BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Pengujian pada aplikasi VRGamelanSaron dilakukan dengan menggunakan metode *Black Box testing* dengan teknik *Equivalence Partitions*. Dengan menggunakan *Black Box testing*, kita dapat menguji setiap fungsionalitas pada aplikasi tersebut dan melihat apakah *output* dari setiap fungsi aplikasi sudah sesuai atau belum. Setelah dilakukan pengujian, dapat disimpulkan bahwa setiap fungsi yang terdapat pada aplikasi VRGamelanSaron sudah dapat beroperasi sesuai dengan ekspektasi yang telah ditentukan. Namun, untuk meningkatkan akurasi dari setiap area *test case* yang ada di dalam aplikasi VRGamelanSaron, penggunaan teknik pengujian alternatif dapat digunakan bersamaan dengan metode yang terdapat pada penelitian ini.

5.2. Saran

Berdasarkan hasil yang telah diperoleh dari bab sebelumnya, terdapat beberapa saran yang dapat dilakukan untuk penelitian dengan topik sejenis kedepannya dapat berjalan dengan lebih baik. Beberapa saran tersebut yaitu:

1. Pengujian dapat dilakukan dengan menggabungkan metode *Black Box* dengan metode lainnya seperti *White Box* agar dapat lebih mengetahui bagian internal pada aplikasi / *software* yang diuji, serta juga mendapatkan hasil yang lebih detail.
2. Pengujian dapat dilakukan dengan menggunakan perangkat VR dengan opsi dan variasi yang berbeda untuk mengetahui perbedaan hasil cara kerja aplikasi yang diuji.

DAFTAR PUSTAKA

[1] Ahmad Triaji, "Pembuatan Aplikasi Augmented Reality Sebagai Media Pengenalan Alat Musik Gamelan Jawa Berbasis Android," *J. Multi Media dan IT*, vol. 5, no. 2, Dec. 2021, doi: https://doi.org/10.46961/jommit.v5i2.446.

[2] pengelola web kemdikbud, "Gamelan Jadi Warisan Budaya Dunia, Mendikbudristek Sampaikan Apresiasi Kepada Pegiat Budaya ," *kemdikbud.go.id*, Dec. 16, 2021.

[3] Redaktur, "Jadi Warisan Budaya Dunia, Gamelan Yang Terlupakan Pewarisnya," *inijabar.com*, Mar. 22, 2022.

[4] E. Irmania, A. Trisiana, and C. Salsabila, "Upaya mengatasi pengaruh negatif budaya asing terhadap generasi muda di Indonesia," *Din. Sos. Budaya*, vol. 23, no. 1, pp. 148–160, 2021, [Online]. Available: http://journals.usm.ac.id/index.php/jdsb

[5] Y. K. Dwivedi *et al.*, "Metaverse beyond the hype: Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy," *Int. J. Inf. Manage.*, vol. 66, Oct. 2022, doi: 10.1016/j.ijinfomgt.2022.102542.

[6] S. G. Fussell and D. Truong, "Using virtual reality for dynamic learning: an extended technology acceptance model," *Virtual Real.*, vol. 26, no. 1, pp. 249–267, Mar. 2022, doi: 10.1007/s10055-021-00554-x.

[7] A. Kurniawan, A. Maulana, V. R. Sukma, W. Keumala, and A. Saifudin, "Jurnal Teknologi Sistem Informasi dan Aplikasi Pengujian Black Box pada Aplikasi Penjualan Berbasis Web Menggunakan Metode Equivalents Partitions (Studi Kasus: PT Arap Store)," vol. 3, no. 1, pp. 2654–4229, 2020, [Online]. Available: http://openjournal.unpam.ac.id/index.php/JTSI50

[8] M. E. Khan and F. Khan, "A Comparative Study of White Box, Black Box and Grey Box Testing Techniques," 2012. [Online]. Available:

www.ijacsa.thesai.org

[9] U. Nugraha and T. Sianturi, “Blackbox Testing On E-Commerce System Web-Based Evermos (Feature: Registration Experiment & Revamp),” 2021.

[10] M. Albarka Umar and C. Zhanfang, “A Comparative Study of Dynamic Software Testing Techniques,” *Int. J. Adv. Netw. Appl.*, pp. 4575–4584, 2020.

[11] D. Debiyanti, S. Sutrisna, B. Budrio, A. K. Kamal, and Y. Yulianti, “Pengujian Black Box pada Perangkat Lunak Sistem Penilaian Mahasiswa Menggunakan Teknik Boundary Value Analysis,” *J. Inform. Univ. Pamulang*, vol. 5, no. 2, p. 162, 2020, doi: 10.32493/informatika.v5i2.5446.

[12] M. Syah Anwar Kesuma Jaya, P. Gumilang, Y. Philipus Andersen, and dan Teti Desyani, “Pengujian Black Box pada Aplikasi Sistem Penunjang Keputusan Seleksi Calon Pegawai Negeri Sipil Menggunakan Teknik Equivalence Partitions,” vol. 4, no. 4, pp. 2622–4615, 2019, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika

[13] E. H. Kusuma Dewi, I. S. Pratama, A. S. Putera, and C. Carudin, “Black Box Testing pada Aplikasi Pencatatan Peminjaman Buku Menggunakan Boundary Value Analysis,” *STRING (Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 6, no. 3, p. 315, 2022, doi: 10.30998/string.v6i3.11958.

[14] F. Lumban Gaol, H. Leslie Hendric Spits Warnars, and B. Soewito, “SOFTWARE TESTING BY USING THE BLACK-BOX METHOD AND THE EQUIVALENCE PARTITION TECHNIQUE TO PREDICT THE ACCURACY OF THE NEURAL NETWORK BASE,” 2020. [Online]. Available: https://ijact.in

[15] M. Sholeh, I. Gisfas, Cahiman, and M. A. Fauzi, “Black Box Testing on ukmbantul.com Page with Boundary Value Analysis and Equivalence Partitioning Methods,” in *Journal of Physics: Conference Series*, Mar. 2021, vol. 1823, no. 1. doi: 10.1088/1742-6596/1823/1/012029.

[16] G. Indah Marthasari *et al.*, "Pengujian Website Infotech Menggunakan Teknik Black-Box Decision Table," *J. Inform. Univ. Pamulang*, vol. 7, no. 1, pp. 115–119, 2022, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika

[17] D. S. Taley, "Comprehensive Study of Software Testing Techniques and Strategies : A Review," vol. 9, no. 08, pp. 817–822, 2020.

[18] M. A. Jamil, M. Arif, N. S. A. Abubakar, and A. Ahmad, "Software testing techniques: A literature review," *Proc. - 6th Int. Conf. Inf. Commun. Technol. Muslim World, ICT4M 2016*, no. November, pp. 177–182, 2017, doi: 10.1109/ICT4M.2016.40.

[19] A. B. Brohi, P. K. Butt, and S. Zhang, *Software Quality Assurance : Tools and Techniques*, vol. 1. Springer International Publishing, 2019. doi: 10.1007/978-3-030-24900-7.

[20] S. Thapliyal and R. Bahuguna, "A Newly Proposed Ambidextrous Software Testing Model Based on Conventional Black Box Testing Strategy Using the Applications of Gaussian Distribution," *Int. Res. J. Innov. Eng. Technol.*, vol. 5, no. 8, pp. 94–101, 2021, doi: 10.47001/IRJIET/2021.508016.

[21] M. E. Khan, "Different approaches to white box testing technique for finding errors," *Int. J. Softw. Eng. its Appl.*, vol. 5, no. 3, pp. 1–14, 2011, doi: 10.5121/ijsea.2011.2404.

[22] D. Yulistyanti, T. Y. Akhirina, T. Afrizal, A. Paramita, and N. Farkhatin, "Testing Learning Media for English Learning Applications Using BlackBox Testing Based on Equivalence Partitions," *Scope J. English Lang. Teach.*, vol. 6, no. 2, p. 73, 2022, doi: 10.30998/scope.v6i2.12845.

[23] E. Novalia and A. Voutama, "Black Box Testing dengan Teknik Equivalence Partitions Pada Aplikasi Android M-Magazine Mading Sekolah," vol. 11, no. 11, pp. 23–34, 2022.

[24] A. Ernawati, D. Kurnia, and A. Hindasyah, "Sistem Informasi Quality

Assurance Proses Produksi Menggunakan Metode Agile Berbasis Web," *Inform. Univ. Pamulang*, vol. 6, no. 3, pp. 491–497, 2021, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika/article/view/10272/pdf

[25] K. Sistem *et al.*, "Sampul Software Quality Assurance pada Perusahaan Pengembang Perangkat Lunak Skala Kecil dan Menengah," 2020.

[26] R. P. Anggara, P. Musa, Sri Lestari, and S. Widodo, "Application of Electronic Learning by Utilizing Virtual Reality (VR) and Augmented Reality (AR) Methods in Natural Sciences Subjects (IPA) in Elementary School Students Grade 3," *JTP - J. Teknol. Pendidik.*, vol. 23, no. 1, pp. 58–69, 2021, doi: 10.21009/jtp.v23i1.20203.

[27] H. A. Musril, J. Jasmienti, and M. Hurrahman, "Implementasi Teknologi Virtual Reality Pada Media Pembelajaran Perakitan Komputer," *J. Nas. Pendidik. Tek. Inform.*, vol. 9, no. 1, p. 83, 2020, doi: 10.23887/janapati.v9i1.23215.

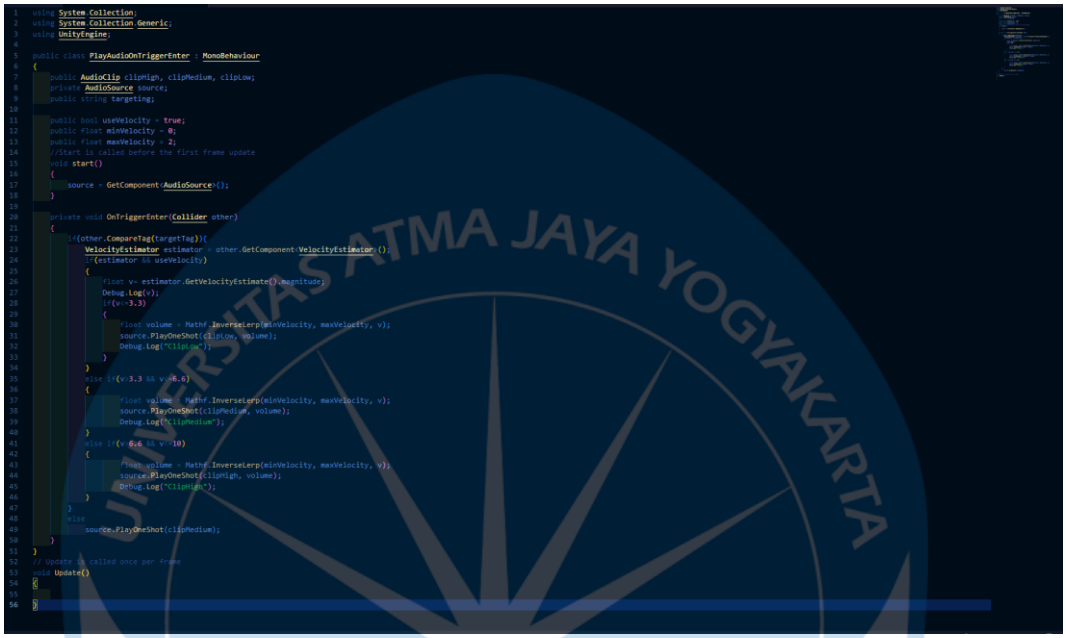
TABEL REVISI

No	Revisi	Halaman
1	Mengganti simbol “<=” Menjadi “≤”	Telah ditambahkan pada halaman : 13-14 ,19 ,& 21-23
2	Mengganti kata “*Scenario”* menjadi “skenario”	Telah ditambahkan pada halaman : ix, 18-21, 25-26
3	Merapikan Daftar Pustaka	Telah ditambahkan pada halaman : 32-35

Lampiran

Lampiran 1 EventListener

Bagian 1



```
using System.Collection;
using System.Collection.Generic;
using UnityEngine;

public class PlayAudioOnTriggerEnter : MonoBehaviour
{
    public AudioClip clipHigh, clipMedium, clipLow;
    private AudioSource source;
    public string targeting;

    public bool useVelocity = true;
    public float minVelocity = 0;
    public float maxVelocity = 2;
    //Start is called before the first frame update
    void start()
    {
        source = GetComponent<AudioSource>();
    }

    private void OnTriggerEnter(Collider other)
    {
        if(other.CompareTag(targetTag)){
            VelocityEstimator estimator = other.GetComponent<VelocityEstimator>();
            if(estimator && useVelocity)
            {
                float v= estimator.GetVelocityEstimate().magnitude;
                Debug.Log(v);
                if(v<=3.3)
                {
                    float volume = Mathf.InverseLerp(minVelocity, maxVelocity, v);
                    source.PlayOneShot(clipLow, volume);
                    Debug.Log("ClipLow");
                }
            }
            else if(v 3.3 && v <6.6)
            {
                    float volume = Mathf.InverseLerp(minVelocity, maxVelocity, v);
                    source.PlayOneShot(clipMedium, volume);
                    Debug.Log("ClipMedium");
            }
            else if(v 6.6 && v <10)
            {
                    float volume = Mathf.InverseLerp(minVelocity, maxVelocity, v);
                    source.PlayOneShot(clipHigh, volume);
                    Debug.Log("ClipHigh");
            }
        }
        else
            source.PlayOneShot(clipMedium);
    }
}
// Update is called once per frame
void Update()
{

}
```

Lampiran 2 Velocity Estimator

Bagian 1

```
using UnityEngine;
using System.Collections;

namespace Valve.VR.InteractionSystem
{
	//-------------------------------------------------------------------------
	public class VelocityEstimator : MonoBehaviour
	{
		[Tooltip( "How many frames to average over for computing velocity" )]
		public int velocityAverageFrames = 5;
		[Tooltip( "How many frames to average over for computing angular velocity" )]
		public int angularVelocityAverageFrames = 11;

		public bool estimateOnAwake = false;

		private Coroutine routine;
		private int sampleCount;
		private Vector3[] velocitySamples;
		private Vector3[] angularVelocitySamples;


		//-------------------------------------------------
		public void BeginEstimatingVelocity()
		{
			FinishEstimatingVelocity();

			routine = StartCoroutine( EstimateVelocityCoroutine() );
		}


		//-------------------------------------------------
		public void FinishEstimatingVelocity()
		{
			if ( routine != null )
			{
				StopCoroutine( routine );
				routine = null;
```

Bagian 2

```
        }
    }


    //------------------------------------------------
    public Vector3 GetVelocityEstimate()
    {
        // Compute average velocity
        Vector3 velocity = Vector3.zero;
        int velocitySampleCount = Mathf.Min( sampleCount, velocitySamples.Length );
        if ( velocitySampleCount != 0 )
        {
            for ( int i = 0; i < velocitySampleCount; i++ )
            {
                velocity += velocitySamples[i];
            }
            velocity *= ( 1.0f / velocitySampleCount );
        }

        return velocity;
    }


    //------------------------------------------------
    public Vector3 GetAngularVelocityEstimate()
    {
        // Compute average angular velocity
        Vector3 angularVelocity = Vector3.zero;
        int angularVelocitySampleCount = Mathf.Min( sampleCount, angularVelocitySamples.Length );
        if ( angularVelocitySampleCount != 0 )
        {
            for ( int i = 0; i < angularVelocitySampleCount; i++ )
            {
                angularVelocity += angularVelocitySamples[i];
            }
            angularVelocity *= ( 1.0f / angularVelocitySampleCount );
        }
```

Bagian 3

```
        return angularVelocity;
    }


    //------------------------------------------------
    public Vector3 GetAccelerationEstimate()
    {
        Vector3 average = Vector3.zero;
        for ( int i = 2 + sampleCount - velocitySamples.Length; i < sampleCount; i++ )
        {
            if ( i < 2 )
                continue;

            int first = i - 2;
            int second = i - 1;

            Vector3 v1 = velocitySamples[first % velocitySamples.Length];
            Vector3 v2 = velocitySamples[second % velocitySamples.Length];
            average += v2 - v1;
        }
        average *= ( 1.0f / Time.deltaTime );
        return average;
    }


    //------------------------------------------------
    void Awake()
    {
        velocitySamples = new Vector3[velocityAverageFrames];
        angularVelocitySamples = new Vector3[angularVelocityAverageFrames];

        if ( estimateOnAwake )
        {
            BeginEstimatingVelocity();
        }
    }
```

Bagian 4

```
//------------------------------------------------
private IEnumerator EstimateVelocityCoroutine()
{
        sampleCount = 0;

        Vector3 previousPosition = transform.position;
        Quaternion previousRotation = transform.rotation;
        while ( true )
        {
                yield return new WaitForEndOfFrame();

                float velocityFactor = 1.0f / Time.deltaTime;

                int v = sampleCount % velocitySamples.Length;
                int w = sampleCount % angularVelocitySamples.Length;
                sampleCount++;

                // Estimate linear velocity
                velocitySamples[v] = velocityFactor * ( transform.position - previousPosition );

                // Estimate angular velocity
                Quaternion deltaRotation = transform.rotation * Quaternion.Inverse( previousRotation );

                float theta = 2.0f * Mathf.Acos( Mathf.Clamp( deltaRotation.w, -1.0f, 1.0f ) );
                if ( theta > Mathf.PI )
                {
                        theta -= 2.0f * Mathf.PI;
                }

                Vector3 angularVelocity = new Vector3( deltaRotation.x, deltaRotation.y, deltaRotation.z );
                if ( angularVelocity.sqrMagnitude > 0.0f )
                {
                        angularVelocity = theta * velocityFactor * angularVelocity.normalized;
                }

                angularVelocitySamples[w] = angularVelocity;

                        previousPosition = transform.position;
                        previousRotation = transform.rotation;
                }
        }
}
}
```

Lampiran 3 Tampilan

Bagian 1 Tampilan Pilihan Main Menu

Bagian 2 Tampilan Environment