BAB IV
HASIL DAN PEMBAHASAN

4.1 Perancangan Game

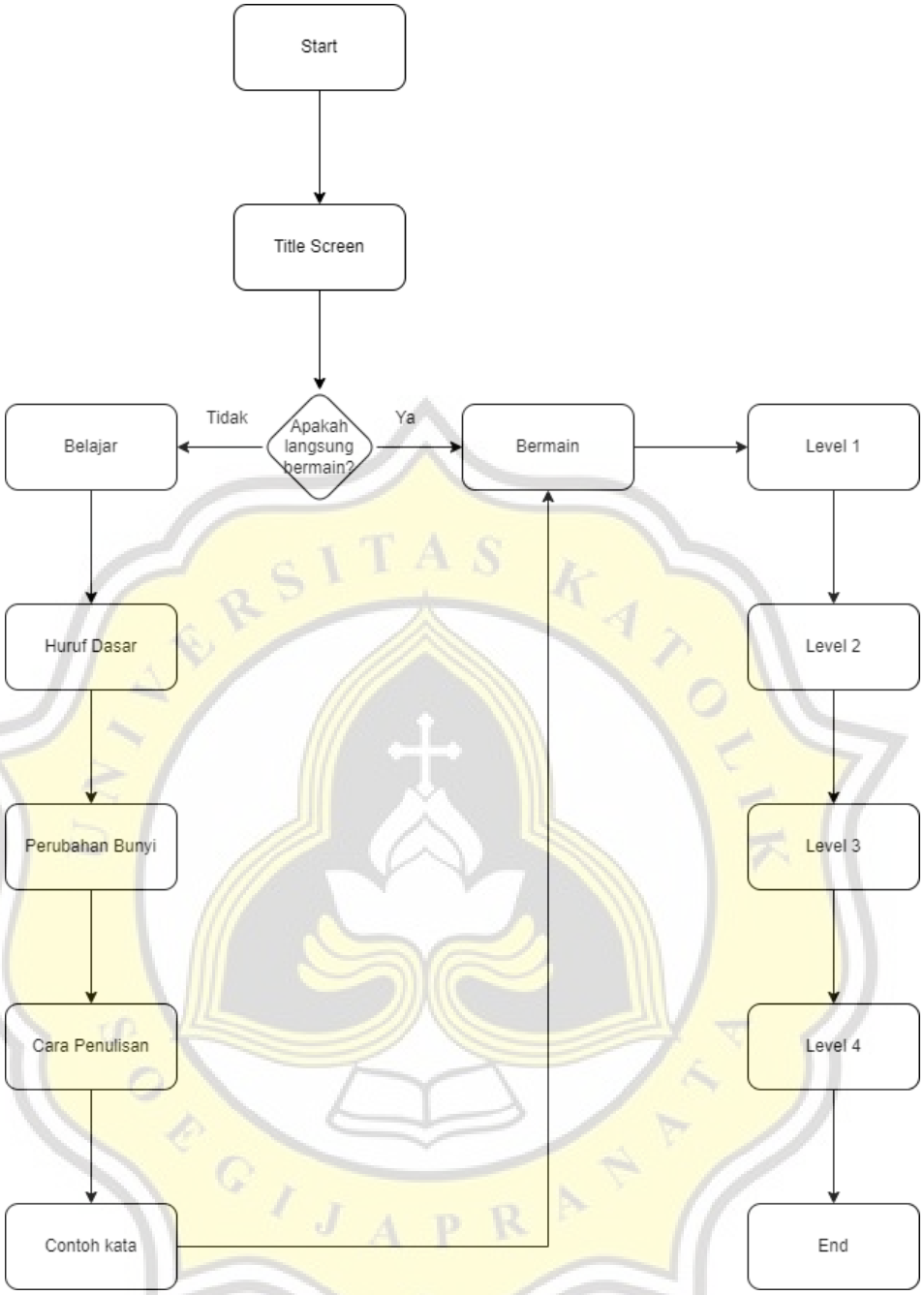
"Kaganga Games" adalah game yang dirancang untuk pengenalan aksara Rejang atau sering dikenal juga dengan aksara Kaganga. Game ini menggunakan perspektif 2D yang didalamnya terdapat 4 minigame dengan masing-masing memiliki unsur pengenalan terhadap aksara kaganga, dengan mengumpulkan huruf kaganga saja dan menghindari huruf dan rintangan lainnya. Supaya "Kaganga Games" ini menarik untuk dimainkan, maka diperlukan tantangan saat memainkan seperti semakin lama dapat bertahan maka tingkat kesulitan menjadi semakin tinggi. Game ini juga dilengkapi dengan pilihan awal selain bermain yakni untuk belajar, didalam belajar dijelaskan tentang dasar-dasar huruf kaganga itu sendiri seperti semua huruf pokok, perubahan bunyi, cara penulisan dan contoh kata.

4.1.1 Games Design

"Kaganga Games" merupakan game yang memiliki 4 gameplay yang berbeda didalamnya, yaitu "Platformer", "Side Scrolling ", "Top Down" dan "Trivia". Rata-rata gameplay dari game ini memilih aksara kaganga dan menghindari rintangan lainnya serta dilengkapi kuis yang berhubungan dengan huruf kaganga.

4.1.2 Perancangan Gameplay

Sebelum membuat game Kaganga Games maka hal pertama yang perlu untuk dibuat adalah rancangan agar game yang dibuat tidak melenceng dari yang ingin dicapai. Untuk rancangan gameplay dari Kaganga Games sendiri seperti pada Gambar 4.1 di bawah ini.



Gambar 4.1 Perancangan Gameplay

Perancangan Gameplay awalnya hanya akan membuat sistem belajar dan bermain yang berfokus pada game trivia saja namun supaya game ini lebih menarik untuk dimainkan maka game lebih divariasi dengan 4 gameplay yang berbeda, yaitu “Platformer”, “Side Scrolling Endless”, “Top Down Endless” dan “Trivia”. Pada

bagian belajar huruf kaganga juga terdapat sub bagian yakni “Huruf Dasar”, “Peubahan Bunyi”, “Cara Penulisan” dan “Contoh Kata”.

4.2 Implementasi Pembuatan Game

Untuk implementasi pembuatan game ini akan dijelaskan tentang setiap gameplay yang ada mulai dari tutorial hingga kondisi menyelesaikan game sesuai dengan perancangan sebelumnya.

4.2.1 Implementasi Tutorial

a. Platformer (Level 1)

Pada level pertama sebelum memulai game akan diperlihatkan cara bermain dari platformer seperti pada Gambar 4.2 yang menampilkan tentang penjelasan bagaimana cara bermain dan fungsi dari setiap tombol yang ada seperti tombol gerak kiri dan kanan, tombol lompat pada gameplay “Platformer”. Panel tutorial ini akan hilang dan memulai game saat pemain menekan tombol “oke”.

Gambar 4.2 Tampilan Tutorial Level 1

b. Side Scrolling (Level 2)

Selanjutnya pada level 2 juga diberikan tampilan cara bermain seperti di Gambar 4.3 yang menjelaskan tentang cara bermain dan fungsi dari setiap tombol yang ada di level 2. Berbeda dengan level sebelumnya pada level ini hanya ada tombol gerak ke kiri dan kanan bergerak kesamping sesuai sumbu x tanpa adanya tombol loncat.

Gambar 4.3 Tampilan Tutorial Level 2

c. Top Down (Level 3)

Pada Level 3 juga diberikan keterangan cara bermain seperti pada Gambar 4.4 menampilkan tentang penjelasan cara bermain dan fungsi dari setiap tombol yang ada di level 3. Pada level 3 ini tombol yang digunakan adalah tombol gerak ke atas dan bawah yang membuat karakter bergerak ke atas dan ke bawah sesuai sumbu y.

Gambar 4.4 Tampilan Tutorial Level 3

4.2.2 Implementasi Gameplay

a. Platformer (Level 1)

Sebelum memulai game pemain memilih terlebih dahulu level yang ingin dimainkan dapat dilihat seperti gambar di bawah ini. Pada awal permainan hanya ada level 1 yang terbuka dan bisa dimainkan seperti pada Gambar 4.5 hanya level 1 yang bisa dipilih dan untuk level lainnya masih terkunci. Untuk bisa membuka level selanjutnya maka harus menyelesaikan level ini terlebih dahulu.

Gambar 4.5 Tampilan Menu Hanya Level 1 yang Terbuka

Untuk tampilan gameplaynya ada pada gambar di bawah ini. Setelah membaca cara bermain maka kontrol akan terbuka dan pemain bisa memulai menggerakkan karakter untuk memulai bermain mengumpulkan huruf kaganga yang terpisah dengan melewati rintangan yang ada seperti pada Gambar 4.6. Pada level ini juga terdapat sistem *respawn* ketika terkena musuh ataupun terjatuh ke dalam jurang. Pemain yang mati akan hidup kembali pada tanda merah yang terakhir kali disentuh sedangkan ketika semua nyawa telah habis dan pemain kembali terjatuh ataupun terkena musuh maka pemain dinyatakan kalah dan bisa mengulang dari awal permainan.

Gambar 4.6 Gameplay Level 1

Untuk menggerakkan karakter dan semua situasi pada penjelasan sebelumnya maka diperlukan skrip kode seperti pada gambar 4.7.

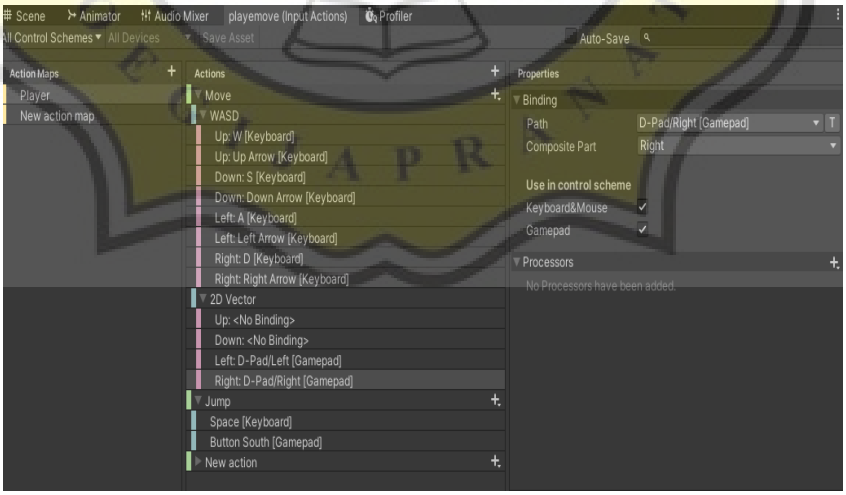
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.InputSystem;

public class PlayerController : MonoBehaviour
{
    public Rigidbody2D theRB;

    public float moveSpeed, jumpForce;

    public Transform groundPoint;
    public LayerMask whatIsGround;

    private bool isGrounded;

    public Animator anim;

    private float inputX;

    // Start is called before the first frame update
    void Start()
    {
        Time.timeScale = 0;
    }

    // Update is called once per frame
    void Update()
    {

        theRB.velocity = new Vector2(inputX * moveSpeed,
theRB.velocity.y);

        //check if on the ground
        isGrounded = Physics2D.OverlapCircle(groundPoint.position,
.2f, whatIsGround);

        anim.SetFloat("speed", Mathf.Abs(theRB.velocity.x));
        anim.SetBool("isGrounded", isGrounded);
      if(theRB.velocity.x > 0f)
        {
            transform.localScale = Vector3.one;
        } else if(theRB.velocity.x < 0f)
        {
            transform.localScale = new Vector3(-1f, 1f, 1f);
        }

      }
```

Gambar 4.7 Skrip Menggerakkan Karakter

```
    // gerak karakter

    public void Move(InputAction.CallbackContext context)
    {
        inputX = context.ReadValue<Vector2>().x;
    }

    // lompat karakter
    public void Jump(InputAction.CallbackContext context)
    {
        if (context.performed && isGrounded)
        {
            theRB.velocity = new Vector2(theRB.velocity.x,
jumpForce);
        }

    }
}
```

Gambar 4.7 Skrip Menggerakkan Karakter (lanjutan)

Setelah membuat skrip untuk menggerakkan karater seperti pada Gambar 4.7, tambahkan fungsi tombol pada *input action manager* yang akan digunakan seperti Gambar 4.8 menunjukkan penggunaan tombol yang berfungsi pada keyboard sebagai pengujian dan tombol pada layar yang digunakan oleh pengguna android.



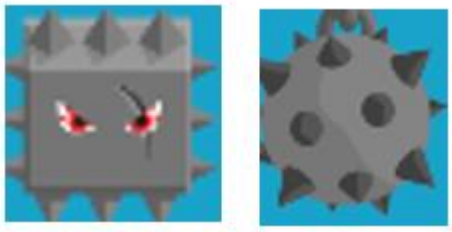
Gambar 4.8 Pengaturan *Input Action Manager*

Gambar 4.9 menjelaskan tentang skrip yang bisa membuat rintangan menjadi berpindah posisi agar dapat menghalau pemain menyelesaikan level dengan mudah. Di bawah ini adalah gambar dari rintangan yang dapat menghalangi pemain pada level pertama.

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

public class Mace : MonoBehaviour
{

    public float speed = 0.8f;
    public float range = 3;

    float startingY;
    int dir = 1;
    // Start is called before the first frame update
    void Start()
    {
        startingY = transform.position.y;
    }
        // Update is called once per frame
    void Update()
    {
        transform.Translate(Vector2.up * speed * Time.deltaTime *
dir);
        if (transform.position.y < startingY || transform.position.y
> startingY + range)
            dir *= -1;
    }
}
```

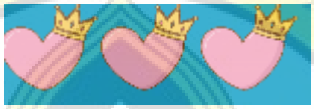
Gambar 4.9 Skrip Menggerakkan Rintangan

Gambar 4.10 menunjukkan rintangan yang akan dihadapi pada level pertama. Rintangan berguna untuk mempersulit permainan, pada level ini rintangan akan bergerak naik turun dan kiri kanan yang mana jika terkena pemain maka nyawa dari pemain berkurang 1.



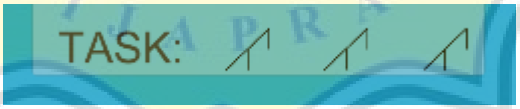
Gambar 4.10 Rintangan Bergerak

Setiap awal permainan nyawa pemain diberi kesempatan 3 kali seperti terlihat pada Gambar 4.11, jika nyawa sudah habis sebelum menyelesaikan level ini maka tidak bisa membuka level berikutnya.



Gambar 4.11 Tampilan Nyawa Dimiliki

Pemain dapat dikatakan menyelesaikan level jika sudah melengkapi tugas seperti gambar di bawah ini. Gambar 4.12 menunjukkan tampilan misi yang harus dikerjakan untuk bisa menyelesaikan level pertama pemain harus mengumpulkan huruf kaganga yang telah disebar di berbagai daerah pada map.



Gambar 4.12 Tampilan Misi yang Harus Diselesaikan

Gambar 4.13 berisi tentang perintah jika pemain menyentuh musuh maka nyawa yang dimiliki akan berkurang serta akan dipindah menuju *safe point* tapi ketika tidak ada sisa nyawa lagi pemain akan dinyatakan kalah sedangkan jika

pemain menyentuh finish maka akan membuka level selanjutnya. Sistem akan mendeteksi sentuhan pemain menggunakan *collider 2D* pada Unity Engine. Kemudian benda yang telah disebarkan supaya dapat di ambil oleh pemain adalah dengan menggunakan skrip di bawah ini.

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

public class PlayerCollision : MonoBehaviour
{
    public static Vector2 lastCheckPointPos = new Vector2(-3, 0);
    private void OnCollisionEnter2D(Collision2D collision)

    {
        if(collision.transform.tag == "Enemy")
        {
            HeartSkrip.health -= 1;

GameObject.FindGameObjectWithTag("Player").transform.position =
lastCheckPointPos;

        }

  if (collision.transform.tag == "Finish")
        {
            PlayerManager.isFinish = true;
            gameObject.SetActive(false);

        }

    }
}
```

Gambar 4.13 Skrip Menyentuh Benda Lain

Gambar 4.14 menjelaskan tentang skrip yang mengatur saat pemain menemukan salah satu huruf kaganga maka huruf yang ada pada *task* akan berkurang satu persatu, saat huruf pada *task* sudah habis maka tanda *finish* akan muncul di map dan jika menyentuhnya level akan selesai dan membuka level selanjutnya.

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

public class Taskquest1 : MonoBehaviour
{
    public GameObject ini1, ini2, ini3;
    public static int ini;
    void Start()
    {
        ini = 3;
        ini1.gameObject.SetActive(true);
        ini2.gameObject.SetActive(true);
        ini3.gameObject.SetActive(true);
        Time.timeScale = 1;
    }
void Update()
    {
        if (ini > 3)
            ini = 3;

          switch (ini)
        {
            case 3:
                ini1.gameObject.SetActive(true);
                ini2.gameObject.SetActive(true);
                ini3.gameObject.SetActive(true);
                break;

case 2:
                ini1.gameObject.SetActive(true);
                ini2.gameObject.SetActive(true);
                ini3.gameObject.SetActive(false);
                break;
            case 1:
                ini1.gameObject.SetActive(true);
                ini2.gameObject.SetActive(false);
                ini3.gameObject.SetActive(false);
                break;
             case 0:
                ini1.gameObject.SetActive(false);
                ini2.gameObject.SetActive(false);
                ini3.gameObject.SetActive(false);
                PlayerManager.isComplete = true;
                break;
        }
    }
}
```

Gambar 4.14 Skrip Misi yang Harus Dikerjakan

Gambar 4.15 menunjukkan skrip yang digunakan untuk membuka level selanjutnya dengan penambahan satu level, jadi pada saat sedang memainkan level 2 dan menyelesaikan level tersebut maka akan membuka level selanjutnya yakni level 3.

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

public class LevelSkrip : MonoBehaviour
{
    // Start is called before the first frame update
    public void Pass()
    {
        int currentLevel = SceneManager.GetActiveScene().buildIndex;

        if(currentLevel >= PlayerPrefs.GetInt("levelsUnlocked"))
        {
            PlayerPrefs.SetInt("levelsUnlocked",currentLevel + 1);
        }
        Debug.Log("Level " + PlayerPrefs.GetInt("levelsUnlocked") +
"Unlocked");
    }
}
```

Gambar 4.15 Skrip Membuka Level Selanjutnya

b. Side Scrolling Endless (Level 2)

Setelah menyelesaikan level pertama maka pada level 2 yang sudah terbuka tampilannya sebagai berikut ini. Gambar 4.16 menunjukkan tampilan saat sudah menyelesaikan level pertama dan level kedua sudah terbuka sehingga bisa untuk dimainkan. Dan ketika sudah sekali terbuka maka level akan tetap terbuka selama belum direset gamenya dengan menggunakan *player prefs.*

Gambar 4.16 Tampilan Level 2 yang Terbuka

Untuk implementasi kode level yang seperti ini maka menggunakan kode seperti pada Gambar 4.17 mengatur level yang sedang terbuka agar level yang terbuka tetap sesuai dengan yang telah dipilih oleh pemain dan game pun tetap lancar dimainkan sehingga ditambahkan skrip menghapus semua progres untuk mengatur ulang level yang telah dicapai.

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

public class LevelManager : MonoBehaviour
{
    int levelsUnlocked;

    public Button[] buttons;
// Start is called before the first frame update
    void Start()
    {
        levelsUnlocked = PlayerPrefs.GetInt("levelsUnlocked", 1);
        for (int i = 0; i < buttons.Length; i++)
        {
            buttons[i].interactable = false;
        }
        for (int i = 0; i < levelsUnlocked; i++)
        {
            buttons[i].interactable = true;
        }
    }

    public void LoadLevel(int levelIndex)
    {
        SceneManager.LoadScene(levelIndex);
    }
    public void DeleteAll()
    {
        PlayerPrefs.DeleteAll();
    }
    // Update is called once per frame
    void Update()
    {

    }
}
```

Gambar 4.17 Skrip Mengatur Level Keseluruhan

Untuk tampilan gameplaynya adalah seperti pada Gambar 4.18 menunjukkan gameplay dari Side Scrolling Endless terdapat karakter yang dapat bergerak ke kiri dan kanan sesuai dengan tombol yang ditekan. Tugas pemain hanya mengumpulkan huruf kaganga dan yang menjadi rintangan adalah menghindari huruf selain kaganga itu sendiri. Untuk kode menggerakkan karakter disini sama

dengan level 1 hanya saja pemain dibuat tidak bisa melompat. Pemain dapat mengumpulkan 30 poin untuk membuka level selanjutnya tanpa adanya batasan waktu, namun jika semua nyawa telah habis makan pemain dinyatakan gagal dan level selanjutnya pun tidak dapat terbuka.

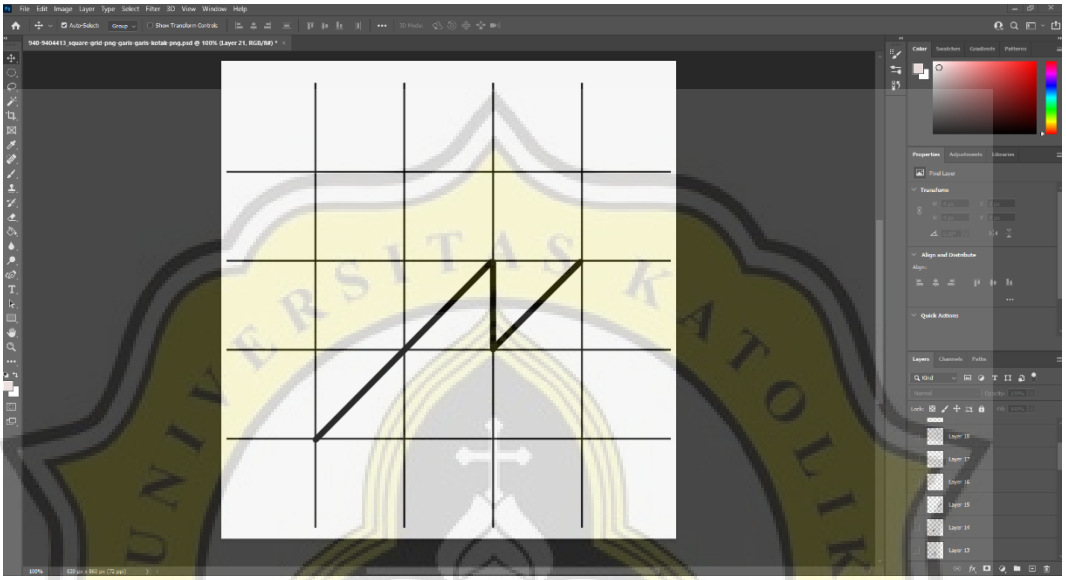
Gambar 4.18 Gameplay Level 2

Sedangkan untuk huruf-huruf yang turun secara acak baik huruf kaganga maupun huruf non-kaganga dari atas menggunakan kode seperti pada Gambar 4.19 berfungsi menurunkan objek yang merupakan huruf kaganga maupun bukan huruf kaganga secara acak dengan semakin lama pemain bertahan maka kecepetan memunculkan objeknya semakin cepat agar tingkat kesulitan pun menjadi lebih tinggi.

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

public class jatoh : MonoBehaviour
{
    [SerializeField] GameObject[] fruitPrefab;
    [SerializeField] float secondSpawn = 0.5f;
    [SerializeField] float minTras;
    [SerializeField] float maxTras;

   // public static int score;
    public float minSpawn;
    // Start is called before the first frame update
    void Start()
    {
        StartCoroutine(FruitSpawn());
        Time.timeScale = 0;
    }
IEnumerator FruitSpawn()
    {
        while(true)
        {
            var wanted = Random.Range(minTras, maxTras);
            var position = new Vector3(wanted,
transform.position.y);

            GameObject gameObject =
Instantiate(fruitPrefab[Random.Range(0, fruitPrefab.Length)],

            position, Quaternion.identity);
            yield return new WaitForSeconds(secondSpawn);
            Destroy(gameObject, 3f);
        }
    }
    void Update()
    {
        if(secondSpawn > minSpawn)
        secondSpawn -= 0.05f * Time.deltaTime;
       // if(score > 50)
        //{
        //    PlayerManager.isFinish = true;
//Time.timeScale = 0;
       // }
    }
}
```

Gambar 4.19 Skrip Menjatuhkan Benda

Untuk pembuatan objek menggunakan Adobe Photoshop seperti pada Gambar 4.20 menunjukkan contoh huruf kaganga yang dibuat menggunakan Adobe Photoshop yang kemudian diekspor dalam format PNG supaya nantinya bisa dimasukkan ke dalam Unity dan bisa digunakan di semua scene yang ada.



Gambar 4.20 Pembuatan Huruf Kaganga Menggunakan Photoshop

Untuk objek yang menjadi rintangan pada level ini ada pada Gambar 4.21 merupakan contoh huruf non-kaganga yang harus dihindari, jika terkena maka akan mengurangi nyawa yang dimiliki oleh pemain untuk skrip yang digunakan pada setiap karakter non-kaganga adalah sebagai berikut.

Gambar 4.21 Simbol Lain

Pada Gambar 4.22 juga diberikan fungsi *Destroy* yang berguna untuk menghancurkan objek yang telah turun kemudian bersentuhan dengan tanah. Menghancurkan objek ini berguna supaya tidak adanya penumpukan objek sehingga game pun bisa dimainkan dengan lancar.

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

public class Enemies : MonoBehaviour
{
    public GameObject effect;

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.transform.tag == "Player")
        {
            Instantiate(effect, transform.position,
Quaternion.identity);
            HeartSkrip.health -= 1;
            Destroy(gameObject);

        }
      if (collision.transform.tag == "tanah")
        {
            Destroy(gameObject);

        }
    }
}
```

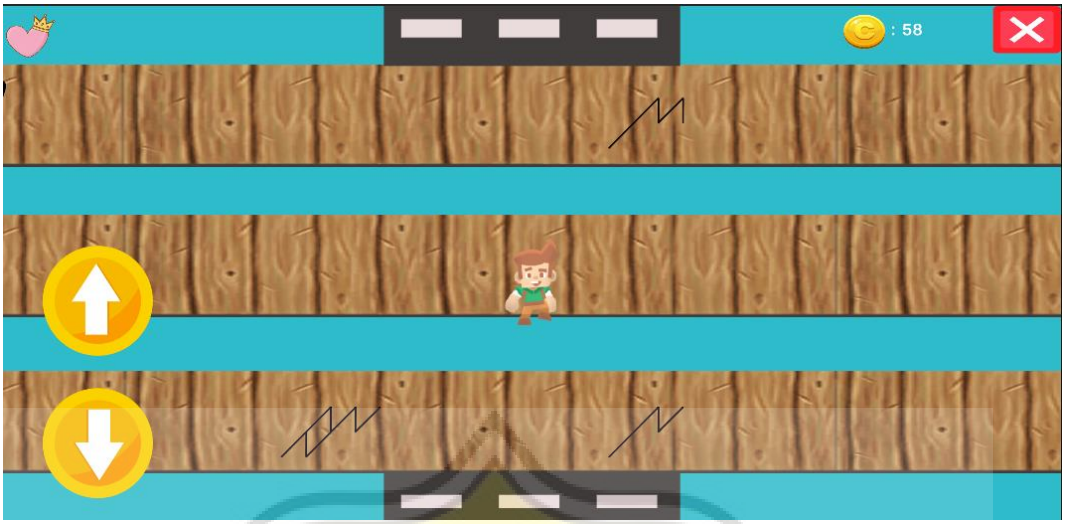
Gambar 4.22 Skrip Mengatur Huruf Non-Kagang

c. Top Down Endless (Level 3)

Setelah melewati level 2 maka level 3 pun dapat tebuka tampilannya akan berubah seperti pada Gambar 4.23 menunjukkan tampilan game setelah menyelesaikan level 2 sehingga pada level 3 ini bisa untuk dimainkan. Terjadi perubahan disini yakni pada level 3 yang sebelumnya terkunci sudah terbuka dan bisa untuk dimainkan namun pada level 4 masih terkunci dan belum bisa untuk dimainkan.

Gambar 4.23 Tampilan Level 3 yang Sudah Terbuka

Sedangkan untuk gameplay dari level 3 ini sendiri ketika dimainkan maka akan muncul tampilan seperti pada gambar 4.24, untuk sistem permainannya pun cukup mudah yaitu dengan bergerak ke atas dan ke bawah mengumpulkan huruf kaganga dan menghindari huruf selain huruf kaganga.



Gambar 4.24 Tampilan Level 3

Skrip yang digunakan agar tercipta kondisi seperti ini adalah seperti pada Gambar 4.25 adalah skrip dengan kode yang berguna menggerakkan karakter supaya bisa ke atas dan ke bawah. Sedangkan untuk memunculkan objek berupa huruf kaganga yaitu berada pada kode berikut.

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

public class ControlPlayerNew : MonoBehaviour
{
    private NewInput playerInput;
    private Rigidbody2D rb;

    [SerializeField] private float speed = 10f;

    private void Awake()
    {
        playerInput = new NewInput();
        rb = GetComponent<Rigidbody2D>();
    }

       private void OnEnable()
    {
        playerInput.Enable();

    }


     private void OnDisable()
    {
        playerInput.Disable();

    }
    // Update is called once per frame
    void FixedUpdate()
    {
        Vector2 moveInput =
playerInput.Movement.Move.ReadValue<Vector2>();
        rb.velocity = moveInput * speed;
    }
}
```

Gambar 4.25 Skrip Menggerakkan Karakter

Di dalam Gambar 4.26 skrip ini selain memunculkan benda juga mengatur waktu kemunculan benda. Kemunculan benda diatur agar semakin lama pemain bertahan maka objek yang akan dimunculkan menjadi semakin cepat muncul. Sehingga jarak antara objek satu dengan objek lainnya menjadi lebih rapat. Hal ini berguna untuk menambah tingkat kesulitan selama permainan.

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

public class Spawner : MonoBehaviour
{
    // Start is called before the first frame update
    public GameObject[] obstaclePatterns;

    private float timeBtwSpawn;
    public float startTimeBtwSpawn;
    public float decreaseTime;
    public float minTime = 0.65f;
    void Start()
    {
        Time.timeScale = 0;
    }

      private void Update()
    {
        if (timeBtwSpawn <= 0)
        {
            int rand = Random.Range(0, obstaclePatterns.Length);
            Instantiate(obstaclePatterns[rand], transform.position,
Quaternion.identity);
            timeBtwSpawn = startTimeBtwSpawn;
            if (startTimeBtwSpawn > minTime)
            {
                startTimeBtwSpawn -= decreaseTime;
            }
        }
        else
        {
            timeBtwSpawn -= Time.deltaTime;
        }
    }
}
```

Gambar 4.26 Skrip Memunculkan Benda

Dan yang terakhir adalah dengan penambahan skrip untuk menambah poin jika karakter menyentuh huruf kaganga itu sendiri seperti pada Gambar 4.27. Penambahan poin ini berguna untuk menghitung total poin yang didapat dan juga sebagai tolak ukur dalam membuka level selanjutnya. Untuk skrip nya ada pada skrip mengatur skor seperti di bawah ini dengan memanfaatkan *PlayerPrefs* pada

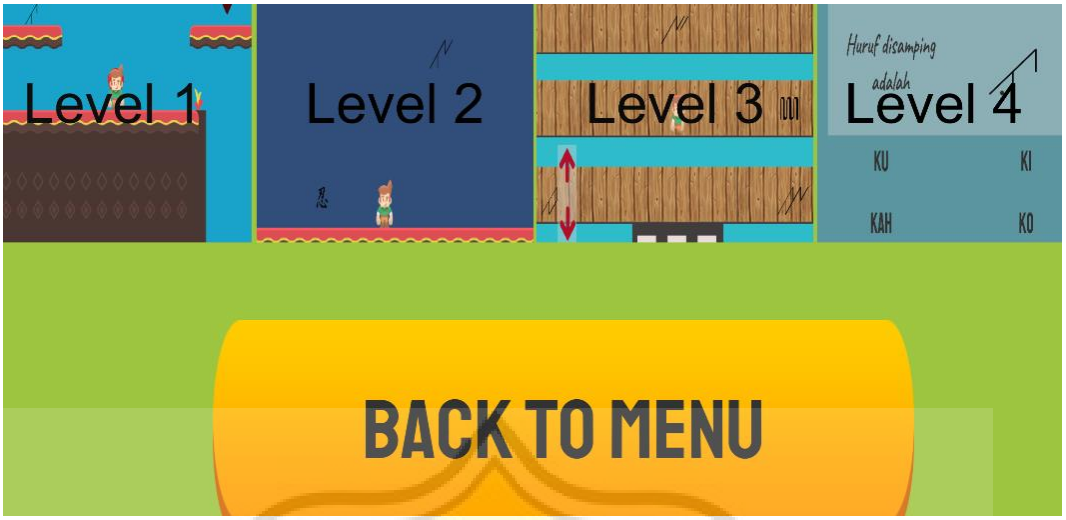
Unity yang berguna menyimpan suatu nilai sehingga tidak hilang saat memulai kembali game tersebut. Dan untuk skor di dalam game di atur oleh skrip di Pada gambar 4.27 diperlihatkan skrip yang mengatur skor dalam permainan dengan hanya menyentuh huruf kaganga yang telah diberikan *Collider 2d.*

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

public class Coin : MonoBehaviour
{
  private void OnTriggerEnter2D(Collider2D collision)
    {
        if(collision.transform.tag == "Player")
        {
            PlayerManager.numberOfCoins++;
            PlayerPrefs.SetInt("NumberOfCoins",
PlayerManager.numberOfCoins);
            Destroy(gameObject);
            Taskquest1.ini -= 1;
            HeartSkrip.score++;
        }
    }
}
```

Gambar 4.27 Skrip Mengatur Skor

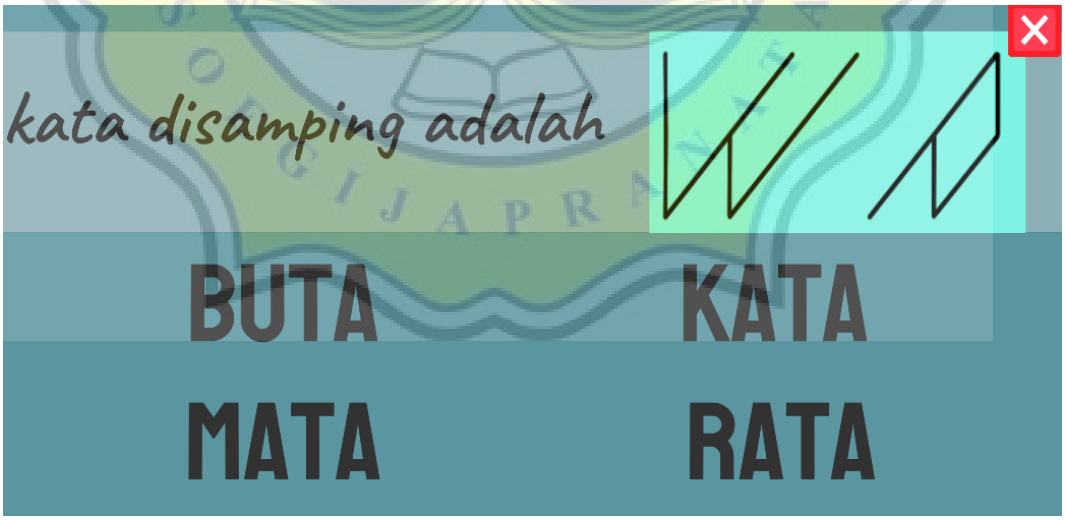
d. Trivia

Setelah menyelesaikan level 3 maka semua level akan terbuka dan dapat dimainkan. Dapat terlihat seperti pada Gambar 4.28 menunjukkan bahwa semua level sudah terbuka dan disini pemain bisa memilih level manapun tanpa harus mengulang dari awal.



Gambar 4.28 Tampilan Semua Level Terbuka

Setelah dimainkan akan muncul tampilan seperti pada Gambar 4.29. Pada game Trivia ini akan di suguhkan dengan 10 pertanyaan secara acak dari 15 pertanyaan yang telah dibuat. Setiap pertanyaan hanya ada 1 jawaban yang benar dan jika benar pemain akan mendapatkan 1 poin. Untuk contoh soal pada level trivia ini seperti pada Gambar 4.29.

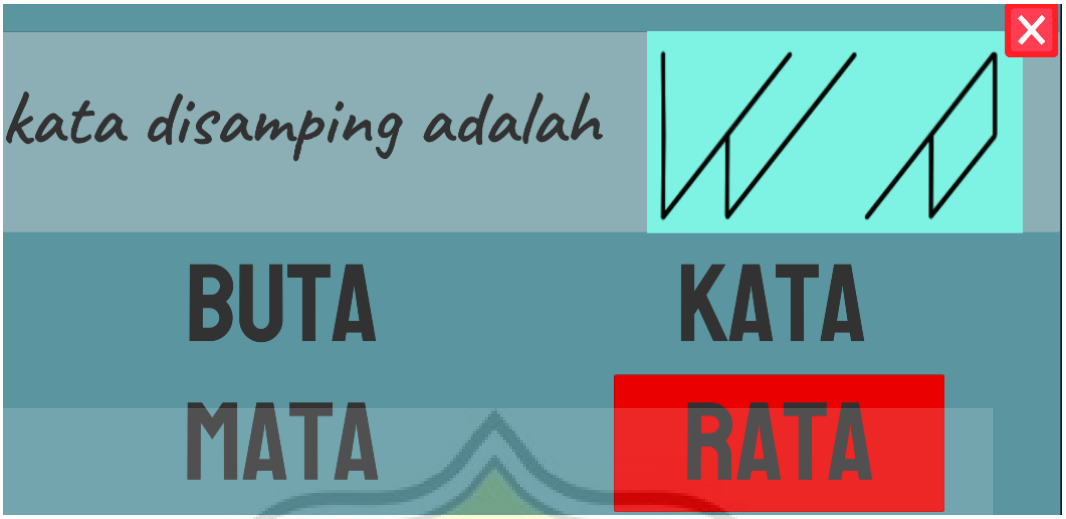


Gambar 4.29 Contoh Salah Satu Pertanyaan

Untuk contoh jawaban yang benar akan menampilkan seperti pada Gambar 4.30 menunjukkan jika pemain memilih jawaban yang benar maka pada jawaban yang dipilih akan berwarna hijau.

Gambar 4.30 Ketika Memilih Jawaban yang Benar

Sedangkan jika salah dalam memilih jawaban maka tampilan yang akan muncul adalah seperti pada Gambar 4.31 yang menunjukkan jika pemain memilih jawaban yang salah maka pada jawaban yang dipilih akan berwarna merah. Disaat pemain menjawab salah tidak akan mempengaruhi gameplay hanya saja mempengaruhi penilaian diakhir yang akan menunjukkan seberepa menguasai pemain dalam pengenalan huruf kaganga. Untuk membuat kondisi agar pertanyaan dapat muncul dan mengelola jawaban yang dipilih benar ataupun salah adalah dengan membuat skrip sebagai berikut.



Gambar 4.31 Ketika Memilih Jawaban yang Salah

Pada Gambar 4.32 berisi skrip yang mengatur tentang pertanyaan-pertanyaan yang akan dimunculkan pada panel dan mengatur jumlah pertanyaan yang akan ditampilkan secara acak. Selain itu pada Skrip Mengatur Pertanyaan ini mengatur setiap jawaban yang dipilih apakah benar atau salah serta memberikan jeda pada setiap pertanyaan agar tidak langsung menuju ke pertanyaan selanjutnya terlalu cepat.

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

public class QuizManager : MonoBehaviour
{
    public List<QuestionAndAnswers> QnA;
    public GameObject[] options;
    public int currentQuestion;

    public GameObject Quizpanel;
    public GameObject GoPanel;

    public Text QuestionTxt;


    public Text ScoreTxt;
    public Image GambarImg;
    int totalQuestions = 0;
    public int score;

private void Start()
    {
        Time.timeScale = 1;
        totalQuestions = QnA.Count;
        GoPanel.SetActive(false);
        generateQuestion();
    }
    public void retry()
    {

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
    }
    void GameOver()
    {
        Quizpanel.SetActive(false);
        GoPanel.SetActive(true);
        ScoreTxt.text = score + "/" + "10";
    }
    public void correct()
    {
        //when you are right
        score += 1;
        QnA.RemoveAt(currentQuestion);
        StartCoroutine(waitForNext());
    }
```

Gambar 4.32 Skrip Mengatur Pertanyaan

```
    public void wrong()
    {
        //when you answer wrong
        QnA.RemoveAt(currentQuestion);
        StartCoroutine(waitForNext());
    }

    IEnumerator waitForNext()
    {
        yield return new WaitForSeconds(1);
        generateQuestion();
    }

void SetAnswers()
    {
        for (int i = 0; i < options.Length; i++)
        {
            options[i].GetComponent<Image>().color =
options[i].GetComponent<AnswerSkrip>().startColor;
            options[i].GetComponent<AnswerSkrip>().isCorrect = false;

options[i].transform.GetChild(0).GetComponent<Text>().text =
QnA[currentQuestion].Answers[i];
            // GambarImg.GetComponent<Image>().sprite =
QnA[currentQuestion].overrideSprite;
            if (QnA[currentQuestion].CorrectAnswer == i+1)
            {
                options[i].GetComponent<AnswerSkrip>().isCorrect =
true;
            }
        }
    }
    void generateQuestion()
    {
        if(QnA.Count > 5)
        {
            currentQuestion = Random.Range(0, QnA.Count);

            QuestionTxt.text = QnA[currentQuestion].Question;
            GambarImg.sprite = QnA[currentQuestion].overrideSprite;
            SetAnswers();
        }
        else
        {
            Debug.Log("Out of Questions");
            GameOver();
        }

    }
}
```

Gambar 4.32 Skrip Mengatur Pertanyaan (Lanjutan)

Pada Gambar 4.33 merupakan skrip yang mengatur pemilihan jawaban yang benar atau salah dan memberikan warna hijau pada saat pemain menjawab dengan benar dan memberikan warna merah pada saat pemain menjawab dengan salah. Untuk daftar pertanyaan dan jawaban yang muncul diatur pada Inspector. Untuk implementasi tentang mengoreksi jawaban yang dipilih menggunakan skrip di bawah ini.

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

public class AnswerSkrip : MonoBehaviour
{
    public bool isCorrect = false;
    public QuizManager quizManager;

    public Color startColor;

    private void Start()
    {
        startColor = GetComponent<Image>().color;
    }

    public void Answer()
    {
        if(isCorrect)
        {
            GetComponent<Image>().color = Color.green;
            Debug.Log("Correct Answer");
            quizManager.correct();
        }
        else
        {
            GetComponent<Image>().color = Color.red;
            Debug.Log("Wrong Answer");
            quizManager.wrong();
        }
    }

}
```

Gambar 4.33 Skrip Mengatur Jawaban yang Dipilih

Setelah membuat skrip koreksi maka pada Unity juga perlu disesuaikan seperti pada Gambar 4.34 yang menunjukkan daftar pertanyaan pada inspector, jika ingin menambahkan pertanyaan lainya pun bisa diatur melalui inspector ini dan akan otomatis masuk ke dalam daftar pertanyaan yang akan muncul secara acak. Pada Gameplay trivia inilah pemain dapat menentukan apakah sudah memahami tentang huruf kaganga itu sendiri atau belum, jika belum makan pemain dapat memahaminya terlebih dahulu pada menu “belajar” yang dimiliki di dalam game ini.

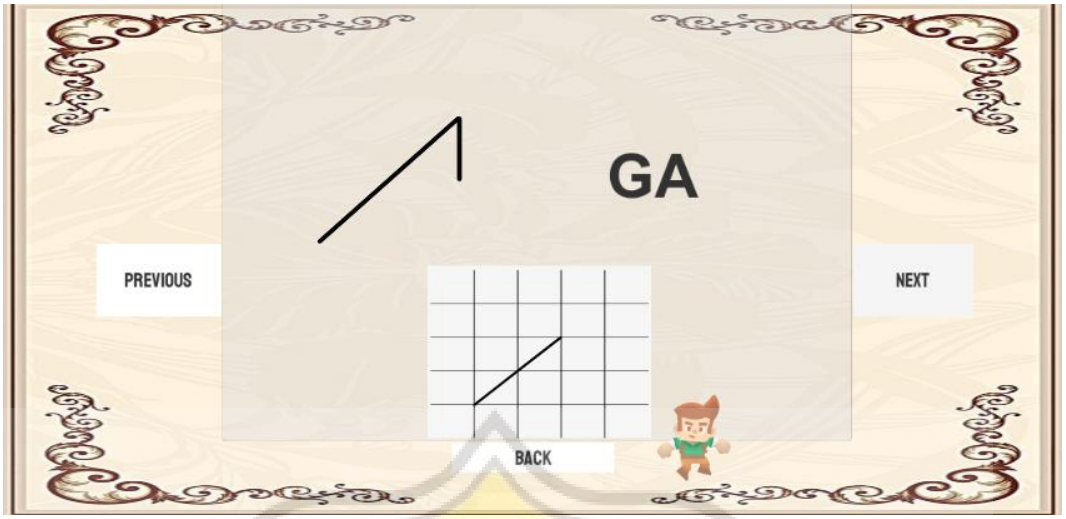
Gambar 4.34 Inspector Daftar Pertanyaan

4.2.3 Implementasi Belajar

Untuk implementasi belajar sendiri memiliki tampilan seperti di Gambar 4.35 adalah tampilan awal dari menu “belajar”, disini pemain dapat belajar dari awal tentang huruf kaganga, mulai dari semua huruf kaganga, perubahan-perubahan huruf vokalnya, cara menulis huruf kaganga itu sendiri dan contoh kata sehari-hari menggunakan huruf kaganga.

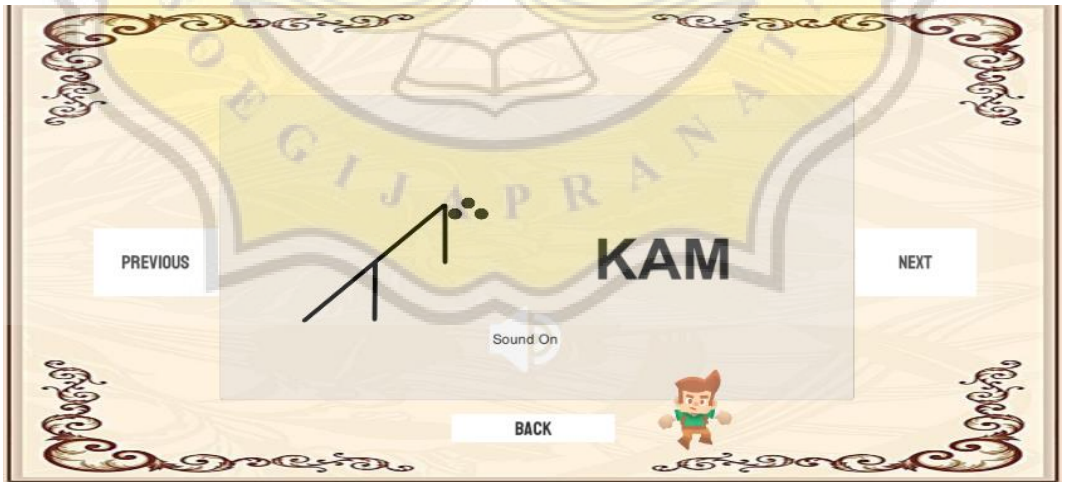
Gambar 4.35 Tampilan Menu Belajar

Pada menu pertama yaitu tampilan huruf dasar dapat dilihat dari Gambar 4.36 menampilkan huruf dasar yang dipilih, pada menu ini menampilkan ke-27 huruf dasar kaganga beserta vidio singkat proses pembuatan setiap huruf kaganga itu sendiri untuk melihat huruf selanjutnya bisa menekan tombol next atau menarik layar dari kanan ke kiri menggunakan jari pada android.



Gambar 4.36 Tampilan Huruf Dasar

Sedangkan pada menu selanjutnya yaitu perubahan bunyi terdapat tambahan fitur seperti pada gambar Gambar 4.37 adalah tampilan dari huruf yang sudah diberi tanda agar huruf vokal dari huruf dasar berubah perubahan ini terdapat 12 perubahan. Pada menu ini memiliki fitur suara yang bisa di dengar agar lebih jelas dalam pemahaman perubahan bunyi ini.



Gambar 4.37 Tampilan Huruf yang Diberi Perubahan Bunyi

Selanjutnya untuk cara penulisan dapat dilihat seperti yang ada pada Gambar 4.38 adalah tampilan dari langkah-langkah penulisan huruf dimulai dari pemberian garis utama, hingga pemberian tanda untuk perubahan bunyi. Setiap layar akan diberi keterangan agar pemain dapat mudah memahami dalam menulis huruf kaganga.

Gambar 4.38 Langkah Dalam Pembuatan Huruf Kaganga

Dan menu terakhir yaitu contoh kata memiliki tampilan seperti di Gambar 4.39 adalah tampilan dari contoh kata yang menggunakan huruf kaganga, dapat dilihat terdapat dua huruf kaganga yang disusun menjadi satu kata. Jika ingin membuat sebuah kalimat maka hanya perlu diberi spasi yang lebih antar satu kata dengan kata berikutnya.

Gambar 4.39 Tampilan kata dengan huruf kaganga

Pada semua bagian menu terdapat fitur untuk menuju ke huruf selanjutnya menggunakan tombol next, hal ini diatur menggunakan skrip yang ada pada Gambar 4.40 berisikan skrip dengan kode yang dapat membuat pemain bisa menggeser halaman berikutnya sehingga pemain melihat keseluruhan huruf kaganga yang telah disusun secara berurutan. Di saat sudah bergeser posisi canvas telah diatur menyesuaikan pada tengah halaman agar terlihat lebih rapi.

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

public class SwipeControl : MonoBehaviour
{
    public GameObject scrollbar;
    float ScrollPos = 0;
    float [] pos;
    int posisi = 0;
    // Start is called before the first frame update
    void Start()
    {

    }
    //Ke slide selanjutnya
    public void next()
    {
        if (posisi < pos.Length - 1)
        {
            posisi += 1;
            ScrollPos = pos[posisi];
        }
    }

      //ke slide sebelumnya
    public void prev()
    {
        if (posisi > 0)
        {
            posisi -= 1;
            ScrollPos = pos[posisi];
        }
    }
// Update is called once per frame

void Update()
    {
        pos = new float[transform.childCount];
        float distance = 1f / (pos.Length - 1f);

for (int i = 0; i < pos.Length; i++)
        {
            pos [i] = distance * i;
        }
```

Gambar 4.40 Skrip Menggeser Canvas

```
if (Input.GetMouseButton(0))
        {
            ScrollPos = scrollbar.GetComponent<Scrollbar>().value;
        } else
        {
            for (int i = 0; i < pos.Length; i++)
            {
                if (ScrollPos < pos [i] + (distance / 2) &&
ScrollPos > pos [i] - (distance / 2))
                {
                    scrollbar.GetComponent<Scrollbar>().value =
Mathf.Lerp(scrollbar.GetComponent<Scrollbar>().value, pos[i],
0.15f);
                    posisi = i;
                }
            }
        }
    }
}
```

Gambar 4.40 Skrip Menggeser Canvas (Lanjutan)

Setelah membuat sebuah skrip maka pengaturan pada unity juga diatur seperti pada Gambar 4.41 menampilkan pengaturan untuk tombol ketika ditekan agar memunculkan bagian yang ingin dilihat dan menyembunyikan bagian yang baru saja dibuka.

Gambar 4.41 Inspector Pada Tombol

4.3 Pengujian Statistik

Setelah menyelesaikan pengembangan game Kaganga Games maka peneliti menentukan hipotesa yang berguna untuk memprediksi hasil dari penelitian dalam pengenalan huruf kaganga. Berikut hipotesa yang telah dirancang:

H1: Pengenalan Aksara Kaganga melalui Game Kaganga Games menarik dan mengedukasi.

H2: Pengenalan Aksara Kaganga melalui Game Kaganga Games mudah untuk dipahami.

H3: Permainan-permainan yang ada pada Game Kaganga Games seru dan menantang untuk dimainkan.

Hipotesa yang telah dirancang ini akan diuji dengan menggunakan metode survei melalui penyebaran form kuesioner yang akan diisi oleh pemain yang sudah memainkan Kaganga Games.

4.3.1 Rancangan Pertanyaan Kuesioner

Berdasarkan hipotesa yang telah disusun maka peneliti telah merancang daftar pertanyaan yang mendukung hipotesa agar mendapatkan hasil yang maksimal dari para responden. Berikut daftar pertanyaan yang akan menjadi tolak ukur dalam penelitian.

a. Performance Expectancy

   PE1. Menurut saya memainkan game "Kaganga Games" bermanfaat untuk mengedukasi saya mengenai Aksara Rejang.

   PE2. Memainkan game "Kaganga Games" dapat membantu saya menjadi mengenal Aksara Rejang.

   PE3. Dengan memainkan game edukasi "Kaganga Games" dapat membantu saya untuk bisa membaca Aksara Rejang.

PE4. Memainkan game “Kaganga Games” dapat mendorong saya untuk terus ingin mengenal lebih dalam Aksara Rejang di kehidupan nyata.

b. Effort Expectancy

EE1. Memamahami cara bermain beberapa permainan pada game “Kaganga Games” mudah bagi saya.

EE2. Menurut saya, instruksi bermain di game “Kaganga Games” sangat jelas dan mudah untuk dipahami.

EE3. Beberapa permainan dalam game “Kaganga Games” mudah untuk dipahami.

EE4. Saya sangat cepat memahami cara bermain game “Kaganga Games”.

c. Hedonic Motivation

HM1. Menurut saya, permainan yang ada di dalam game "Kaganga Games” seru.

HM2. Menurut saya, level pada game "Kaganga Games" sangat menantang.

HM3. Memainkan game “Kaganga Games” dapat membuat saya merasa senang.

d. Behavioral Intention

BI1. Saya berencana untuk tetap memainkan game “Kaganga Games” di kemudian hari.

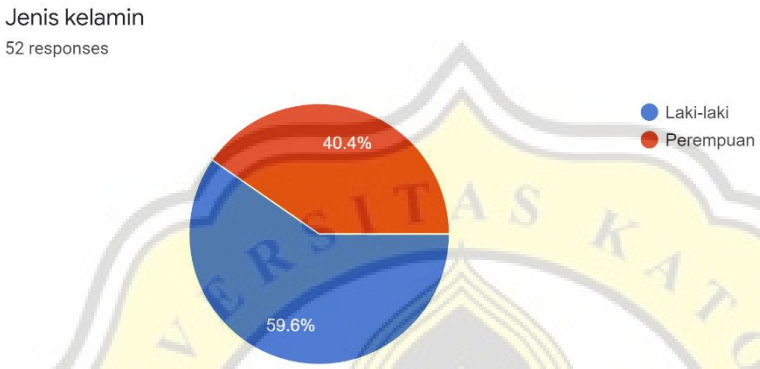
BI2. Saya akan terus mencoba memainkan game “Kaganga Games” di kehidupan sehari-hari saya.

BI3. Saya berencana untuk memainkan game “Kaganga Games” lebih sering.

4.3.2 Data Responden

1. Jenis Kelamin

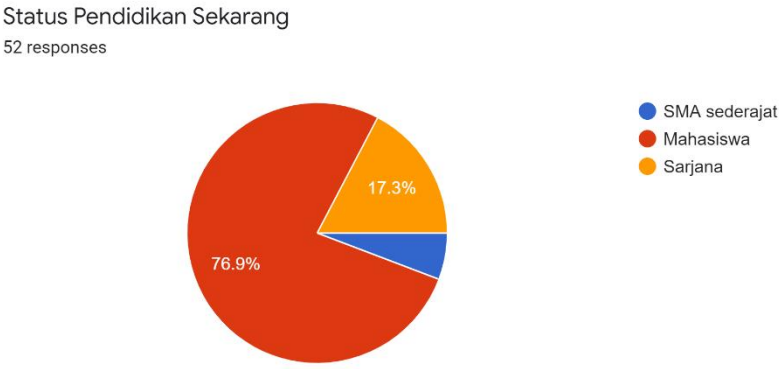
Pada Gambar 4.42 dapat dilihat Dari 52 responden, jenis kelamin responden terbanyak adalah laki-laki dengan persentase 59,6% yaitu 31 responden dari total 52 responden, sedangkan perempuan 40,4% yaitu 21 responden dari total 52 responden.



Gambar 4.42 Persentase Jenis Kelamin Responden

2. Status Pendidikan Sekarang

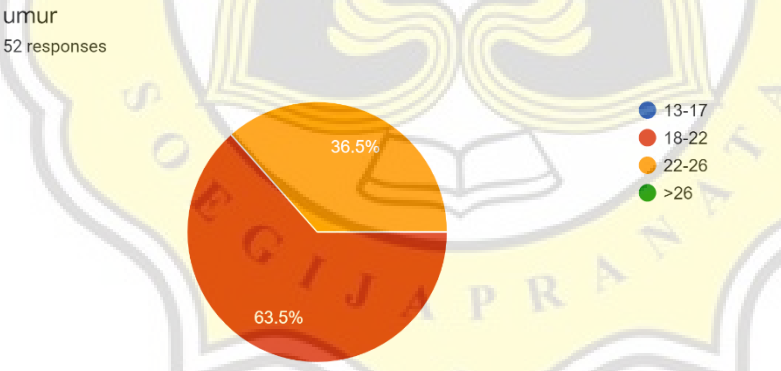
Dapat dilihat pada gambar 4.43 dari 52 responden, Status pendidikan saat ini responden yang telah mengisi terbanyak adalah mahasiswa dengan persentase 76,9% yaitu 40 responden dari total 52 responden, selanjutnya ada yang sudah sarjana dengan persentase 17,3% yaitu 9 responden dari total 52 responden dan terakhir ada yang masih SMA sederajat dengan persentase 5,8% yaitu 3 responden dari 52 total responden.



Gambar 4.43 Persentase Status Pendidikan Responden

3. Usia

Pada gambar 4.44 menunjukkan dari 52 responden, Usia responden yang telah mengisi terbanyak ada pada rentang usia 18-22 tahun dengan persentase 63,5% yaitu 33 responden dari total 52 responden, sedangkan pada rentang 22-26 tahun memiliki persentase 36,5% yaitu 19 responden dari total 52 responden.



Gambar 4.44 Persentase Usia Responden

4.3.3 Statistik Deskriptif

Statistik Deskriptif ini berisikan tentang data keseluruhan dari para responden, dari total 52 responden dapat dilihat pada tabel 4.1 dibawah dengan ketentuan pada Gender angka 1 adalah laki-laki dan angka 2 adalah perempuan. Pada tabel Edukasi, angka 1 menunjukkan bahwa responden saat ini merupakan seorang pelajar tingkat SMA sederajat, sedangkan angka 2 menunjukkan saat ini responden merupakan seorang mahasiswa. Pada tabel Umur angka 1 menunjukkan kisaran usia responden adalah 18-22 tahun dan angka 2 menunjukkan kisaran usia responden adalah 22-26 tahun. Untuk keterangan umur memiliki rata-rata nilai 1.4 dan nilai tengahnya adalah 1, pada status pendidikan responden memiliki nilai rata-rata 2.12 dan nilai tengah 2 sedangkan pada status umur responden memiliki nilai rata-rata 1.37 dan nilai tengahnya adalah 1.

Tabel 4.1 Tabel Hasil Uji Statistik Deskriptif

		G	Edu	A
N	Valid	52	52	52
	Missing	0	0	0
Mean		1.40	2.12	1.37
Median		1.00	2.00	1.00
Mode		1	2	1
Sum		73	110	71

4.3.4 Uji Validitas

Uji Validitas merupakan suatu pengujian yang berguna untuk mengetahui apakah instrumen yang akan diuji merupakan data yang valid (sahih) atau tidak valid. Data yang dimaksud adalah data yang telah dikumpulkan melalui kuesioner yang telah disebarkan kepada responden [13]. Untuk pengujian pada pengenalan Aksara Kaganga melalui Game Kaganga Games menguji variabel PE, EE, HM, dan BI. Bisa dilihat pada tabel pengujian berikut untuk pengujian pertama. Dapat dilihat

pada Tabel 4.2 bahwa uji validitas yang telah dilakukan pada semua variabel dinyatakan valid karena semua variabel memiliki nilai yang berkelompok dan memiliki pola respon yang sama dengan nilai diatas 0.4 sehingga dapat dikatakan data sudah valid pada pengujian pertama.

Tabel 4.2 Uji Validitas

	Component	
	1	2
[PE1]	.062	.746
[PE2]	.232	.764
[PE3]	.270	.611
[PE4]	.598	.493
[EE1]	.221	.466
[EE2]	.022	.802
[EE3]	.298	.776
[EE4]	.475	.559
[HM1]	.552	.519
[HM2]	.721	.090
[HM3]	.634	.318
[BI1]	.787	.233
[BI2]	.812	.087
[BI3]	.845	.200

Extraction Method: Principal Component Analysis.
Rotation Method: Equamax with Kaiser Normalization.

4.3.5 Uji Realibilitas

Uji realibilitas digunakan untuk mengukur tingkat konsisten dari suatu data artinya data tersebut tetap sama hasilnya jika responden mengisi ulang sebuah kuesioner tersebut [13]. Sebelum melakukan uji realibilitas ini harus melakukan uji validitas terlebih dahulu untuk menentukan variabel apa saja yang valid. Dalam

pengujian pada data kuesioner Kaganga Games setelah melalui uji validitas dan semua variabel valid maka untuk uji realibilitasnya mendapatkan hasil sebagai berikut.

Tabel 4.3 menunjukkan hasil uji realibiltas dari varibel PE dengan hasil 0.778.

Tabel 4.3 Uji Realibilitas PE

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.775	.778	4

Tabel 4.4 menunjukkan hasil uji realibiltas dari varibel EE dengan hasil 0.760.

Tabel 4.4 Uji Realibilitas EE

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.757	.760	4

Tabel 4.5 menunjukkan hasil uji realibiltas dari varibel HM dengan hasil 0.700.

Tabel 4.5 Uji Realibilitas HM

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.697	.700	3

Tabel 4.6 menunjukkan hasil uji realibiltas dari varibel BI dengan hasil 0.850.

Reliability Statistics

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.850	.850	3

Tabel 4.6 Uji Realibilitas BI

Untuk keterangan hasil nilai dari Uji Realibilitasnya adalah seperti pada Tabel 4.7 menunjukkan standard rentang nilai yang digunakan dalam penilaian suatu variabel. Sehingga penilaian untuk setiap variabel adalah sebagai berikut.

Tabel 4.7 Rentang Nilai Uji Realibilitas

Cronbach's Alpha	**Internal Consistency**
$\alpha \geq 0.9$	Excellent
$0.9 > \alpha \geq 0.8$	Good
$0.8 > \alpha \geq 0.7$	Acceptable
$0.7 > \alpha \geq 0.6$	Questionable

$0.6 > \alpha \geq 0.5$	Poor
$0.5 > \alpha$	Unacceptable

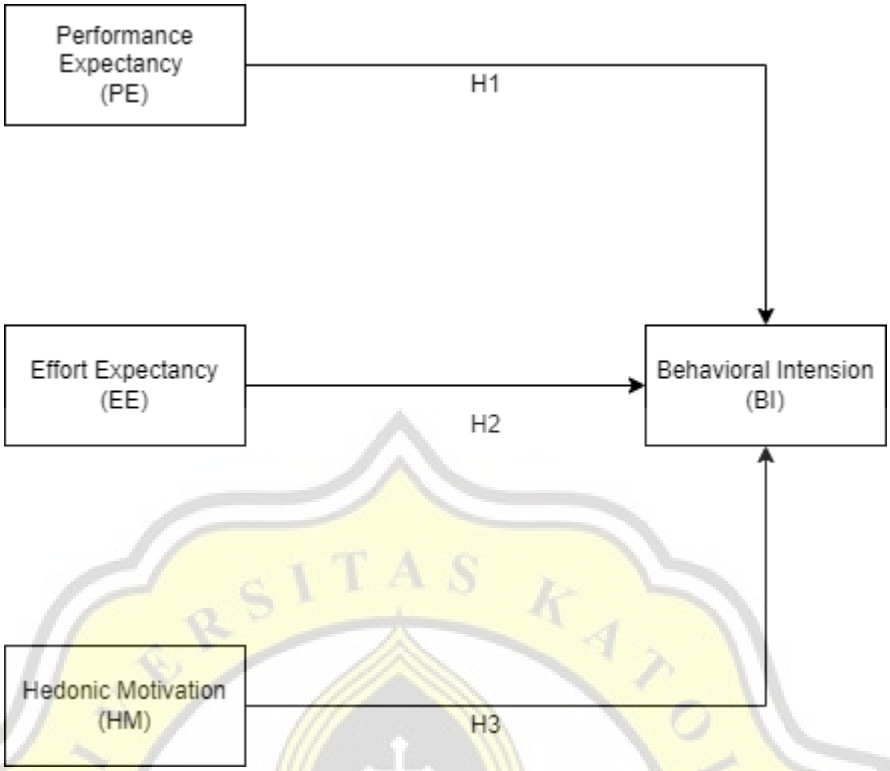
Pada Tabel 4.8 dapat dilihat bahwa hasil uji realibilitas pada variabel PE, EE dan HM adalah *Acceptable* sedangkan pada variaabel BI adalah *Good*.

Tabel 4.8 Hasil Uji Realibilitas

Variabel	**Cronbach's Alpha**	**Internal Consistency**
PE	0.778	Acceptable
EE	0.760	Acceptable
HM	0.700	Acceptable
BI	0.850	Good

4.3.6 Uji Korelasi

Setelah instrumen pengujian validitas dan realibilitas sudah dinyatakan lolos maka selanjutnya adalah pengujian hipotesa yang sudah dibuat di awal dengan menghubungkan korelasi antar variabel [15]. Pada Gambar 4.45 menunjukkan korelasi apa saja yang akan diuji yakni hubungan variabel PE dengan variabel BI, hubungan variabel EE dengan BI dan hubungan variabel HM dengan variabel BI.



Gambar 4.45 Korelasi Pengujian

Dapat dilihat pada Tabel 4.9 hasil dari uji korelasi dapat dilihat bahwa nilai yang keluar memiliki hubungan yang bagus dengan ditandai bintang sebagai petunjuk bahwa variabel yang berhubungan dengan nilai tersebut memiliki hubungan (korelasi) yang kuat. Pada Tabel 4.9 dapat dilihat bahwa nilai korelasi masing-masing ditandai dengan bintang 2 dan nilai significant dibawah 0.004 dari total 52 data. Hasil uji korelasi antar rata-rata variabel PE, EE, HM dan BI yang telah diubah menjadi RPE, REE, RHM dan RBI sehingga dapat disimpulkan bahwa:

H1 terbukti variabel PE dan Variabel BI memiliki korelasi yang baik dengan nilai korelasi di atas 0.4 dan memiliki 2 bintang.

H2 terbukti Variabel EE dan Variabel BI memiliki korelasi yang baik dengan nilai korelasi di atas 0.4 serta mendapatkan 2 bintang.

H3 terbukti Variabel HM dan Variabel BI memiliki korelasi yang baik dengan nilai korelasi di atas 0.4 dan memiliki 2 bintang

Tabel 4.9 Tabel Hasil Uji Korelasi

		RPE	REE	RHM	RBI
RPE	Pearson Correlation	1	.719**	.607**	.552**
	Sig. (2-tailed)		.000	.000	.000
	N	52	52	52	52
REE	Pearson Correlation	.719**	1	.616**	.467**
	Sig. (2-tailed)	.000		.000	.000
	N	52	52	52	52
RHM	Pearson Correlation	.607**	.616**	1	.721**
	Sig. (2-tailed)	.000	.000		.000
	N	52	52	52	52
RBI	Pearson Correlation	**.552****	**.467****	**.721****	1
	Sig. (2-tailed)	.000	.000	.000	
	N	52	52	52	52

**. Correlation is significant at the 0.01 level (2-tailed).

.