LAMPIRAN

Dokumentasi Proses Implementasi Hasil Pembuatan Aplikasi, di TK Paud Terpadu Melati Kota Pare-pare

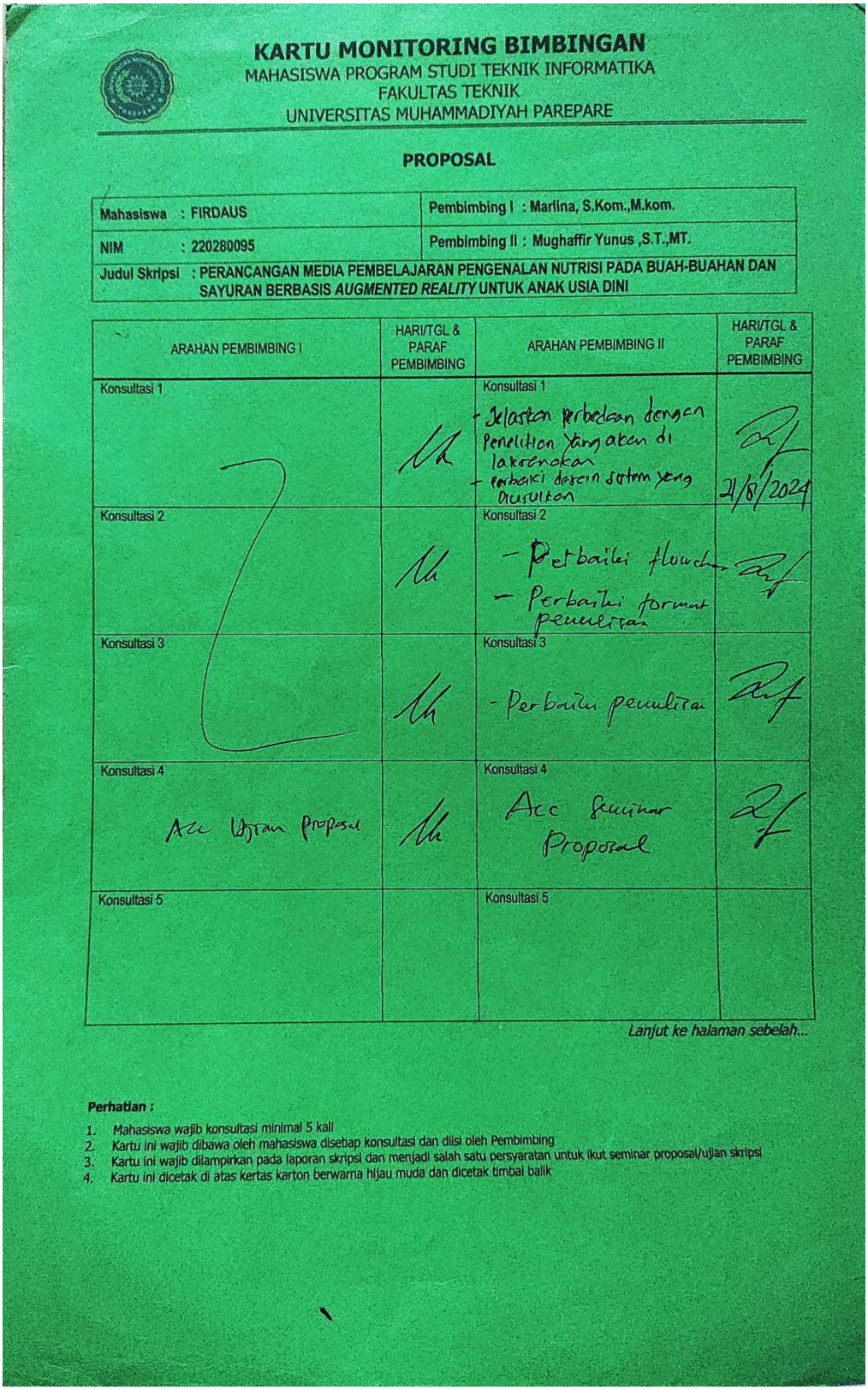
Proses mencoba Aplikasi

Aplikasi dijalankan kepada Anak-anak

Proses Guru mencoba aplikasi

Foto bersama dengan Anak-anak setelah mencoba aplikasi

A. Kartu Monitoring Bimbingan Proposal



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : FIRDAUS	**Pembimbing I : Marlina, S.Kom.,M.kom.**
NIM : 220280095	**Pembimbing II : Mughaffir Yunus ,S.T.,MT.**
Judul Skripsi : PERANCANGAN MEDIA PEMBELAJARAN PENGENALAN NUTRISI PADA BUAH-BUAHAN DAN SAYURAN BERBASIS *AUGMENTED REALITY* UNTUK ANAK USIA DINI	

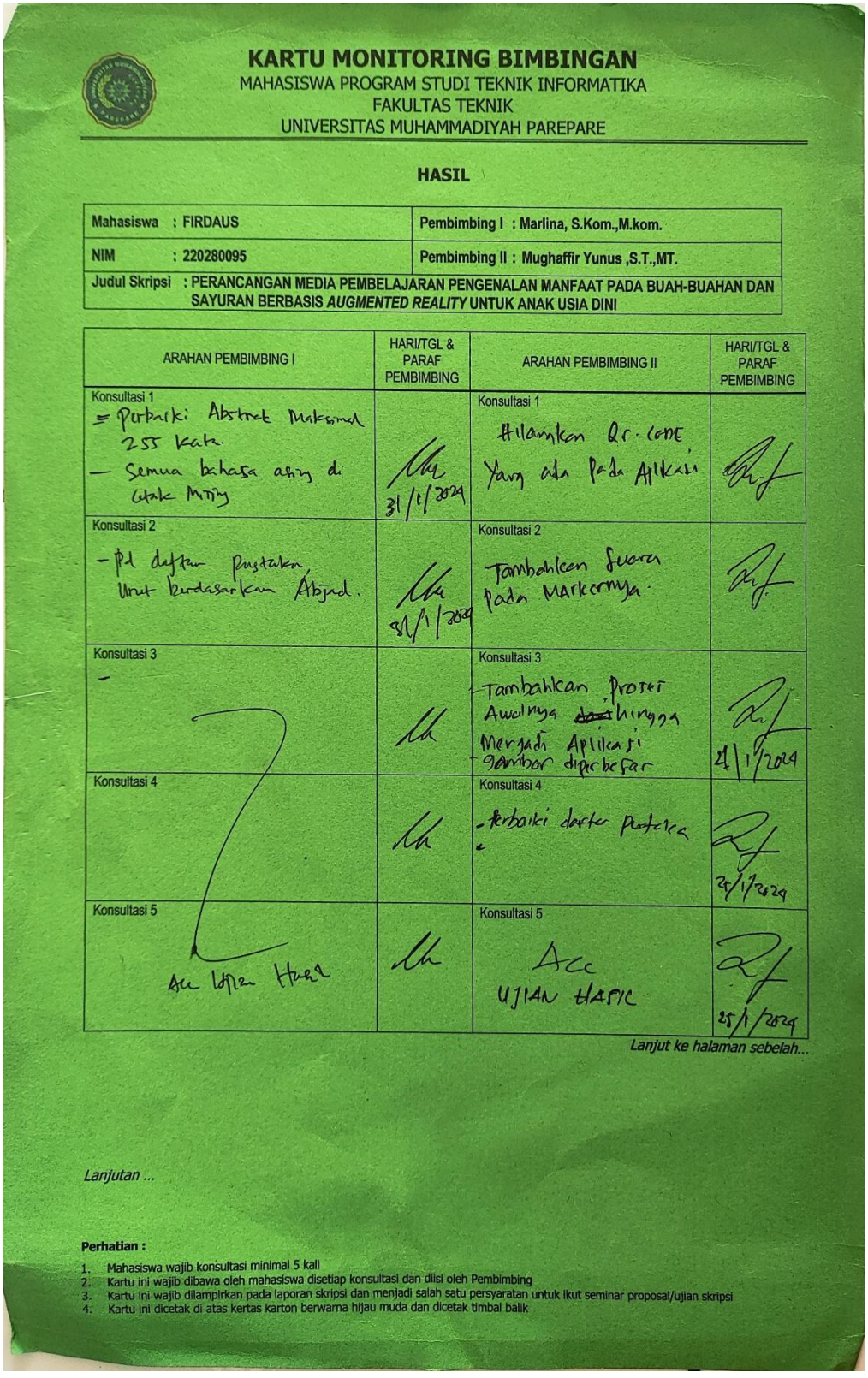
ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1		Konsultasi 1 - Jelaskan perbedaan dengan penelitian yang akan di laksanakan - perbaiki desain sistem yang diusulkan	21/8/2024
Konsultasi 2		Konsultasi 2 - Perbaiki flowchart - Perbaiki format penulisan	
Konsultasi 3		Konsultasi 3 - Perbaiki penulisan	
Konsultasi 4 Acc ujian Proposal		Konsultasi 4 Acc seminar Proposal	
Konsultasi 5		Konsultasi 5	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

B. Kartu Monitoring Bimbingan Hasil



KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

HASIL

Mahasiswa : FIRDAUS	Pembimbing I : Marlina, S.Kom.,M.kom.
NIM : 220280095	Pembimbing II : Mughaffir Yunus ,S.T.,MT.
Judul Skripsi : PERANCANGAN MEDIA PEMBELAJARAN PENGENALAN MANFAAT PADA BUAH-BUAHAN DAN SAYURAN BERBASIS *AUGMENTED REALITY* UNTUK ANAK USIA DINI	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 = Perbaiki Abstrak Maksimal 255 kata. — Semua bahasa asing di cetak Miring	31/1/2024	Konsultasi 1 Hilangkan Qr. code Yang ada Pada Aplikasi	
Konsultasi 2 - Pd daftar Pustaka, Untuk berdasarkan Abjad.	31/1/2024	Konsultasi 2 Tambahkan Suara pada Markernya.	
Konsultasi 3 -		Konsultasi 3 - Tambahkan Proses Awalnya dan hingga Menjadi Aplikasi - gambar diperbesar	24/1/2024
Konsultasi 4		Konsultasi 4 - Perbaiki daftar Pustaka	25/1/2024
Konsultasi 5 Acc Ujian Hasil		Konsultasi 5 Acc UJIAN HASIL	25/1/2024

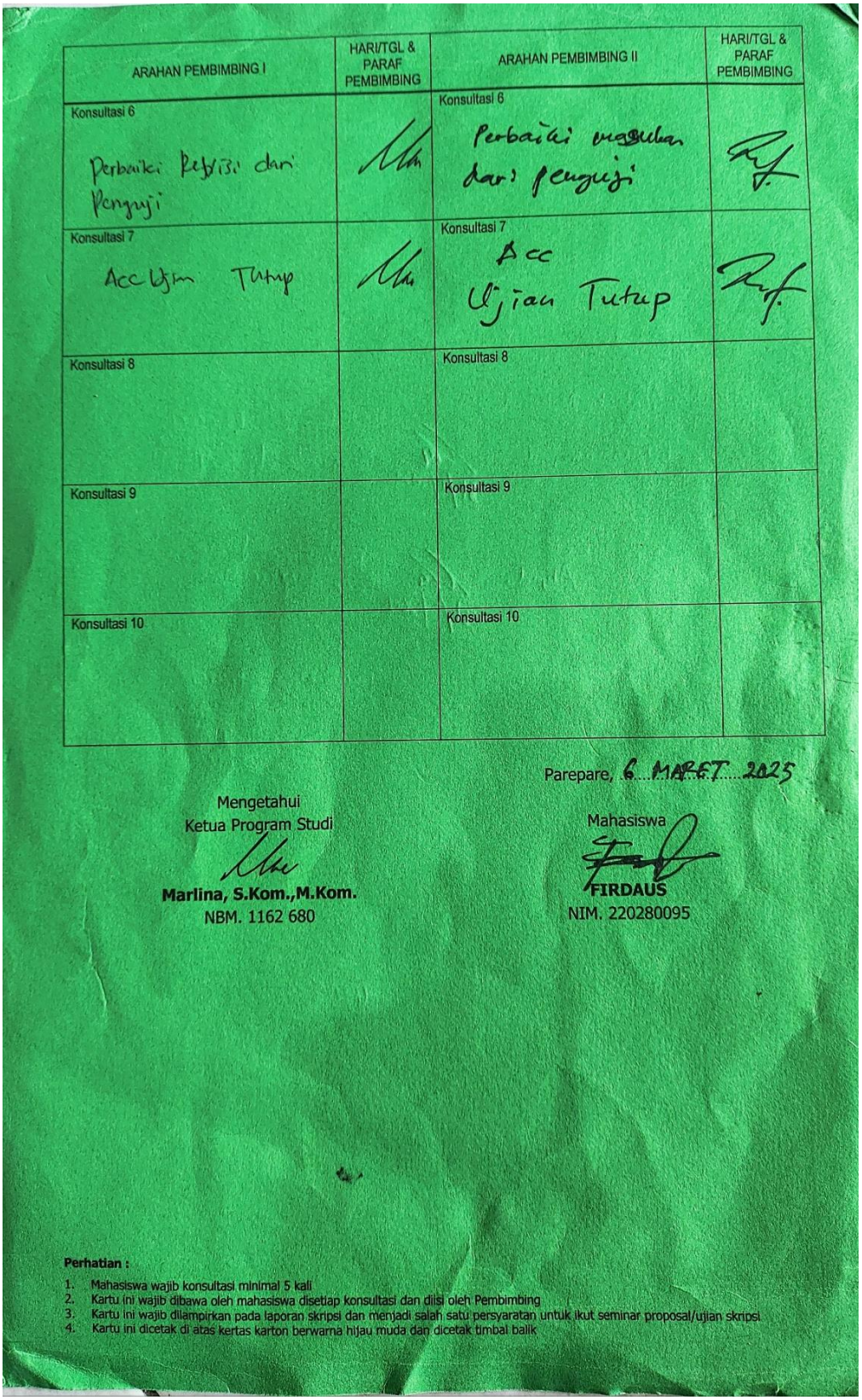
Lanjut ke halaman sebelah...

Lanjutan ...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

C. Kartu Monitoring Bimbingan Tutup



ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6 Perbaiki Revisi dari Penguji		Konsultasi 6 Perbaiki masukan dari penguji	
Konsultasi 7 Acc Ujian Tutup		Konsultasi 7 Acc Ujian Tutup	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare, 6 MARET 2025

Mengetahui
Ketua Program Studi

Marlina, S.Kom.,M.Kom.
NBM. 1162 680

Mahasiswa

FIRDAUS
NIM. 220280095

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Capture And Detect

```
using System.Collections;
using Newtonsoft.Json.Linq;
using UnityEngine;
using UnityEngine.Networking;
using UnityEngine.UI;
using UnityEngine.SceneManagement; //
Menambahkan import untuk SceneManager

public class CaptureAndDetect : MonoBehaviour
{
   // URL API Flask
   private string apiUrl = "http://127.0.0.1:5000/detect";

   // Button untuk menangkap gambar
   public Button captureButton;

   // Tombol untuk reload scene
   public Button reloadButton; // Tombol untuk reload
scene

   public Button infoApel;
   public Button infojeruk;
   public Button infoPisang;

   // Objek 3D untuk Apel, Pisang, dan Jeruk
   public GameObject apelObject; // Class 0 (Apel)
   public GameObject pisangObject; // Class 2 (Pisang)
   public GameObject jerukObject; // Class 1 (Jeruk)

   // Kamera utama yang digunakan untuk menangkap
tampilan
   public Camera mainCamera;

   void Start()
   {
      // Set listener untuk tombol capture
      captureButton.onClick.AddListener(CaptureAndSe
ndImage);

      // Set listener untuk tombol reload scene
      if (reloadButton != null)
      {
         reloadButton.onClick.AddListener(ReloadScene
); // Tombol reload scene
      }

      if (infoApel != null)
      {
         infoApel.gameObject.SetActive(false);
      }

      if (infojeruk != null)
      {
         infojeruk.gameObject.SetActive(false);
      }

      if (infoPisang != null)
      {
         infoPisang.gameObject.SetActive(false);
      }

      // Sembunyikan semua objek 3D di awal
      apelObject.SetActive(false);
      pisangObject.SetActive(false);
      jerukObject.SetActive(false);
   }

   // Fungsi untuk menangkap gambar dan mengirim ke
API Flask
   public void CaptureAndSendImage()
   {
      StartCoroutine(CaptureAndSend());
   }

   private IEnumerator CaptureAndSend()
   {
      // Tunggu sampai frame selesai dirender
      yield return new WaitForEndOfFrame();

      // Tangkap screenshot dari layar
      Texture2D screenshot = new Texture2D(
         Screen.width,
         Screen.height,
         TextureFormat.RGB24,
         false
      );
      screenshot.ReadPixels(new Rect(0, 0,
Screen.width, Screen.height), 0, 0);
      screenshot.Apply();

      // Kirim gambar ke server Flask untuk deteksi
      StartCoroutine(SendImageForDetection(screenshot
));
   }

   private IEnumerator
SendImageForDetection(Texture2D texture)
   {
      // Konversi gambar menjadi byte array (JPG)
      byte[] imageBytes = texture.EncodeToJPG();

      // Buat form data untuk upload gambar
      WWWForm form = new WWWForm();
      form.AddBinaryData("image", imageBytes,
"image.jpg", "image/jpeg");

      // Buat permintaan POST ke API Flask
      using (UnityWebRequest request =
UnityWebRequest.Post(apiUrl, form))
      {
         yield return request.SendWebRequest();

         if (request.result ==
UnityWebRequest.Result.Success)
         {
            Debug.Log("Deteksi Sukses: " +
request.downloadHandler.text);
            ProcessDetectionResults(request.downloadHa
ndler.text);
         }
         else
         {
            Debug.LogError("Gagal menghubungi API: "
+ request.error);
         }
      }
   }

   // Fungsi untuk memproses hasil deteksi dari server
Flask
   private void ProcessDetectionResults(string
response)
   {
      // Parsing JSON hasil deteksi sebagai array (karena
response adalah array)
      JArray detections = JArray.Parse(response);
```

```
        Debug.Log("Jumlah deteksi: " +
detections.Count);

        // Sembunyikan semua objek terlebih dahulu
        apelObject.SetActive(false);
        pisangObject.SetActive(false);
        jerukObject.SetActive(false);

        // Threshold untuk confidence
        float threshold = 0.7f;

        // Variabel untuk menyimpan confidence tertinggi
masing-masing buah
        float bestAppleConfidence = 0f;
        float bestBananaConfidence = 0f;
        float bestOrangeConfidence = 0f;

        string bestAppleClass = "";
        string bestBananaClass = "";
        string bestOrangeClass = "";

        // Iterasi untuk memeriksa setiap deteksi
        foreach (var detection in detections)
        {
            // Get the class name (as a string)
            string className =
detection["class"].ToString();
            float confidence =
(float)detection["confidence"];
            Debug.Log($"Deteksi - Class: {className},
Confidence: {confidence}");

            if (confidence > threshold)
            {
                if (className == "Apel") // Apel (Class ID 0)
                {
                    if (confidence > bestAppleConfidence)
                    {
                        bestAppleConfidence = confidence;
                        bestAppleClass = className;
                    }
                }
                else if (className == "Jeruk") // Jeruk (Class
ID 1)
                {
                    if (confidence > bestOrangeConfidence)
                    {
                        bestOrangeConfidence = confidence;
                        bestOrangeClass = className;
                    }
                }
                else if (className == "Pisang") // Pisang
(Class ID 2)
                {
                    if (confidence > bestBananaConfidence)
                    {
                        bestBananaConfidence = confidence;
                        bestBananaClass = className;
                    }
                }
            }
        }

        // Tampilkan objek jika terdeteksi dengan
confidence tertinggi
        if (bestAppleConfidence > 0)
        {
            apelObject.SetActive(true);
            Debug.Log("Apel terdeteksi dengan confidence
tertinggi: " + bestAppleConfidence);
            if (captureButton && infoApel != null)
            {
                captureButton.gameObject.SetActive(false);
                infoApel.gameObject.SetActive(true);
            }
        }
        if (bestBananaConfidence > 0)
        {
            pisangObject.SetActive(true);
            Debug.Log("Pisang terdeteksi dengan confidence
tertinggi: " + bestBananaConfidence);
            if (captureButton && infoPisang != null)
            {
                captureButton.gameObject.SetActive(false);
                infoPisang.gameObject.SetActive(true);
            }
        }
        if (bestOrangeConfidence > 0)
        {
            jerukObject.SetActive(true);
            Debug.Log("Jeruk terdeteksi dengan confidence
tertinggi: " + bestOrangeConfidence);
            if (captureButton && infojeruk != null)
            {
                captureButton.gameObject.SetActive(false);
                infojeruk.gameObject.SetActive(true);
            }
        }
    }

    // Fungsi untuk reload scene
    private void ReloadScene()
    {
        SceneManager.LoadScene(SceneManager.GetActive
Scene().name); // Reload scene yang sedang aktif
        Debug.Log("Scene telah direload");
    }
}
```

Jawab

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class jawab : MonoBehaviour
{
    public GameObject feed_benar, feed_salah;

    void Start()
    {

    }

    public void jawaban(bool jawab)
    {
        if (jawab){
            feed_benar.SetActive(false);
            feed_benar.SetActive(true);
            int nilai = PlayerPrefs.GetInt("nilai")+1;
            PlayerPrefs.SetInt("nilai", nilai);
        }else{
            feed_salah.SetActive(false);
            feed_salah.SetActive(true);
        }
        gameObject.SetActive(false);
```

```
        transform.parent.GetChild(gameObject.transform.Ge
tSiblingIndex()+1).gameObject.SetActive(true);
    }
    // Update is called once per frame
    void Update()
    {

    }
}
```

Menu

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class menu : MonoBehaviour
{
    public void ScanButton(string scenename)
    {
        SceneManager.LoadScene(scenename);
    }

    public void BackButton(string scenename)
    {
        SceneManager.LoadScene(scenename);
    }
    public void QuitButton()
    {
        Application.Quit();
        Debug.Log("ANJAY");
    }
}
```

Nilai

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class nilai : MonoBehaviour
{

    // Start is called before the first frame update
    void Start()
    {
        PlayerPrefs.SetInt("nilai", 0);
    }

    // Update is called once per frame
    void Update()
    {
        GetComponent<TextMeshProUGUI>().text =
PlayerPrefs.GetInt("nilai").ToString();
    }
}
```

Nilai Akhir

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class NilaiAkhir : MonoBehaviour
{
    public int jumlahSoal = 15; // Jumlah soal
    public int jawabanBenar;    // Jumlah jawaban benar
    private int nilaiAkhir;     // Nilai akhir

    // Start is called before the first frame update
    void Start()
    {
        PlayerPrefs.SetInt("jawabanBenar", 0);
        PlayerPrefs.SetInt("nilaiAkhir", 0);
    }

    // Update is called once per frame
    void Update()
    {
        jawabanBenar = PlayerPrefs.GetInt("nilai");
        nilaiAkhir = (jawabanBenar * 100) / jumlahSoal;
        PlayerPrefs.SetInt("nilaiAkhir", nilaiAkhir);
        GetComponent<TextMeshProUGUI>().text =
PlayerPrefs.GetInt("nilaiAkhir").ToString();
    }
}
```

Panel Click

```
using System.Collections;
using UnityEngine;
using UnityEngine.UI;

public class PanelClick : MonoBehaviour
{
    public GameObject targetObject; // Objek yang ingin di
nonaktifkan
    private Button panelButton;     // Panel yang akan
diklik (pastikan memiliki komponen Button)

    void Start()
    {
        // Mendapatkan komponen Button dari panel
        panelButton = GetComponent<Button>();

        // Menambahkan listener pada event klik button
        if (panelButton != null)
        {
            panelButton.onClick.AddListener(OnPanelClick);
        }
        else
        {
            Debug.LogError("Button component not found on
this panel.");
        }
    }

    void OnPanelClick()
    {
        // Menonaktifkan objek target saat panel diklik
        if (targetObject != null)
        {
            targetObject.SetActive(false);
        }
        else
        {
            Debug.LogError("Target object not assigned.");
        }
    }
}
```

Pop Up Controller

```
using UnityEngine;

public class PopupController : MonoBehaviour
{
    private Animator animator;
    private bool canShowPopup = true;

    void Start()
    {
        animator = GetComponent<Animator>();
    }

    public void ShowPopup()
    {
        if (canShowPopup)
        {
            animator.SetTrigger("Show");
            canShowPopup = false;
        }
    }

    public void HidePopup()
    {
        animator.SetTrigger("Hide");
    }

    // Panggil fungsi ini di akhir animasi "Hide"
    public void EnableShow()
    {
        canShowPopup = true;
    }
}
```

Rotate Object Touch

```
using UnityEngine;

public class PopupController : MonoBehaviour
{
    private Animator animator;
    private bool canShowPopup = true;

    void Start()
    {
        animator = GetComponent<Animator>();
    }

    public void ShowPopup()
    {
        if (canShowPopup)
        {
            animator.SetTrigger("Show");
            canShowPopup = false;
        }
    }

    public void HidePopup()
    {
        animator.SetTrigger("Hide");
    }

    // Panggil fungsi ini di akhir animasi "Hide"
    public void EnableShow()
    {
        canShowPopup = true;
    }
}
```

Scan Controller

```
using UnityEngine;
using UnityEngine.UI;
using Vuforia;

public class ScanController : MonoBehaviour
{
    public GameObject ObjekPertama;
    public GameObject deskripsiPopup;
    public GameObject btnCariTahu;
    private AudioSource audioSource; // AudioSource
sekarang menjadi variabel private
    private AudioSource backgroundMusic; //
Menambahkan variabel untuk background music
    private ObserverBehaviour observerBehaviour;
    private float originalVolume; // Menyimpan volume
asli background music

    void Start()
    {
        if (ObjekPertama != null)
        {
            ObjekPertama.SetActive(false); // Mulai dengan
objek tersembunyi
        }

        if (deskripsiPopup != null)
        {
            deskripsiPopup.SetActive(false);
        }

        if (btnCariTahu != null)
        {
            btnCariTahu.SetActive(false);
        }

        // Inisialisasi ObserverBehaviour dan daftarkan event
handler
        observerBehaviour =
GetComponent<ObserverBehaviour>();
        if (observerBehaviour != null)
        {
            observerBehaviour.OnTargetStatusChanged +=
OnTargetStatusChanged;
        }

        // Temukan AudioSource untuk background music
        backgroundMusic =
GameObject.Find("backsoundController").GetComponen
t<AudioSource>(); // Ganti "BackgroundMusic" dengan
nama GameObject yang sesuai
        if (backgroundMusic != null)
        {
            originalVolume = backgroundMusic.volume; //
Menyimpan volume asli
            backgroundMusic.Play(); // Memutar background
music
        }
    }

    void OnDestroy()
    {
        if (observerBehaviour != null)
        {
            observerBehaviour.OnTargetStatusChanged -=
OnTargetStatusChanged;
        }
    }
```

```
    // Fungsi yang dipanggil ketika tombol deskripsi diklik
    public void ShowDeskripsiPopup()
    {
        if (deskripsiPopup != null)
        {
            deskripsiPopup.SetActive(true);
        }
    }

    // Fungsi yang dipanggil ketika objek terdeteksi
    public void OnObjectScanned()
    {
        if (ObjekPertama != null)
        {
            ObjekPertama.SetActive(true);
        }

        if (btnCariTahu != null)
        {
            btnCariTahu.SetActive(true);
        }

        if (deskripsiPopup != null)
        {
            deskripsiPopup.SetActive(false);
        }

        // Temukan AudioSource di dalam Image Target
        if (observerBehaviour != null)
        {
            audioSource =
observerBehaviour.GetComponentInChildren<AudioSour
ce>();
            if (audioSource != null &&
!audioSource.isPlaying)
            {
                // Mengecilkan volume background music
                if (backgroundMusic != null)
                {
                    backgroundMusic.volume = originalVolume
* 0f;
                }

                audioSource.Play();
                StartCoroutine(ResetVolumeAfterSound());
            }
        }
    }

    // Fungsi yang dipanggil ketika objek hilang
    public void OnObjectLost()
    {
        if (ObjekPertama != null)
        {
            ObjekPertama.SetActive(false);
        }

        if (btnCariTahu != null)
        {
            btnCariTahu.SetActive(false);
        }

        // Hentikan suara saat objek hilang
        if (audioSource != null && audioSource.isPlaying)
        {
            audioSource.Stop();
        }
    }

    // Event handler untuk perubahan status target
    private void
OnTargetStatusChanged(ObserverBehaviour behaviour,
TargetStatus status)
    {
        if (status.Status == Status.TRACKED ||
            status.Status == Status.EXTENDED_TRACKED)
        {
            // Objek ditemukan
            OnObjectScanned();
        }
        else
        {
            // Objek hilang
            OnObjectLost();
        }
    }

    private System.Collections.IEnumerator
ResetVolumeAfterSound()
    {
        // Tunggu sampai audio source selesai diputar
        yield return new
WaitForSeconds(audioSource.clip.length);

        // Mengembalikan volume background music ke
nilai asli
        if (backgroundMusic != null)
        {
            backgroundMusic.volume = originalVolume;
        }
    }
}
```

Sound Controller

```
using UnityEngine;
using UnityEngine.SceneManagement;
using System.Collections; // Pastikan ini ditambahkan

public class soundController : MonoBehaviour
{
    public AudioClip buttonSound;
    private AudioSource audioSource;

    void Start()
    {
        audioSource = GetComponent<AudioSource>();
    }

    public void OnButtonPress(string sceneName)
    {
        audioSource.PlayOneShot(buttonSound);
        StartCoroutine(ChangeScene(sceneName));
    }

    private IEnumerator ChangeScene(string sceneName)
    {
        yield return new
WaitForSeconds(buttonSound.length);
        SceneManager.LoadScene(sceneName);
    }
}
```

Target Frame Rate

```
using UnityEngine;

public class TargetFrameRate : MonoBehaviour
{
   void Awake()
   {
      Application.targetFrameRate = 60;
   }
}
```

Voice Management

```
using System.Collections;
using UnityEngine;
using UnityEngine.UI;

public class voiceManagement : MonoBehaviour
{
   // Variabel untuk menyimpan suara
   public AudioClip pengertian;
   public AudioClip ciri;
   public AudioClip manfaat;
   private AudioSource audioSource;
   private AudioSource backgroundMusic;
   private float originalVolume;

   // Tombol untuk memainkan suara
   public Button btnPengertian;
   public Button btnCiri;
   public Button btnManfaat;

   // Tombol Oke untuk mengembalikan volume latar
belakang
   public Button btnOke;

   void Start()
   {
      // Inisialisasi AudioSource
      audioSource = GetComponent<AudioSource>();

      // Tambahkan event listener ke tombol
      btnPengertian.onClick.AddListener(() =>
PlayAudio(pengertian));
      btnCiri.onClick.AddListener(() => PlayAudio(ciri));
      btnManfaat.onClick.AddListener(() =>
PlayAudio(manfaat));

      // Tambahkan event listener untuk tombol Oke
      btnOke.onClick.AddListener(RestoreBackgroundVol
ume);

      // Cari dan inisialisasi background music
      backgroundMusic =
GameObject.Find("backsoundController").GetComponen
t<AudioSource>();
      if (backgroundMusic != null)
      {
         originalVolume = backgroundMusic.volume; //
Menyimpan volume asli
         backgroundMusic.Play(); // Memutar background
music
      }
   }

   void PlayAudio(AudioClip clip)
   {
      if (audioSource == null || backgroundMusic == null ||
clip == null)
         return;

      // Berhentikan suara yang sedang diputar di
audioSource
      audioSource.Stop();

      // Mengecilkan volume background music
      backgroundMusic.volume = originalVolume * 0f; //
Sesuaikan angka untuk tingkat pengecilan

      // Mainkan suara baru di audioSource
      audioSource.clip = clip;
      audioSource.Play();

      // Kembalikan volume background music setelah
suara selesai
      StartCoroutine(ResetVolumeAfterSound(audioSourc
e.clip.length));
   }

   private IEnumerator ResetVolumeAfterSound(float
delay)
   {
      yield return new WaitForSeconds(delay);

      // Kembalikan volume background music ke nilai
asli
      if (backgroundMusic != null)
      {
         backgroundMusic.volume = originalVolume;
      }
   }

   // Fungsi untuk mengembalikan volume background
music ketika tombol Oke ditekan
   private void RestoreBackgroundVolume()
   {
      if (backgroundMusic != null)
      {
         backgroundMusic.volume = originalVolume;
      }
   }
}
```