**L**

**A**

**M**

**P**

**I**

**R**

**A**

**N**

**StandradCustomShader.shader**

/ *Unity* built-in shader source. Copyright (c) 2016 *Unity* Technologies. MIT license (see license.txt)

```
Shader "StandardCustomShader"
{
   Properties
   {
      _Color("Color", Color) = (1,1,1,1)
      _MainTex("Albedo", 2D) = "white" {}

      _Cutoff("Alpha Cutoff", Range(0.0, 1.0)) = 0.5

      _Glossiness("Smoothness", Range(0.0, 1.0)) = 0.5
      _GlossMapScale("Smoothness Scale", Range(0.0, 1.0)) = 1.0
      [Enum(Metallic Alpha,0,Albedo Alpha,1)] _SmoothnessTextureChannel("Smoothness texture channel", Float) = 0

      [Gamma] _Metallic("Metallic", Range(0.0, 1.0)) = 0.0
      _MetallicGlossMap("Metallic", 2D) = "white" {}

      [ToggleOff] _SpecularHighlights("Specular Highlights", Float) = 1.0
      [ToggleOff] _GlossyReflections("Glossy Reflections", Float) = 1.0

      _BumpScale("Scale", Float) = 1.0
      [Normal] _BumpMap("Normal Map", 2D) = "bump" {}

      _Parallax("Height Scale", Range(0.005, 0.08)) = 0.02
      _ParallaxMap("Height Map", 2D) = "black" {}

      _OcclusionStrength("Strength", Range(0.0, 1.0)) = 1.0
      _OcclusionMap("Occlusion", 2D) = "white" {}

      _EmissionColor("Color", Color) = (0,0,0)
      _EmissionMap("Emission", 2D) = "white" {}

      _DetailMask("Detail Mask", 2D) = "white" {}

      _DetailAlbedoMap("Detail Albedo x2", 2D) = "grey" {}
      _DetailNormalMapScale("Scale", Float) = 1.0
      [Normal] _DetailNormalMap("Normal Map", 2D) = "bump" {}

      [Enum(UV0,0,UV1,1)] _UVSec("UV Set for secondary textures", Float) = 0


      // Blending state
      [HideInInspector] Mode("_mode", Float) = 0.0
      [HideInInspector] SrcBlend("_src", Float) = 1.0
      [HideInInspector] DstBlend("_dst", Float) = 0.0
      [HideInInspector] ZWrite("_zw", Float) = 1.0
   }

   CGINCLUDE
#define UNITY_SETUP_BRDF_INPUT MetallicSetup
      ENDCG

      SubShader
   {
      Tags { "RenderType" = "Opaque" "PerformanceChecks" = "False" }
      LOD 300

      // ------------------------------------------------------------------
```

```
        //  Base forward pass (directional light, emission, lightmaps, ...)

        Pass

        {

            Name "FORWARD"

            Tags { "LightMode" = "ForwardBase" }


            Blend[_SrcBlend][_DstBlend]

            ZWrite[_ZWrite]


            CGPROGRAM

            #pragma target 3.0


        // -------------------------------------


        #pragma shader_feature_local _NORMALMAP

        #pragma shader_feature_local _ _ALPHATEST_ON _ALPHABLEND_ON _ALPHAPREMULTIPLY_ON

        #pragma shader_feature _EMISSION

        #pragma shader_feature_local _METALLICGLOSSMAP

        #pragma shader_feature_local _DETAIL_MULX2

        #pragma shader_feature_local _SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_A

        #pragma shader_feature_local _SPECULARHIGHLIGHTS_OFF

        #pragma shader_feature_local _GLOSSYREFLECTIONS_OFF

        #pragma shader_feature_local _PARALLAXMAP


        #pragma multi_compile_fwdbase

        #pragma multi_compile_fog

        #pragma multi_compile_instancing

        // Uncomment the following line to enable dithering LOD crossfade. Note: there are more in the file to uncomment for other passes.

        //#pragma multi_compile _ LOD_FADE_CROSSFADE


        #pragma vertex vertBase

        #pragma fragment fragBase

        #include "Unity*StandardCoreForward.cginc"


        ENDCG

    }

        // ------------------------------------------------------------------

        //  Additive forward pass (one light per pass)

        Pass

        {

            Name "FORWARD_DELTA"

            Tags { "LightMode" = "ForwardAdd" }

            Blend[_SrcBlend] One

            Fog { Color(0,0,0,0) } // in additive pass fog should be black

            ZWrite Off

            ZTest LEqual


            CGPROGRAM

            #pragma target 3.0


        // -------------------------------------


        #pragma shader_feature_local _NORMALMAP

        #pragma shader_feature_local _ _ALPHATEST_ON _ALPHABLEND_ON _ALPHAPREMULTIPLY_ON

        #pragma shader_feature_local _METALLICGLOSSMAP

        #pragma shader_feature_local _SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_A

        #pragma shader_feature_local _SPECULARHIGHLIGHTS_OFF

        #pragma shader_feature_local _DETAIL_MULX2
```

```
        #pragma shader_feature_local
_PARALLAXMAP

        #pragma multi_compile_fwdadd_fullshadows

        #pragma multi_compile_fog

        // Uncomment the following line to enable
dithering LOD crossfade. Note: there are more in the
file to uncomment for other passes.

        //#pragma multi_compile _
LOD_FADE_CROSSFADE

        #pragma vertex vertAdd

        #pragma fragment fragAdd

        #include "UnityStandardCoreForward.cginc"

        ENDCG

    }

        // ------------------------------------------------------
----------

        //  Shadow rendering pass

        Pass {

            Name "ShadowCaster"

            Tags { "LightMode" = "ShadowCaster" }

            ZWrite On ZTest LEqual

            CGPROGRAM

            #pragma target 3.0

        // ------------------------------------

        #pragma shader_feature_local _
_ALPHATEST_ON _ALPHABLEND_ON
_ALPHAPREMULTIPLY_ON

        #pragma shader_feature_local
_METALLICGLOSSMAP

        #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A

        #pragma shader_feature_local
_PARALLAXMAP

        #pragma multi_compile_shadowcaster

        #pragma multi_compile_instancing

        // Uncomment the following line to enable
dithering LOD crossfade. Note: there are more in the
file to uncomment for other passes.

        //#pragma multi_compile _
LOD_FADE_CROSSFADE

        #pragma vertex vertShadowCaster

        #pragma fragment fragShadowCaster

        #include "UnityStandardShadow.cginc"

        ENDCG

    }

        // ------------------------------------------------------
----------

        //  Deferred pass

        Pass

        {

            Name "DEFERRED"

            Tags { "LightMode" = "Deferred" }

            CGPROGRAM

            #pragma target 3.0

            #pragma exclude_renderers nomrt

        // ------------------------------------

        #pragma shader_feature_local _NORMALMAP

        #pragma shader_feature_local _
_ALPHATEST_ON _ALPHABLEND_ON
_ALPHAPREMULTIPLY_ON

        #pragma shader_feature _EMISSION

        #pragma shader_feature_local
_METALLICGLOSSMAP
```

```
        #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A

        #pragma shader_feature_local
_SPECULARHIGHLIGHTS_OFF

        #pragma shader_feature_local
_DETAIL_MULX2

        #pragma shader_feature_local
_PARALLAXMAP



        #pragma multi_compile_prepassfinal

        #pragma multi_compile_instancing

        // Uncomment the following line to enable
dithering LOD crossfade. Note: there are more in the
file to uncomment for other passes.

        //#pragma multi_compile _
LOD_FADE_CROSSFADE



        #pragma vertex vertDeferred

        #pragma fragment fragDeferred


        #include "UnityStandardCore.cginc"


        ENDCG

      }


        // ------------------------------------------------------
----------

        // Extracts information for lightmapping, GI
(emission, albedo, ...)

        // This pass it not used during regular rendering.

        Pass

        {

          Name "META"

          Tags { "LightMode" = "Meta" }


          Cull Off


          CGPROGRAM

          #pragma vertex vert_meta

          #pragma fragment frag_meta


          #pragma shader_feature _EMISSION

          #pragma shader_feature_local
_METALLICGLOSSMAP

          #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A

          #pragma shader_feature_local
_DETAIL_MULX2

          #pragma shader_feature
EDITOR_VISUALIZATION


          #include "UnityStandardMeta.cginc"

          ENDCG

        }

      }


        SubShader

      {

        Tags { "RenderType" = "Opaque"
"PerformanceChecks" = "False" }

        LOD 150


        // ------------------------------------------------------
----------

        //  Base forward pass (directional light,
emission, lightmaps, ...)

        Pass

        {

          Name "FORWARD"

          Tags { "LightMode" = "ForwardBase" }


          Blend[_SrcBlend][_DstBlend]

          ZWrite[_ZWrite]


          CGPROGRAM

          #pragma target 2.0
```

```
            #pragma shader_feature_local
_NORMALMAP

            #pragma shader_feature_local _
_ALPHATEST_ON _ALPHABLEND_ON
_ALPHAPREMULTIPLY_ON

            #pragma shader_feature _EMISSION

            #pragma shader_feature_local
_METALLICGLOSSMAP

            #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A

            #pragma shader_feature_local
_SPECULARHIGHLIGHTS_OFF

            #pragma shader_feature_local
_GLOSSYREFLECTIONS_OFF

        // SM2.0: NOT SUPPORTED
shader_feature_local _DETAIL_MULX2

        // SM2.0: NOT SUPPORTED
shader_feature_local _PARALLAXMAP



        #pragma skip_variants SHADOWS_SOFT
DIRLIGHTMAP_COMBINED



        #pragma multi_compile_fwdbase

        #pragma multi_compile_fog



        #pragma vertex vertBase

        #pragma fragment fragBase

        #include "Unity StandardCoreForward.cginc"



        ENDCG

    }

        // ------------------------------------------------------
----------

        //  Additive forward pass (one light per pass)

        Pass

        {

            Name "FORWARD_DELTA"

            Tags { "LightMode" = "ForwardAdd" }

            Blend[_SrcBlend] One

            Fog { Color(0,0,0,0) } // in additive pass fog
should be black

            ZWrite Off

            ZTest LEqual



            CGPROGRAM

            #pragma target 2.0



            #pragma shader_feature_local
_NORMALMAP

            #pragma shader_feature_local _
_ALPHATEST_ON _ALPHABLEND_ON
_ALPHAPREMULTIPLY_ON

            #pragma shader_feature_local
_METALLICGLOSSMAP

            #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A

            #pragma shader_feature_local
_SPECULARHIGHLIGHTS_OFF

            #pragma shader_feature_local
_DETAIL_MULX2

        // SM2.0: NOT SUPPORTED
shader_feature_local _PARALLAXMAP

        #pragma skip_variants SHADOWS_SOFT



        #pragma multi_compile_fwdadd_fullshadows

        #pragma multi_compile_fog



        #pragma vertex vertAdd

        #pragma fragment fragAdd

        #include "Unity StandardCoreForward.cginc"



        ENDCG

    }

        // ------------------------------------------------------
----------

        //  Shadow rendering pass

        Pass {

            Name "ShadowCaster"

            Tags { "LightMode" = "ShadowCaster" }
```

```
            ZWrite On ZTest LEqual

            CGPROGRAM
            #pragma target 2.0

            #pragma shader_feature_local _
_ALPHATEST_ON _ALPHABLEND_ON
_ALPHAPREMULTIPLY_ON
            #pragma shader_feature_local
_METALLICGLOSSMAP
            #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A
            #pragma skip_variants SHADOWS_SOFT
            #pragma multi_compile_shadowcaster

            #pragma vertex vertShadowCaster
            #pragma fragment fragShadowCaster

            #include "UnityStandardShadow.cginc"

            ENDCG
        }

        // ------------------------------------------------------------
        // Extracts information for lightmapping, GI (emission, albedo, ...)
        // This pass it not used during regular rendering.
        Pass
        {
            Name "META"
            Tags { "LightMode" = "Meta" }

            Cull Off

            CGPROGRAM
            #pragma vertex vert_meta
            #pragma fragment frag_meta

            #pragma shader_feature _EMISSION
            #pragma shader_feature_local
_METALLICGLOSSMAP
            #pragma shader_feature_local
_SMOOTHNESS_TEXTURE_ALBEDO_CHANNEL_
A
            #pragma shader_feature_local
_DETAIL_MULX2
            #pragma shader_feature
EDITOR_VISUALIZATION

            #include "UnityStandardMeta.cginc"
            ENDCG
        }
    }

    FallBack "VertexLit"
    CustomEditor "StandardShaderGUI"
}
```

**TransparentShader.shader**

```
Shader "TransparencyMask" {
  Properties{
    _Color("Color", Color) = (1,1,1,1)
    _MainTex("Albedo (RGB)", 2D) = "white" {}
    _Glossiness("Smoothness", Range(0,1)) = 0.5
    _Metallic("Metallic", Range(0,1)) = 0.0
  }
    SubShader{
      Tags { "Queue" = "Transparent-499" "RenderType" = "Transparent" }
      Cull Off

      // Don't draw in the RGBA channels; just the depth buffer
      ColorMask 0
      ZWrite On
```

```
        // Do nothing specific in the pass

        Pass { }

        CGPROGRAM

        // Physically based Standard lighting model, and
enable shadows on all light types

        #pragma surface surf Standard
fullforwardshadows alpha:fade


        // Use shader model 3.0 target, to get nicer
looking lighting

        #pragma target 3.0


        sampler2D _MainTex;


        struct Input {

            float2 uv_MainTex;

        };


        half _Glossiness;

        half _Metallic;

        fixed4 _Color;


        // Add instancing support for this shader. You
need to check 'Enable Instancing' on materials that use
the shader.

        // See
https://docs.unity3d.com/Manual/GPUInstancing.html
for more information about instancing.

        // #pragma instancing_options
assumeuniformscaling


UNITY_INSTANCING_BUFFER_START(Props)

            // put more per-instance properties here

        UNITY_INSTANCING_BUFFER_END(Props)


        void surf(Input IN, inout SurfaceOutputStandard
o) {

            // Albedo comes from a texture tinted by color

            fixed4 c = tex2D(_MainTex, IN.uv_MainTex)
* _Color;

            o.Albedo = c.rgb;

            // Metallic and smoothness come from slider
variables

            o.Metallic = _Metallic;

            o.Smoothness = _Glossiness;

            o.Alpha = c.a;

        }

        ENDCG

    }

        FallBack "Diffuse"

}
```

**ARFeatheredPlaneMeshVisualizer.cs**

```
Shader "TransparencyMask" {
  Properties{
    _Color("Color", Color) = (1,1,1,1)
    _MainTex("Albedo (RGB)", 2D) = "white" {}
    _Glossiness("Smoothness", Range(0,1)) = 0.5
    _Metallic("Metallic", Range(0,1)) = 0.0
  }
    SubShader{
      Tags { "Queue" = "Transparent-499"
          "RenderType" = "Transparent" }
      Cull Off

      // Don't draw in the RGBA channels; just the
          depth buffer
      ColorMask 0
      ZWrite On

      // Do nothing specific in the pass
      Pass { }
      CGPROGRAM
      // Physically based Standard lighting model, and
          enable shadows on all light types
      #pragma surface surf Standard
          fullforwardshadows alpha:fade

      // Use shader model 3.0 target, to get nicer
          looking lighting
      #pragma target 3.0

      sampler2D _MainTex;

      struct Input {
        float2 uv_MainTex;
      };

      half _Glossiness;
      half _Metallic;
      fixed4 _Color;

      // Add instancing support for this shader. You
          need to check 'Enable Instancing' on
          materials that use the shader.
```

```
        //                                          See
            https://docs.unity3d.com/Manual/GPUInstan
            cing.html for more information about
            instancing.
        //          #pragma instancing_options
            assumeuniformscaling

            UNITY_INSTANCING_BUFFER_START(
            Props)
          // put more per-instance properties here
        UNITY_INSTANCING_BUFFER_END(Props)

        void surf(Input IN, inout SurfaceOutputStandard
            o) {
          // Albedo comes from a texture tinted by color
          fixed4 c = tex2D(_MainTex, IN.uv_MainTex)
            * _Color;
          o.Albedo = c.rgb;
          // Metallic and smoothness come from slider
            variables
          o.Metallic = _Metallic;
          o.Smoothness = _Glossiness;
          o.Alpha = c.a;
        }
        ENDCG
    }
        FallBack "Diffuse"
}
```

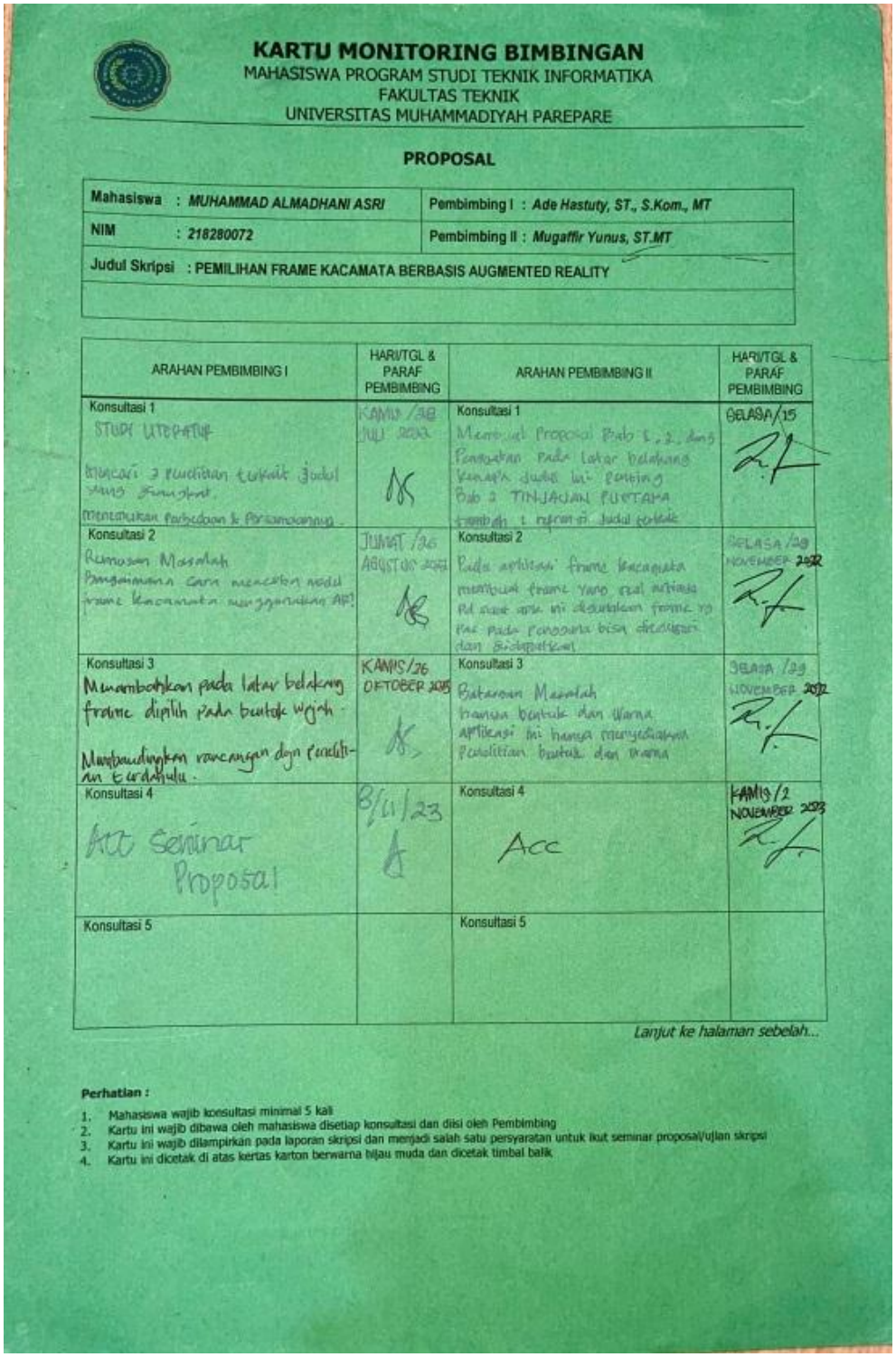
**AROcclussionQualityController.cs**

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

/// <summary>
/// This script check whether your device support
        AROcclussion or not and setup
        AROcclusion quality level
/// </summary>
[RequireComponent(typeof(AROcclusionManager))]
public class AROcclussionQualityController :
        MonoBehaviour
{
  private AROcclusionManager aROcclusionManager;
  public GameObject qualityPanel;
  private Transform bestOptionButton;
  private bool supportDepth = false;
  public GameObject warningPanel;

  private void Awake()
  {
    aROcclusionManager                          =
        GetComponent<AROcclusionManager>();
  }

  private void Start()
  {
    qualityPanel.SetActive(false);
    warningPanel.SetActive(false);
    bestOptionButton                            =
        qualityPanel.transform.Find("Best");

#if UNITY_IOS

        bestOptionButton.gameObject.GetCompone
        nt<Button>().enabled = false;
#endif

    var occlusionDescriptors = new
        List<XROcclusionSubsystemDescriptor>();

        SubsystemManager.GetSubsystemDescriptor
        s(occlusionDescriptors);

    if (occlusionDescriptors.Count > 0)
    {
      foreach (var occlusionDescriptor in
        occlusionDescriptors)
      {
        //Check whether your device support AR
        Occlusion or not by checking support Depth
        API to your device
        if
        (occlusionDescriptor.supportsEnvironmentD
        epthImage                                ||
        occlusionDescriptor.supportsEnvironmentDe
        pthConfidenceImage
          ||
        occlusionDescriptor.supportsHumanSegment
        ationDepthImage                          ||
        occlusionDescriptor.supportsHumanSegment
        ationStencilImage)
        {
          supportDepth = true;
        }
      }
    }
  }

  /// <summary>
  /// Set AR Occlusion quality level to Medium settings
  /// </summary>
  public void ChangeQualityToMedium()
  {

        aROcclusionManager.requestedEnvironment
        DepthMode                                =
        EnvironmentDepthMode.Medium;
    ShowQualityPanel(false);
  }

  /// <summary>
  /// Set AR Occlusion quality level to Fastest settings
  /// </summary>
  public void ChangeQualityToFastest()
  {

        aROcclusionManager.requestedEnvironment
```

```
            DepthMode                                =
            EnvironmentDepthMode.Fastest;
        ShowQualityPanel(false);
    }

    /// <summary>
    /// Set AR Occlusion quality level to Fastest settings
    /// </summary>
    public void ChangeQualityToBest()
    {

            aROcclusionManager.requestedEnvironment
            DepthMode = EnvironmentDepthMode.Best;
        ShowQualityPanel(false);
    }

    /// <summary>
    /// Disable AR Occlusion
    /// </summary>
    public void ChangeQualityToNoOcclussion()
    {

            aROcclusionManager.requestedEnvironment
            DepthMode                                =
            EnvironmentDepthMode.Disabled;
        ShowQualityPanel(false);
    }

    /// <summary>
    /// Show/Hide Quality control panel
    /// </summary>
    public void ShowQualityPanel(bool show)
    {
        if (supportDepth)
        {
            qualityPanel.SetActive(show);
        }
        else
        {
            //show warning panel for 2 seconds
            warningPanel.SetActive(true);
            StartCoroutine(ShowWarningPanel());
        }
    }

    IEnumerator ShowWarningPanel()
    {
        yield return new WaitForSeconds(2);
        warningPanel.SetActive(false);
    }
}
```

**DoubleTap.cs**

```
using UnityEngine;

/// <summary>
/// To detect Double tap
/// </summary>
public class DoubleTap : MonoBehaviour
{
    private int tapCount = 0;
    private bool waitingTimeStarted = false;
    private float time = 0;
    private float waitingTime = 0.5f;

    private                const                float
            MINIMUMTIMETOSECONDTAPSTART
            = 0.2f;

    void Update()
    {
        // Handle screen touches
        if (Input.touchCount == 1)
        {
            if (!waitingTimeStarted)
            {
                time = 0;
                tapCount = 0;
            }
            waitingTimeStarted = true;
            Touch touch = Input.GetTouch(0);
            if (tapCount < 1)
            {
                if (touch.phase == TouchPhase.Began)
                {
                    tapCount++;
                }
                else if (touch.phase == TouchPhase.Moved
                 &&                time                >
                 MINIMUMTIMETOSECONDTAPSTART)
                {
                    tapCount = 0;
                    time = 0;
                }
            }
            else
            {
                if (touch.phase == TouchPhase.Began)
                {
                    tapCount++;
                }
                else if (touch.phase == TouchPhase.Moved
                 &&                time                >
                 MINIMUMTIMETOSECONDTAPSTART)
                {
                    tapCount = 0;
                    time = 0;
                }

            }

            if (time <= waitingTime && tapCount > 1)
            {
```



# KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

## PROPOSAL

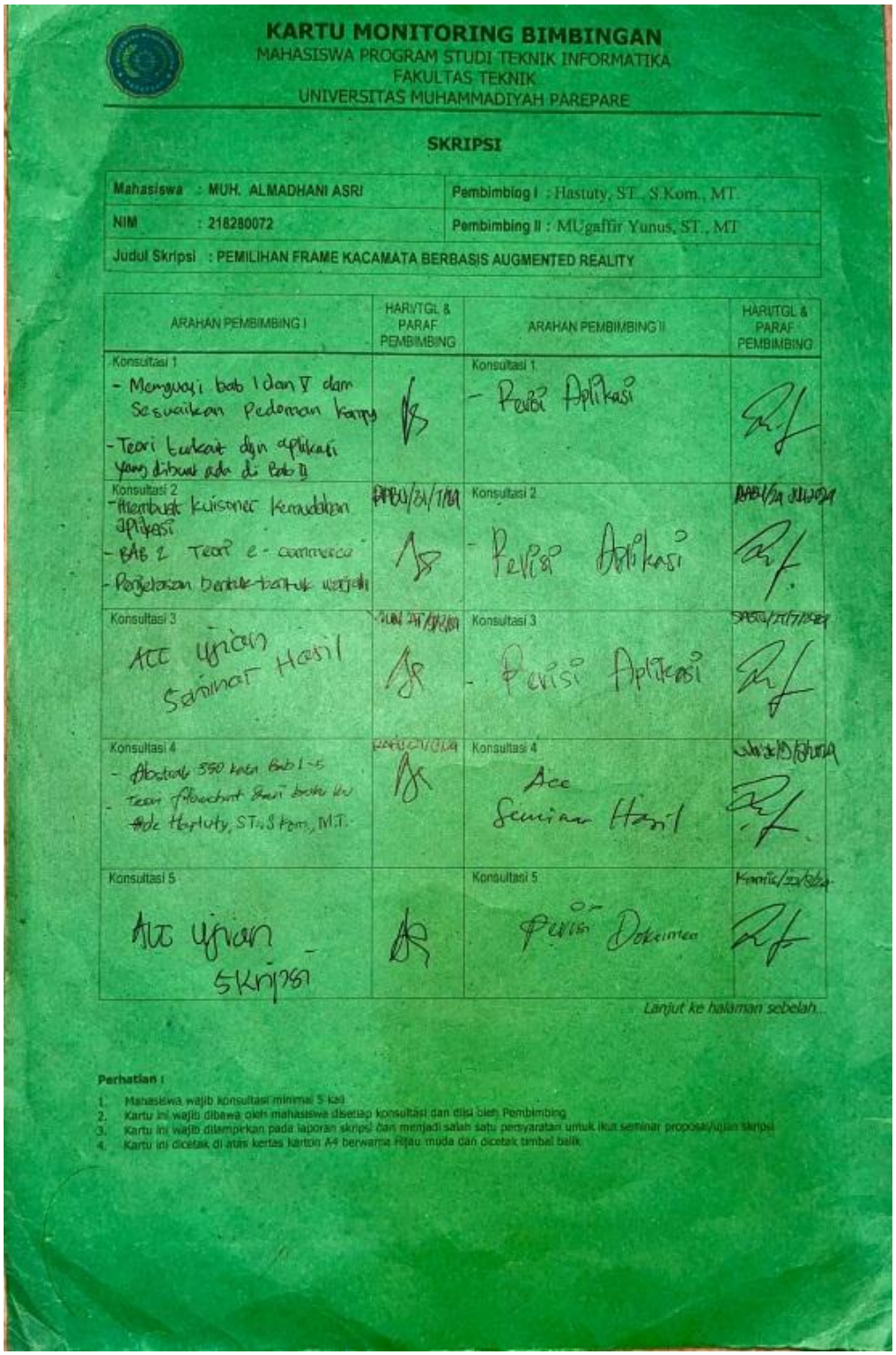
| | |
|---|---|
| Mahasiswa : *MUHAMMAD ALMADHANI ASRI* | Pembimbing I : *Ade Hastuty, ST., S.Kom., MT* |
| NIM : *218280072* | Pembimbing II : *Mugaffir Yunus, ST.MT* |
| Judul Skripsi : PEMILIHAN FRAME KACAMATA BERBASIS AUGMENTED REALITY | |

| ARAHAN PEMBIMBING I | HARI/TGL & PARAF PEMBIMBING | ARAHAN PEMBIMBING II | HARI/TGL & PARAF PEMBIMBING |
|---|---|---|---|
| Konsultasi 1<br>STUDI LITERATUR<br>Mencari 2 Puchisan terkait Judul yang Diangkat.<br>Menemukan perbedaan & Persamaannya. | KAMIS / 20 JULI 2023 | Konsultasi 1<br>Memperbaiki Proposal Bab 1, 2, dan Penambahan pada latar belakang kenapa judul ini penting<br>Bab 2 TINJAUAN PUSTAKA tambah 1 referensi judul terkait | SELASA / 15 |
| Konsultasi 2<br>Rumusan Masalah<br>Bagaimana cara mencoba model frame kacamata menggunakan AR? | JUMAT / 25 AGUSTUS 2023 | Konsultasi 2<br>Pada aplikasi frame kacamata membuat frame yang real nyatanya di sini aplikasinya frame yg pas pada perosona bisa dicoba dan disimpan | SELASA / 28 NOVEMBER 2023 |
| Konsultasi 3<br>Menambahkan pada latar belakang frame dipilih pada bentuk wajah.<br>Membandingkan rancangan dgn penelitian terdahulu. | KAMIS / 26 OKTOBER 2023 | Konsultasi 3<br>Batasan Masalah<br>hanya bentuk dan warna<br>Aplikasi ini hanya menyediakan Penelitian bentuk dan warna | SELASA / 28 NOVEMBER 2023 |
| Konsultasi 4<br>Acc Seminar Proposal | 8/11/23 | Konsultasi 4<br>Acc | KAMIS / 2 NOVEMBER 2023 |
| 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



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

**SKRIPSI**

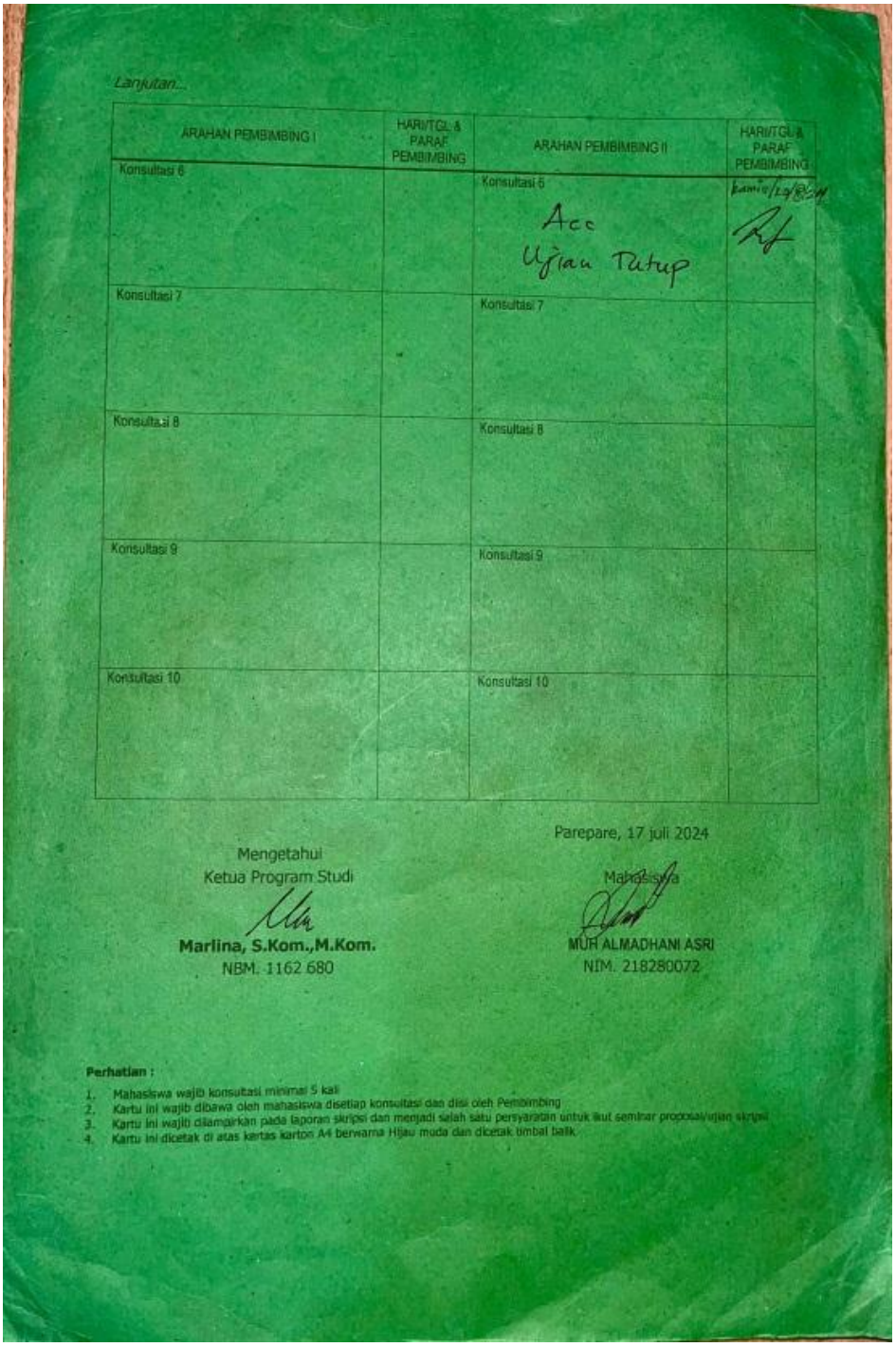
| | |
|---|---|
| Mahasiswa : MUH. ALMADHANI ASRI | Pembimbing I : Hastuty, ST., S.Kom., MT. |
| NIM : 218280072 | Pembimbing II : M. Jaffar Yunus, ST., MT |
| Judul Skripsi : PEMILIHAN FRAME KACAMATA BERBASIS AUGMENTED REALITY | |

| ARAHAN PEMBIMBING I | HARI/TGL & PARAF PEMBIMBING | ARAHAN PEMBIMBING II | HARI/TGL & PARAF PEMBIMBING |
|---|---|---|---|
| Konsultasi 1<br>- Mengacu bab I dan V dan Sesuaikan Pedoman kampus<br>- Teori terkait dan aplikasi yang dibuat ada di Bab II | | Konsultasi 1<br>- Revisi Aplikasi | |
| Konsultasi 2<br>- Membuat kuisioner kemudahan aplikasi<br>- Bab 2 Teori e-commerce<br>- Penjelasan berbentuk uji coba | | Konsultasi 2<br>- Revisi Aplikasi | |
| Konsultasi 3<br>Acc ujian Seminar Hasil | | Konsultasi 3<br>- Revisi Aplikasi | |
| Konsultasi 4<br>- Abstrak 350 kata Bab I-5<br>- Teori pendukung lebih banyak lagi | | Konsultasi 4<br>Acc Seminar Hasil | |
| Konsultasi 5<br>Acc ujian Skripsi | | Konsultasi 5<br>Revisi Dokumen | |

*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 A4 berwarna Hijau muda dan dicetak timbal balik



*Lanjutan...*

| ARAHAN PEMBIMBING I | HARI/TGL & PARAF PEMBIMBING | ARAHAN PEMBIMBING II | HARI/TGL & PARAF PEMBIMBING |
|---|---|---|---|
| Konsultasi 6 | | Konsultasi 6<br>Acc<br>Ujian Tutup | Kamis/18/8-24 |
| Konsultasi 7 | | Konsultasi 7 | |
| Konsultasi 8 | | Konsultasi 8 | |
| Konsultasi 9 | | Konsultasi 9 | |
| Konsultasi 10 | | Konsultasi 10 | |

Parepare, 17 juli 2024

Mengetahui
Ketua Program Studi

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

Mahasiswa

**MUH ALMADHANI ASRI**
NIM. 218280072

**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 A4 berwarna Hijau muda dan dicetak timbal balik