila data sudah didapatkan. Untuk

mengetahui sistem OCR sudah mendapatkan baris MRZ atau belum menjadi tantangan dalam penelitian kali ini, dikarenakan pada penelitian ini MRZ parse juga memecahkan baris MRZ untuk paspor WNA yang di mana baris MRZ setiap negara berbeda-beda.

Pada penelitian kali ini juga melakukan pencocokan wajah antara yang ada pada paspor dan juga foto real guna memvalidasi kepemilikan asli paspor. Biometrik menjadi umum karena otomatisasi, pendekatan yang mudah, dan tingkat akurasi yang terus meningkat, sesuai dengan tuntutan keamanan yang semakin penting dalam kehidupan sehari-hari. Salah satu teknologi yang digunakan untuk tujuan keamanan adalah pengenalan wajah. Pengenalan wajah merupakan teknik biometrik yang populer. Teknologi pengenalan wajah telah mengalami kemajuan yang pesat dalam beberapa tahun terakhir. Dibandingkan dengan metode lain, teknologi ini lebih langsung, mudah digunakan, dan nyaman. Namun, sistem pengenalan wajah rentan terhadap serangan palsu menggunakan wajah palsu. Untuk melindungi sistem dari pemalsuan, deteksi keaktifan diperlukan dalam sistem yang aman (Policepatil & Hatture, 2021). Pada penelitian ini menggunakan *face live-detection* menggunakan sampel wajah dan data wajah yang didapatkan dari proses foto selfie. Hasil liveness menjadi salah satu faktor disetujuinya pengajuan paspor untuk digitalisasi.

Dengan melakukan pengecekan keaslian wajah belum bisa menjadi jawaban apakah paspor yang digunakan asli atau palsu. Dengan melihat penyebab kurangnya hasil penelitian dari sistem pendeteksian oleh wilnotomo yang diakibatkan oleh banyaknya penumpang yang harus dilayani dan proses pemeriksaan keimigrasian

di konter imigrasi berlangsung dengan cepat(Wilonotomo dkk., 2020). Pada penelitian ini melakukan pengecekan bertahap dalam mengecek keaslian paspor pada proses digitalisasi. Data formulir yang telah diisi oleh pengguna akan digunakan untuk proses pengecekan keaslian dengan mengecek data paspor yang ada pada SIMKIM(Sistem Informasi Manajemen Keimigrasian) menggunakan API. SIMKIM sendiri menggunakan *System Development Life Cycle* (SDLC) atau bersifat berkelanjutan dalam bentuk siklus, sehingga Direktorat Sistem dan Teknologi Informasi Keimigrasian diharapkan melakukan evaluasi terus menerus(Cakra & Arisman, 2020). Hasil dari digitalisasi berupa QR Code yang dapat membantu petugas Imigrasi dalam pengecekan paspor hanya menggunakan scan barcode.

4.2 Implementasi

Pada bab ini akan dibahas tentang implementasi rancangan “Pengembangan Scan Paspor Aplikasi Layanan Keimigrasian(Digital Imigrasi) Menggunakan Metode MRZ PARSER dan OCR” yang telah dibuat rancangan sebelumnya pada bab III.

4.2.1 Kebutuhan

Penelitian yang akan dikembangkan memiliki kebutuhan untuk penggunaan yang harus diperhatikan oleh pengguna, sebagai berikut:

- Minimal Android 6.0 (Api Level 23)
- Minimal IOS 13
- Memiliki kamera depan dan belakang yang bisa digunakan

4.2.2 Proses Pengambilan Data Paspor

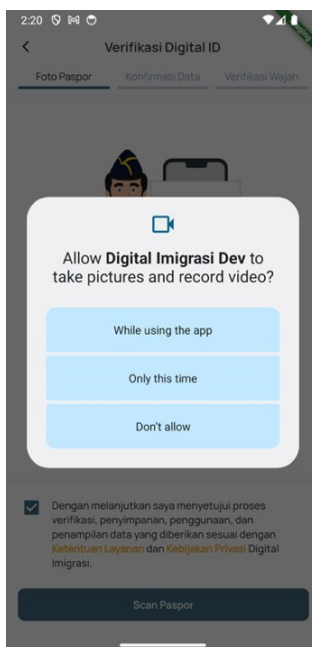
Proses Pengambilan Data Paspor pada penelitian ini menggunakan sistem google yang tersedia di flutter yaitu `google_ml_kit_text_recognition` dengan source code sebagai berikut :

1. Request Camera

```
if (await Permission.camera.request().isGranted) {  
  if (await  
Permission.microphone.request().isGranted) {  
    Get.to(() {  
      return ScalableOCR(passportType:  
passportType);  
    }, binding: ScanBinding());  
  }  
}
```

Segmen Program 4. 1 Request Camera Access

Permintaan kamera diperlukan karena penggunaan OCR menggunakan kamera sebagai alat implementasinya, dengan tampilan seperti berikut.



Gambar 4. 1 Request Camera Access

2. Stream Camera

```
Future startLiveFeed() async {
  _cameras = await availableCameras();
  // scanController.controller =
  //   CameraController(_cameras[0],
  ResolutionPreset.high);
  final camera = _cameras[0];
  scanController.controller = CameraController(
    camera,
    ResolutionPreset.high,
    enableAudio: false,
    imageFormatGroup: (Platform.isAndroid)
      ? ImageFormatGroup.nv21
      : ImageFormatGroup.bgra8888,
  );
  scanController.controller?.initialize().then((_)
  {
    if (!mounted) {
      return;
    }
    scanController.controller?.startImageStream(_proces
    ssCameraImage);
    setState(() {});
  }).catchError((Object e) {
    if (e is CameraException) {
      switch (e.code) {
        case 'CameraAccessDenied':
          log('User denied camera access.');
```

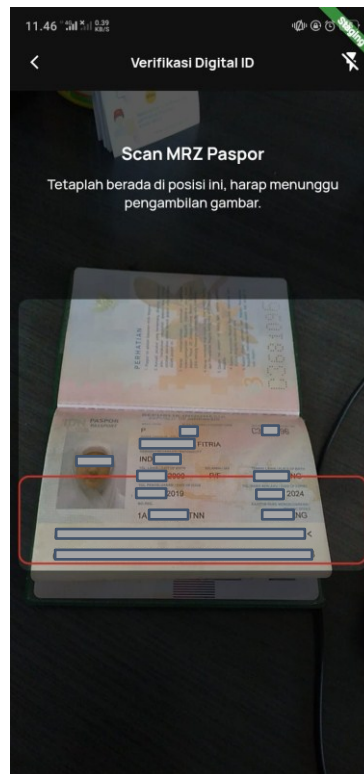
```

    } else {
        DialogService.showDialogProblem(
            imagePath:
"assets/lottie/wongInputMRZ.json",
            title:
Localization.invalidScanPasportMRZTile.tr,
            description:
Localization.invalidScanPasportMRZDescription.tr,
            textButton: Localization.close.tr,
            buttonOnTap: () async {
                var scanController =
Get.find<ScanController>();
                await
scanController.controller?.stopImageStream();
                await
scanController.controller?.dispose();
                scanController.controller = null;
                scanController.update();
                AnyUtils.getBack();
            },
        );
    }
});
}

```

Segmen Program 4.2 Start Stream Camera

Berikut adalah fungsi untuk melakukan stream camera yang digunakan untuk proses pengambilan gambar secara kontinu. Pada stream camera memiliki parameter berupa fungsi apa yang dilakukan saat melakukan stream camera yang mengembalikan value berupa Camera Image. Hasil dari proses stream camera nantinya akan di proses untuk menyesuaikan tipe data yang digunakan untuk melakukan image prosesing.



Gambar 4. 2 Stream Camera

3. Proses Perubahan Tipe Data

Pada Proses ini masuk pada proses yang detail untuk perubahan suatu tipe data hasil stream camera sebelumnya

```
Future _processCameraImage(CameraImage image)
async {
    final WriteBuffer allBytes = WriteBuffer();
    for (final Plane plane in image.planes) {
        allBytes.putUint8List(plane.bytes);
    }
    final bytes =
    allBytes.done().buffer.asUint8List();
    final Size imageSize =
        Size(image.width.toDouble(),
            image.height.toDouble());
    final camera = _cameras[0];
```

```

        final imageRotation =
InputImageRotationValue.fromRawValue(camera.sensor
Orientation);
if (imageRotation == null) return;
        final inputImageFormat =
InputImageFormatValue.fromRawValue(image.format.ra
w);
        if (inputImageFormat == null) return;
// if (image.planes.length != 1) return null;
        final plane = image.planes.first;
        final inputImageData = InputImageMetadata(
            size: imageSize,
            rotation: imageRotation,
            format: inputImageFormat,
            bytesPerRow: plane.bytesPerRow,
        );
        final inputImage =
            InputImage.fromBytes(bytes: bytes,
metadata: inputImageData);
// Karena hanya bisa membaca input image
        processImage(inputImage, image, bytes);
    }

```

Segmen Program 4. 3 Proses Convert Camera Image to Input Image

Proses ini dilakukan karena *text-recognition* hanya bisa membaca gambar dengan tipe input image maka proses convert harus dilakukan. Proses ini yang menjadi pembeda dari sistem ocr saat ini, dikarenakan proses dilakukan secara kontinu.

4. Text Recognizer Image

```
final recognizedText = await
_textRecognizer.processImage(inputImage);

    if (inputImage.metadata?.size != null &&
        inputImage.metadata?.rotation != null &&
        cameraPrev.currentContext != null) {
        final RenderBox renderBox =

cameraPrev.currentContext?.findRenderObject() as
RenderBox;

        log(scanController.listMRZ.length.toString(),
name: "MRZ LENGTH");
```

Segmen Program 4. 4 Fungsi Teks Recognizer

Pada segmen ini merupakan fungsi untuk menggunakan teks recognizer dari package `google_ml_kit_text_recognition` yang menghasilkan data berupa recognized text. Hasil yang dapat dikelola untuk pengambilan suatu baris teks.

5. Pemberian Tanda Data Terambil

```

        for (final textBunk in
recognizedText.blocks) {
            for (final element in textBunk.lines) {
                // 44 karena pengecekan minimal MRZ 44
length
                if (element.text.replaceAll(' ',
'').length >= 44) {
                    // Kondisi MRZ terpenuhi
                    scanController.mrzDescription.value =
Localization.descriptionScanPassportMRZDetected.tr;
                } else {
                    scanController.mrzDescription.value =
Localization.descriptionScanPassportMRZ.tr;
                }
            }
        }
    }

```

Segmen Program 4. 5 Fungsi Pengambilan Baris MRZ

Pada segmen ini fungsi pengambilan baris MRZ dengan melakukan pengecekan menggunakan perulangan FOR-LOOP untuk memenuhi kondisi sebagai berikut :

1. MRZ berada dalam blok teks recognizer
2. MRZ berada dalam sebuah garis
3. MRZ memiliki elemen karakter berjumlah minimal 44 karakter



Gambar 4. 3 Text Recognizer dan Tanda Terambil

4.2.3 Penerapan MRZ Parser

Setelah penerapan OCR didapatkan hasil teks MRZ yang digunakan untuk mengubah ke sebuah data model, proses penguraian dilakukan sebagai berikut.

```

/// Pengambilan Country Code
    final checkEd = input[0].substring(2,
5).getCountryCodePassport;

///Pengecekan Bagian Satu atau dua
    final firstLine = checkEd != null ? input[0] :
input[1];
    final secondLine = checkEd == null ? input[0] :
input[1];

    /// Pengecekan apakah paspor atau bukan
    final isPasporDokument = firstLine[0] == 'P';

```

Segmen Program 4. 6 Pengecekan baris pertama atau kedua MRZ

Pada segmen ini dilakukan pengecekan terlebih dahulu antara baris pertama dan kedua MRZ, perbedaan dibedakan dengan tidak ada kode negara pada bagian kedua. Setelah itu dilakukan pengecekan tipe dokumen apakah paspor atau bukan dengan dilihat baris pertama indeks pertama MRZ.

```
/// Pengambilan baris Nama
    final namesRaw = firstLine.substring(5);
/// Pengambilan nomor dokumen
    final documentNumberRaw =
secondLine.substring(0, 9);
/// Pengambilan kewarganegaraan
    final nationalityRaw =
secondLine.substring(10, 13);
/// Pengambilan Tanggal Lahir
    final birthDateRaw = secondLine.substring(13,
19);
/// Pengecekan Digit Tanggal Lahir
    final birthDateCheckDigitRaw =
secondLine[19];
/// Pengambilan Jenis Kelamin
    final sexRaw = secondLine.substring(20, 21);
/// Pengambilan Tanggal Berakhir
    final expiryDateRaw =
secondLine.substring(21, 27);
/// Pengecekan Digit Tanggal Berakhir
    final expiryDateCheckDigitRaw =
secondLine[27];
```

Segmen Program 4. 7 Pengambilan setiap data berdasarkan indeks MRZ

Pada segmen ini dilakukan pengambilan data sesuai digit indeks yang ada pada baris MRZ. Fungsi substring pada flutter digunakan untuk mengambil bagian sesuai indeks yang ditentukan, substring menyediakan dua parameter, di mana indeks pertama yang wajib diisi dan indeks akhir yang ingin ditentukan yang bersifat opsional.

```

extension _MRZStringExtensions on String {
  static final _validInput = RegExp(r'^[A-Z|0-9|<]+\$');

  bool get isValidMRZInput =>
    _validInput.hasMatch(this);

  String trimChar(String char) {
    if (isEmpty || char.isEmpty) {
      return this;
    }
    var start = 0, end = length - 1;
    while (start < length && this[start] == char) {
      start++;
    }
    while (end >= 0 && this[end] == char) {
      end--;
    }
    return start < end ? substring(start, end + 1)
: '';
  }

  String replaceSimilarDigitsWithLetters() =>
    replaceAll('0', 'O')
      .replaceAll('1', 'I')
      .replaceAll('2', 'Z')
      .replaceAll('8', 'B');

  String replaceSimilarLettersWithDigits() =>
    replaceAll('O', '0')

```

```
.replaceAll('Q', '0')
.replaceAll('U', '0')
.replaceAll('D', '0')
.replaceAll('I', '1')
.replaceAll('Z', '2')
.replaceAll('A', '4')
.replaceAll('S', '5')
.replaceAll('T', '7')
.replaceAll('B', '8')
.replaceAll('G', '9');

String replaceAngleBracketsWithSpaces() =>
replaceAll('<', ' ');
}
```

Segmen Program 4. 8 Pengecekan Hasil OCR

Pada segmen ini terdapat ekstension yang digunakan untuk pengecekan hasil OCR, dikarenakan hasil OCR memiliki kekurangan yang terkadang teks recognizer tidak dapat membedakan huruf atau angka yang memiliki bentuk yang hampir sama seperti 1 dan I, 2 dan Z, 8 dan B, dan lainnya. Penggunaan beberapa ekstension digunakan sesuai kebutuhan seperti nama yang tidak bisa memiliki digit angkat atau tanggal yang tidak bisa memiliki huruf pada format MRZ.

```
class MRZFieldParser {
    MRZFieldParser._();

    static String parseDocumentNumber(String input)
=> _trim(input);

    static String parseDocumentType(String input) =>
_trim(input);

    static String parseCountryCode(String input) =>
_trim(input);

    static String parseNationality(String input) =>
_trim(input);

    static String parseOptionalData(String input) =>
_trim(input);

    static List<String> parseNames(String input) {
        final words = input.trimChar('<').split('<<');
        final result = [
            words.isNotEmpty ? _trim(words[0]) : '',
            words.length > 1 ? _trim(words[1]) : '',
        ];
        return result;
    }

    static DateTime parseBirthDate(String input) {
        final formattedInput = _formatDate(input);
        return _parseDate(formattedInput,
DateTime.now().year - 2000);
    }

    static DateTime parseExpiryDate(String input) {
        final formattedInput = _formatDate(input);
        return _parseDate(formattedInput, 70);
    }

    static Sex parseSex(String input) {
        switch (input) {
            case 'M':
                return Sex.male;
            case 'F':
                return Sex.female;
        }
    }
}
```

```

        default:
            return Sex.none;
        }
    }

    static String _formatDate(String input) =>
    _trim(input);

    static DateTime _parseDate(String input, int
    milestoneYear) {
        final parsedYear = int.parse(input.substring(0,
    2));
        final centennial = (parsedYear > milestoneYear)
    ? '19' : '20';
        return DateTime.parse(centennial + input);
    }

    static String _trim(String input) =>

```

Segmen Program 4. 9 Proses Parsing data ke bentuk form

Segmen ini berfungsi menguraikan dari bentuk yang didapat menjadi format data yang dapat dibaca untuk ditampilkan pada form pengajuan. Pada class ini terdapat fungsi trim, fungsi trim di flutter digunakan untuk menghilangkan spasi di depan atau di belakang kata yang didapat. Pada class ini juga terdapat cara menguraikan sebuah tanggal yang ada pada baris MRZ, proses penguraian dilakukan karena pada format yang didapat tidak lengkap seperti contohnya tahun kelahiran 2000 pada baris paspor tertulis 00. Proses penguraian dilakukan dengan melihat apabila dua digit akhir taun lebih besar dari dua digit tahun ini dikurangi

2000 maka termasuk pada kelahiran 1900-an dan termasuk kelahiran 2000-an apabila lebih kecil.

```

Map<String, String> getCountryCodePassport(
    int start, int end ) {
    var rawCountryCode = substring(start, end);
    Map<String, String> countryCodeResult = {};
    /// Pengecekan semua coutry code negara
    for (var countryCode in listCountryCode) {
        if
        (rawCountryCode.contains(countryCode['alpha_3_code'
] as String)) {
            countryCodeResult = countryCode;
        }
    }
    return countryCodeResult;
}
/// Contoh List Country Code
{
    "num_code": "360",
    "alpha_2_code": "ID",
    "alpha_3_code": "IDN",
    "en_short_name": "Indonesia",
    "nationality": "Indonesia"
},

```

Segmen Program 4. 10 Fungsi Pengambilan Kode Negara

Pada segmen ini cara bagaimana pengambilan kode negara untuk mengetahui asal negara pada paspor yang dilakukan pengambilan data. Proses yang dilakukan yaitu pengecekan data pada daftar data yang tersedia dan diambil *alpha_3_code* sebuah negara yang sesuai.

4.2.4 Penerapan Face Detection

Setelah melakukan proses MRZ Parser, Penerapan Face Detection dilakukan untuk pencocokan wajah pada paspor asli dan real. Proses deteksi wajah pertama dilakukan untuk mengambil sampel data yang digunakan untuk pencocokan. Proses deteksi wajah dilakukan dengan source code sebagai berikut.

```
Color selection = Colors.white;
/// Penggunaan Package google Ml kit face detection
final faces = await
_faceDetector.processImage(inputImage);
if (scanController.loopFace.value >= 1) {
selection = Colors.green;
scanController.mrzDescription.value =
Localization.descriptionScanPassportFaceDetected.tr
;
} else {
selection = Colors.red;
scanController.mrzDescription.value =
Localization.descriptionScanPassportFace.tr;
}
var painter = FaceDetectorPainter(
faces, inputImage.metadata!.size,
inputImage.metadata!.rotation,
renderBox,
color: selection );
customPaint = CustomPaint(painter: painter);
/// Kondisi Benar value >= 4
```

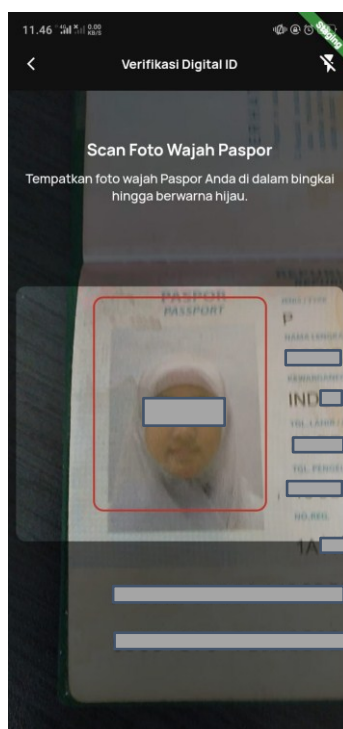
```

if (scanController.loopFace.value >= 4) {
    var cropFullPassport;
    var cropFacePassport;
    try {
    /// Pengambilan Full Passport untuk BE
        cropFullPassport = _cropPassportImageFile(
            Uint8List.fromList(
imglib.encodePng(scanController.imgPassportFull!.value),
            ),
        );
    /// Hasil wajah untuk pasport tampilan
        cropFacePassport = _cropFaceImageFile(
            Uint8List.fromList(imglib.encodePng(
                imglib.copyRotate((Platform.isAndroid)?
convertNV21ToImage(inputImage, bytes)
                :
convertBGRA8888ToImage(image),
                (Platform.isAndroid) ? 90 :
0),
            ),
            ),
            faces[0],);
    } catch (e) {
        throw Exception('Face is empty');
    }
    redirectResultScan(
        cropFullPassport,
        facePassport: cropFacePassport,
    );
}

```

Segmen Program 4. 11 Proses Pengambilan Sampel Data Face Detection

Pada segmen ini dilakukan pengambilan sampel deteksi wajah yang dilakukan dengan menjalankan fungsi yang ada pada `google_ml_kit_face_detection` yaitu `processImage`. Data yang didapatkan pada deteksi wajah ini digunakan untuk sampel pencocokan wajah pada saat liveness nantinya. Proses ini sangat perlu dilakukan karena untuk proses validasi kepemilikan paspor.



Gambar 4. 4 Deteksi Wajah

4.2.5 Validasi Frame

Validasi Frame digunakan untuk memudahkan pengguna dalam pengambilan data paspor, dengan source code sebagai berikut:

```
var painter = TextRecognizerPainter(  
    inputImage,  
    bytes,  
    image,  
    recognizedText,  
    inputImage.metadata!.size,  
    inputImage.metadata!.rotation,  
    renderBox,  
    (value) {},  
    getRawData: (value) {  
        if (widget.getRawData != null) {  
            widget.getRawData!(value);  
        }  
    },  
    paintboxCustom: widget.paintboxCustom,  
);  
customPaint = CustomPaint(painter:  
painter);
```

Segmen Program 4. 12 Frame Validation

Segmen ini digunakan untuk pewarnaan bingkai pada pengambilan data paspor. Apabila kondisi pada segmen sebelumnya terpenuhi bingkai akan berubah menjadi warna hijau dan saat belum terpenuhi akan berwarna merah.

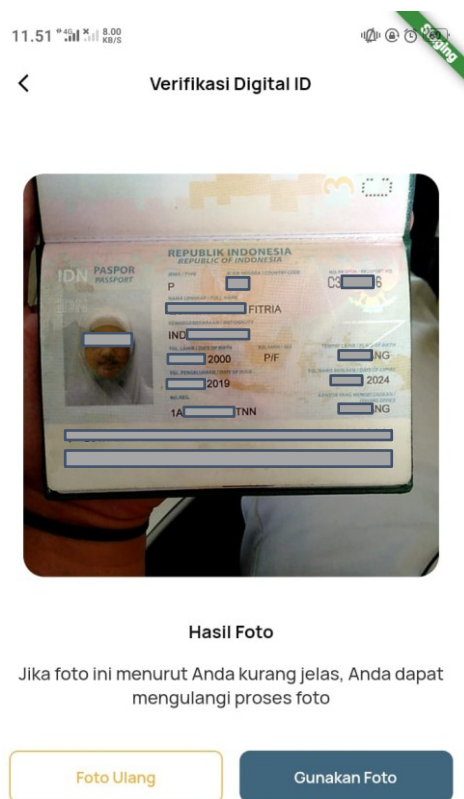
```

void _redirectResultScan(
    imglib.Image fullPassport, {
    imglib.Image? facePassport,
}) async {
    await _stopLiveFeed();
    Get.offUntil(
        GetPageRoute(
            settings: RouteSettings(name: '/result-scan-
view'),
            binding: ResultScanBinding(),
            page: () {
                if (widget.passportType ==
PassportTypes.REGULER) {
                    return ResultScanView(
                        passportType: widget.passportType,
                        passportFullImage: fullPassport,
                        imgPassport: facePassport,
                        mrzResult:
ScanController.to.mrzResult!.value,
                    );
                } else {
                    return ResultScanView(
                        passportType: widget.passportType,
                        passportFullImage: fullPassport,
                        mrzResult:
ScanController.to.mrzResult!.value,
                    );
                }
            },
        ),
        (route) => route.settings.name ==
"/ocrGuideView",
    );
}

```

Segmen Program 4. 13 Redirect Hasil Pengambilan Data Paspor

Segmen ini merupakan segmen terakhir untuk proses pengambilan data paspor. Pada segmen ini merupakan validasi hasil foto yang digunakan untuk proses validasi dari admin. Setelah melakukan validasi hasil foto nantinya akan diarahkan pada halaman form validasi.



11.51 8.00 KB/S

< Verifikasi Digital ID

REPUBLIC INDONESIA
REPUBLIC OF INDONESIA

IDN PASPOR
PASSPORT

P [] C: [] 6

NIK [] FITRIA

IND []

1A [] TNN

2000 P/J [] NG

2019 [] 2024

[] NG

Hasil Foto

Jika foto ini menurut Anda kurang jelas, Anda dapat mengulangi proses foto

Foto Ulang

Gunakan Foto

Gambar 4. 5 Validasi Hasil Foto Data Paspor

4.2.6 Validasi Form Pengajuan

Validasi Form pengajuan digunakan untuk memvalidasi hasil yang didapatkan dari pengambilan data paspor sebelumnya dan mengisi data yang tidak didapatkan dari baris MRZ.

```
/// Daftar yang harus terpenuhi
var listRequest = [
    {
        "type":
        UploadPhotoTypes.getDigitalIdPassportFullType,
        "image": pasporC.data.value.ktpPhoto, //
        PASSPORT IMAGE FULL
    },
    {
        "type":
        UploadPhotoTypes.getDigitalIdPassportCropType,
        "image": pasporC.data.value.pictCropping, //
        PASSPORT IMAGE CROP
    },
];
/// Direct Halaman
if (uploadedCount == listRequest.length) {
    Get.toNamed(
        AppRoutes.formPassportView,
        arguments: FormPassportArgument.reguler(),
    );
} else {
    DialogService.showDialogProblem(
        description: "Terjadi kesalahan pada saat
        upload image",
```

Segmen Program 4. 14 Halaman Validasi Form Pengajuan

Pada segmen ini yaitu pengarahan setelah memenuhi ketentuan sudah melakukan pengambilan data gambar untuk data foto paspor dan juga pengambilan satu halaman data paspor. Data foto satu halaman paspor dibutuhkan untuk pengecekan admin dan untuk foto paspor digunakan untuk tampilan dari digital paspor.

11.53 4G 0.00

< Verifikasi Digital ID

Foto Paspor Konfirmasi Data Verifikasi Wajah

00R

Tempat Lahir

NG

Tanggal Lahir

2000

Jenis Kelamin

Laki-laki Perempuan

Kewarganegaraan

IND

Tanggal Pengeluaran

2020

Tanggal Habis Berlaku

2024

Nomor Registrasi

1A1

Lanjutkan

Gambar 4. 6 Form Validasi

4.2.7 Validasi Keaslian Paspor

Validasi keaslian paspor dilakukan untuk menjadi validasi pertama dalam pembuatan data paspor dengan melakukan pengecekan melalui API dengan mengirim semua parameter yang diperlukan.

```
Future<void> checkSimkim() async {  
    try {  
        var expiredDate =  
DateTime.parse(data.value.expired ?? "");  
        expiredDate.add(const Duration(days: 1));  
        var date = DateFormat('yyyy-MM-  
dd').format(expiredDate.add(const Duration(days:  
1)));  
        DialogService.showLoading();  
        var body = {  
            "no_paspor": data.value.docNo,  
            "jenis_paspor": data.value.passportType,  
            "kode_negara": data.value.nationalityCode,  
            "nama_lengkap": data.value.name,  
            "warga_negara": data.value.nationality,  
            "tanggal_lahir": data.value.doB,  
            "tgl_pengeluaran": data.value.issuerDate,  
            "reg_no": data.value.registerNo,  
            "kelamin": data.value.gender == "female" ?  
"P" : "L",  
            "tempat_lahir": data.value.poB,  
            "tgl_habis_berlaku": date,  
            "nama_kanim_pengeluaran": data.value.issuer  
        };  
        final response = await  
DigitalIdRepositories.checkSimkim(body: body);  
        if (response) {  
            statusPaspor = true;  
        }  
    }  
}
```

```
    } else {  
        statusPaspor = false;  
    }  
    update();  
    DialogService.closeLoading();  
  
    statusPaspor = false;  
}
```

Segmen Program 4. 15 Validasi Keaslian Paspor

Pada segmen ini dilakukan pengecekan keaslian paspor dengan mengirim semua parameter seperti nomor paspor, jenis paspor, kode negara, nama lengkap, warga negara, tanggal lahir, tanggal pengeluan, nomor register, jenis kelamin, tempat lahir, tanggal habis berlaku, dan nama kanim pengeluan dengan metode *post*. Sebelum melakukan parameter dilakukan penambahan satu hari untuk tanggal kedaluwarsa dari paspor, dikarenakan pengecekan untuk tanggal kedaluwarsa tercatat kedaluwarsa apabila melebihi batas waktu kedaluwarsa pada paspor. Perubahan data ini hanya dilakukan untuk pengecekan keaslian, data yang ditampilkan pada paspor digital tetap data asli yang dihasilkan dari validasi form pengajuan. Hasil yang didapatkan dari proses ini yaitu berbentuk *boolean*, apabila data yang dikirimkan sesuai akan memberikan respon *true* dan apabila tidak sesuai akan memberikan respon *false*.

4.2.8 Liveness

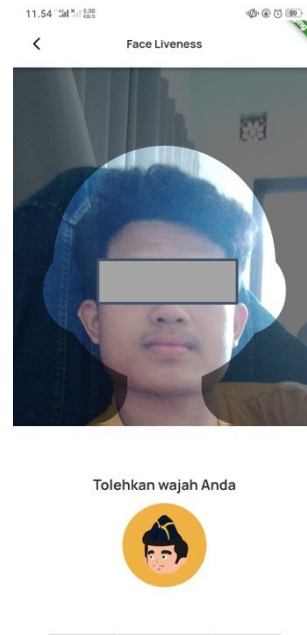
Liveness digunakan untuk pengambilan foto asli dengan real-time, hasil foto liveness akan dilakukan pencocokan dengan hasil data sampel foto pada proses deteksi wajah sebelumnya. Proses liveness merupakan proses validasi kedua yang

dilakukan untuk memvalidasi kebenaran kepemilikan paspor yang diajukan dengan yang mengajukan.



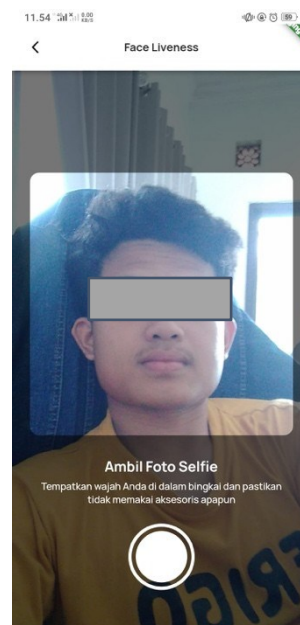
Gambar 4. 7 Tata Cara Liveness

Tampilan awal sebelum liveness, memperlihatkan bagaimana penggunaan verifikasi wajah yang akan dilakukan.



Gambar 4. 8 Cek Bukan Bot

Tampilan awal liveness yaitu pengecekan bukan bot dengan memberikan perintah gerakan seperti tolehkan wajah anda, buka mulut anda, dan kedipkan mata anda. Sistem akan mendeteksi dengan hasil peragaan yang dilakukan pengguna.



Gambar 4. 9 Pengambilan Foto

Tampilan kedua liveness, pengambilan foto untuk data pencocokan wajah.



Gambar 4. 10 Validasi Hasil Foto

Tampilan validasi hasil foto, validasi hasil foto ini digunakan untuk pengecekan hasil foto yang akan digunakan untuk pencocokan wajah apabila merasa kurang bisa melakukan foto ulang, namun jika dirasa sudah sesuai maka bisa gunakan foto yang nantinya akan menjalankan proses pencocokan wajah.

```
static Future<String> executeFaceMatching({
  required File image1,
  required File image2,
  /// ResCropped hasil foto wajahnya saja
  required Function(File? resCroppedFace1, File?
resCroppedFace2) onMatching,
  /// ResCropped hasil foto wajahnya saja
  required Function(File? resCroppedFace1, File?
resCroppedFace2)
  onNotMatching,
```

```

        required Function(File? resCroppedFace1, File?
resCroppedFace2) onNotMatching,

        required Function() onNotSupport,

        required Function(
            File? resCroppedFace1,
            File? resCroppedFace2,) onNotFoundFace,
    }) async {
        bool isCanRunTfLite = await _initializeTfLite();
        /// * Ketika tidak support TF Lite
        if (!isCanRunTfLite) {
            onNotSupport();
            return 'not-support'; }

        /// * Detect Face dari masing2 image
        var detectedFaceFile1 = await
detectFacesFromImageFile(
            source: image1,);

        var detectedFaceFile2 = await
detectFacesFromImageFile(
            source: image2,);

        /// * Detect Recognition Data dari tiap image
        List recognitionPredictedData1 = [];
        List recognitionPredictedData2 = [];

        /// * File 1
        if (detectedFaceFile1.isNotEmpty) {
            recognitionPredictedData1 =
_getRecognitionPredictionImageFile(
                fileImageValue: image1,
                faceDetectedValue: detectedFaceFile1.first,
            );

```

```
/// * File 2
if (detectedFaceFile2.isNotEmpty) {
    recognitionPredictedData2 =
_getRecognitionPredictionImageFile(
    fileImageValue: image2,
    faceDetectedValue: detectedFaceFile2.first,
    );
}
/// * Crop Face dari image (ini bentuk wajah doang)
File? resFace1;
File? resFace2;
var resultListCropFace1 = await
cropFacesFromImage(file: image1,
    );
var resultListCropFace2 = await
cropFacesFromImage(file: image2,
    );
if (resultListCropFace1.isNotEmpty) {
    resFace1 = resultListCropFace1.first;
}
if (resultListCropFace2.isNotEmpty) {
    resFace2 = resultListCropFace2.first;
}
/// * Ketika salah satu tidak terdeteksi wajahnya
if (detectedFaceFile1.isEmpty ||
detectedFaceFile2.isEmpty) {
    onNotFoundFace(resFace1, resFace2);
    return 'not-found-face';
}
var waw = await
detectFacesFromImageFileFace(source: image2);
var wiw = await
detectFacesFromImageFileFace(source: image1);
```



```

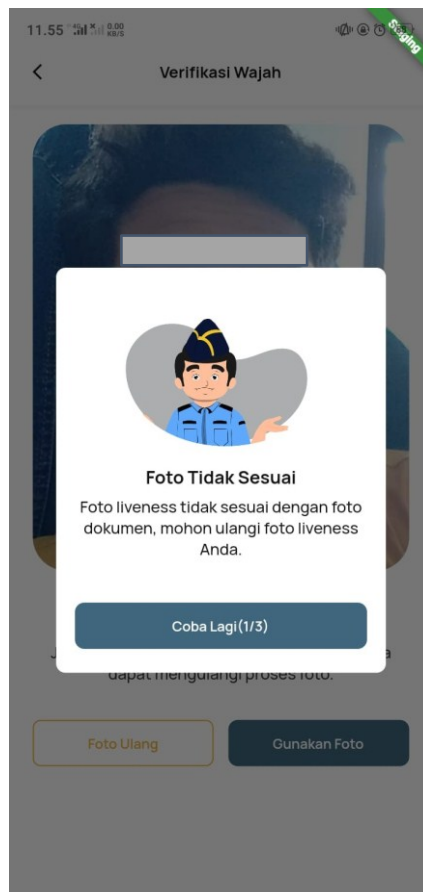
// Variables
double minimumDistanceCalculation = 0;
double maximumDistanceCalculation = 1;
double resultDistanceCalculation = 0.0;
late String accurationPercentage;
/// * Menghitung Akurasi Face Matching nya
resultDistanceCalculation =
_euclideanDistance(wiw, waw);
if (resultDistanceCalculation >=
minimumDistanceCalculation &&
resultDistanceCalculation <
maximumDistanceCalculation) {
if (resultDistanceCalculation <= 0.5) {
accurationPercentage = '100';
} else {
var percentage =
(maximumDistanceCalculation -
resultDistanceCalculation) * 100;
accurationPercentage =
(percentange <= 0) ? "0" :
(percentange.round()).toString();
onMatching(resFace1, resFace2);
return 'match';
} else {
onNotMatching(resFace1, resFace2);
return 'not-match';
}
}
}

```

Segmen Program 4. 16 Proses Pencocokan Wajah

Pada segmen ini yaitu proses yang dilakukan setelah data liveness didapatkan. Yang dilakukan pertama pada proses ini yaitu deteksi wajah data sampel yaitu image1 dan data wajah liveness yaitu image2. Pada proses ini akan

mengembalikan nilai matching apabila wajah yang terdeteksi sesuai seperti data sampel yang didapatkan dan mengembalikan nilai not-matching apabila wajah yang dideteksi tidak sesuai atau berbeda.



Gambar 4. 11 Hasil Negatif Pencocokan Wajah

Hasil pencocokan wajah akan muncul sebagai berikut apabila wajah pada data paspor dan wajah pada proses liveness tadi berbeda. Pengguna akan diberikan tiga kali percobaan untuk melakukan proses pencocokan wajah. Namun apabila wajah pada data paspor dan proses liveness sama akan diarahkan pada halaman selanjutnya.

4.2.9 Tampilan Kartu Paspor Digital

Tampilan kartu paspor yaitu tampilan digital yang dihasilkan setelah proses pengajuan digitalisasi paspor.

```
data3!.docNationality == "WNA"?  
  PassportWnaCard(  
    sensor: false,  
    data: data3,  
    height: height < 1043 ? 269.h : 259.h,  
    detail: false,  
  )  
  : PassportCard(  
    sensor: false,  
    data: data3,  
    height: height < 1043 ? 269.h : 259.h,  
    detail: false,  
  ),
```

Segmen Program 4. 17 Tampilan Kartu Paspor Digital

Pada segmen ini yaitu segmen terakhir dalam proses pengajuan digitalisasi paspor. Tampilan kartu paspor digital memiliki perbedaan apabila diterima atau sesuai kartu akan memiliki label “Terverifikasi” dan apabila tidak diterima pada bagian bawah kartu akan terdapat catatan alasan kenapa tidak diterima. Tampilan untuk paspor WNI dan WNA dibedakan karena ada perbedaan format tampilan pada setiap negara.



Gambar 4. 12 Tampilan Paspur Digital

Tampilan hasil paspor digital, hasil paspor memiliki tiga kondisi yang menentukan tampilan di bawahnya,

1. Apabila hasil foto live dengan foto dokumen sesuai dan dokumen yang diajukan juga sesuai akan menampilkan barcode paspor digital yang telah teverifikasi.
2. Apabila hasil foto live dengan foto pada dokumen sesuai namun dokumen yang diajukan tidak sesuai akan menampilkan hasil “Dokumen Paspur yang Diajukan Tidak Sesuai, Digital ID Dalam Proses Pemeriksaan Admin”
3. Apabila hasil foto live dengan foto pada dokumen tidak sesuai namun dokumen yang diajukan sesuai akan menampilkan hasil

“Foto Live Dan Foto Dokumen Berbeda, Digital ID Dalam Proses Pemeriksaan Admin”

4. Apabila hasil foto live dengan foto pada dokumen tidak sesuai dan dokumen yang diajukan tidak sesuai maka akan menampilkan hasil “Foto Live dan Dokumen Yang Diajukan Tidak Sesuai, Digital ID Dalam Proses Pemeriksaan Admin”.

4.3 Uji Coba

Uji coba pada penelitian ini dilakukan menggunakan dua metode, yaitu black-box dan pengujian semua fitur. Pengujian semua fitur dilakukan oleh 20 orang dengan melakukan testing pengambilan data paspor dengan sistem pemindaian yang baru.

4.3.1 Pengujian Black Box

Pengujian black-box adalah pengujian yang digunakan untuk menguji kebutuhan fungsionalitas sistem dan berfokus pada bagian luar sistem. Pengujian ini dilakukan menggunakan serangkaian input yang dapat dimasukkan melalui tampilan antarmuka untuk memastikan kesesuaian output dari sistem. Pengujian black-box dilakukan karena uji coba black-box lebih efektif daripada uji coba white-box ketika frekuensi uji coba lebih rendah (Wintana dkk., 2022), dan pada uji coba pada penelitian ini memiliki frekuensi uji coba yang tidak terlalu tinggi.

4.3.1.1 Pengujian Pengambilan Data Paspor

Tabel 4. 1 Pengambilan MRZ Paspor

Pengambilan MRZ Paspor	
Input	-
Prosedur	<ol style="list-style-type: none"> 1. Aktor menekan tombol “scan paspor” 2. Aktor memberikan izin penggunaan kamera

	3. Aktor menepatkan baris MRZ dikotak pemindaian
	4. Aktor menunggu hingga bingkai pemindaian berubah menjadi warna hijau
	5. Aktor menunggu hingga kotak pemindaian berubah menjadi kotak pemindaian wajah
Hasil yang diharapkan	Baris MRZ berhasil didapatkan
Hasil Akhir Status	Baris MRZ berhasil didapatkan valid

4.3.1.2 Pengujian Pengambilan Foto Wajah Paspor

Tabel 4. 2 Pengambilan Foto Wajah Paspor

Pengambilan Foto Wajah Paspor	
Input	-
Prosedur	<ol style="list-style-type: none"> 1. Aktor menepatkan wajah paspor pada kotak pemindaian wajah 2. Aktor menunggu hingga frame kota pemindaian wajah berubah menjadi warna hijau
Hasil yang diharapkan	Foto wajah berhasil didapatkan
Hasil Akhir Status	Foto wajah berhasil didapatkan valid

4.3.1.3 Pengujian Parsing MRZ

Tabel 4. 3 Parsing MRZ

Parsing MRZ	
Input	-
Prosedur	1. Aktor menekan tombol “Gunakan Foto”
Hasil yang diharapkan	Data parsing berhasil diparsing ke data model
Hasil Akhir Status	Data parsing berhasil diparsing ke data model Valid

4.3.1.4 Pengujian Pengisian Form

Tabel 4. 4 Pengisian Form

Pengisian Form	
Input	- Aktor mengisi data form yang tidak didapatkan atau tidak sesuai waktu proses parsing
Prosedur	<ol style="list-style-type: none"> 1. Aktor mengecek hasil parsing apakah sesuai 2. Aktor memperbaiki input apabila parsing tidak sesuai 3. Akor mengisi data form yang belum terisi 4. Aktor menekan tombol “Lanjutkan”
Hasil yang diharapkan	Data Form terisi semua
Hasil Akhir Status	Data Form terisi semua Valid

4.3.1.5 Pengujian Pengambilan Foto Diri

Tabel 4. 5 Pengambilan Foto Diri

Pengambilan Foto Diri	
Input	-
Prosedur	<ol style="list-style-type: none"> 1. Aktor memperhatikan tata cara verifikasi wajah 2. Aktor menekan tombol “Ambil Foto” 3. Aktor mengikuti gerakan sesuai panduan yang muncul 4. Aktor menekan tombol lingkaran untuk mengambil foto wajah 5. Aktor menekan tombol “Gunakan Foto”
Hasil yang diharapkan	Gambar Terverifikasi
Hasil Akhir	Gambar Terverifikasi
Status	Valid

4.3.1.6 Pengujian Pengajuan Hasil Digitalisasi

Tabel 4. 6 Pengajuan Hasil Digitalisasi

Pengajuan Hasil Digitalisasi	
Input	-
Prosedur	<ol style="list-style-type: none"> 1. Aktor menekan tombol “Gunakan Foto” 2. Sistem menampilkan hasil digitalisasi
Hasil yang diharapkan	Hasil Digitalisasi berhasil ditampilkan
Hasil Akhir	Hasil Digitalisasi berhasil ditampilkan
Status	Valid

4.3.2 Pengujian Produk

Bagian ini menjelaskan tentang uji coba yang dilakukan terhadap beberapa paspor baik paspor WNA maupun WNI. Berikut data pengujian pengambilan data paspor dari beberapa paspor negara.

Tabel 4. 7 Pengujian Deteksi Baris MRZ Paspor

No	Nama	Jenis Kelamin	Posisi	Kualitas Foto	Jenis	Hasil
1.	EM	Laki-laki	Datar	Tertutup	USA	Berhasil dengan lancar
2.	AM	Laki-laki	Datar	Terbuka	Australia	Berhasil dengan lancar
3.	FA	Laki-laki	Datar	Tertutup	Korea	Berhasil dengan lancar
4.	RF	Laki-laki	Datar	Tertutup	Bangladesh	Berhasil dengan lancar
5.	YF	Laki-laki	Miring	Terbuka	Indonesia	Berhasil dengan lancar
6.	PRL	Perempuan	Miring	Terbuka	Indonesia	Berhasil dengan lancar
7.	DAG	Laki-laki	Datar	Terbuka	Indonesia	Berhasil dengan lancar
8.	AFM	Laki-laki	Datar	Tertutup	Korea	Berhasil dengan lancar
9.	NTH	Perempuan	Datar	Tertutup	Jerman	Berhasil dengan lancar

10.	RC	Laki-laki	Datar	Tertutup	Swiss	Tidak terdeteksi sebentar karena terlalu banyak bergerak, setelah melakukan ulang berhasil dengan mudah.
11.	MAK	Laki-laki	Datar	Terbuka	China	Berhasil dengan lancar
12.	DA	Laki-laki	Datar	Tertutup	Bangladesh	Tidak terdeteksi sebentar karena terlalu banyak bergerak, setelah melakukan ulang berhasil dengan mudah.
13.	AFS	Laki-laki	Datar	Terbuka	Malaysia	Berhasil dengan lancar
14.	QIF	Perempuan	Miring	Terbuka	Indonesia	Berhasil dengan lancar
15.	NL	Perempuan	Miring	Terbuka	Jepang	Berhasil dengan lancar
16.	BF	Laki-laki	Datar	Tertutup	Singapura	Berhasil dengan lancar
17.	MAY	Laki-laki	Datar	Terbuka	Turki	Berhasil dengan lancar
18.	MEH	Laki-laki	Datar	Terbuka	Belanda	Berhasil dengan lancar
19.	AS	Laki-laki	Miring	Tertutup	Kanada	Berhasil dengan lancar
20.	AED	Laki-laki	Miring	Terbuka	Inggris	Berhasil dengan lancar

Hasil dari pengujian deteksi MRZ paspor dari 20 orang dengan beberapa posisi paspor dan kualitas foto profil tersebut mendapatkan hasil yang positif. Data yang diuji menghasilkan 18 data dengan status “Berhasil dengan lancar”, 2 data tidak lancar namun terdeteksi dengan penyebab banyak bergerak dan tidak ada yang gagal melakukan deteksi. Dari hasil tersebut deteksi paspor menggunakan OCR dan pemecahan kode baris MRZ secara langsung dapat menjadi solusi pada masalah yang disebabkan karena pengambilan gambar secara manual dengan tingkat keberhasilan 90% dengan lancar.