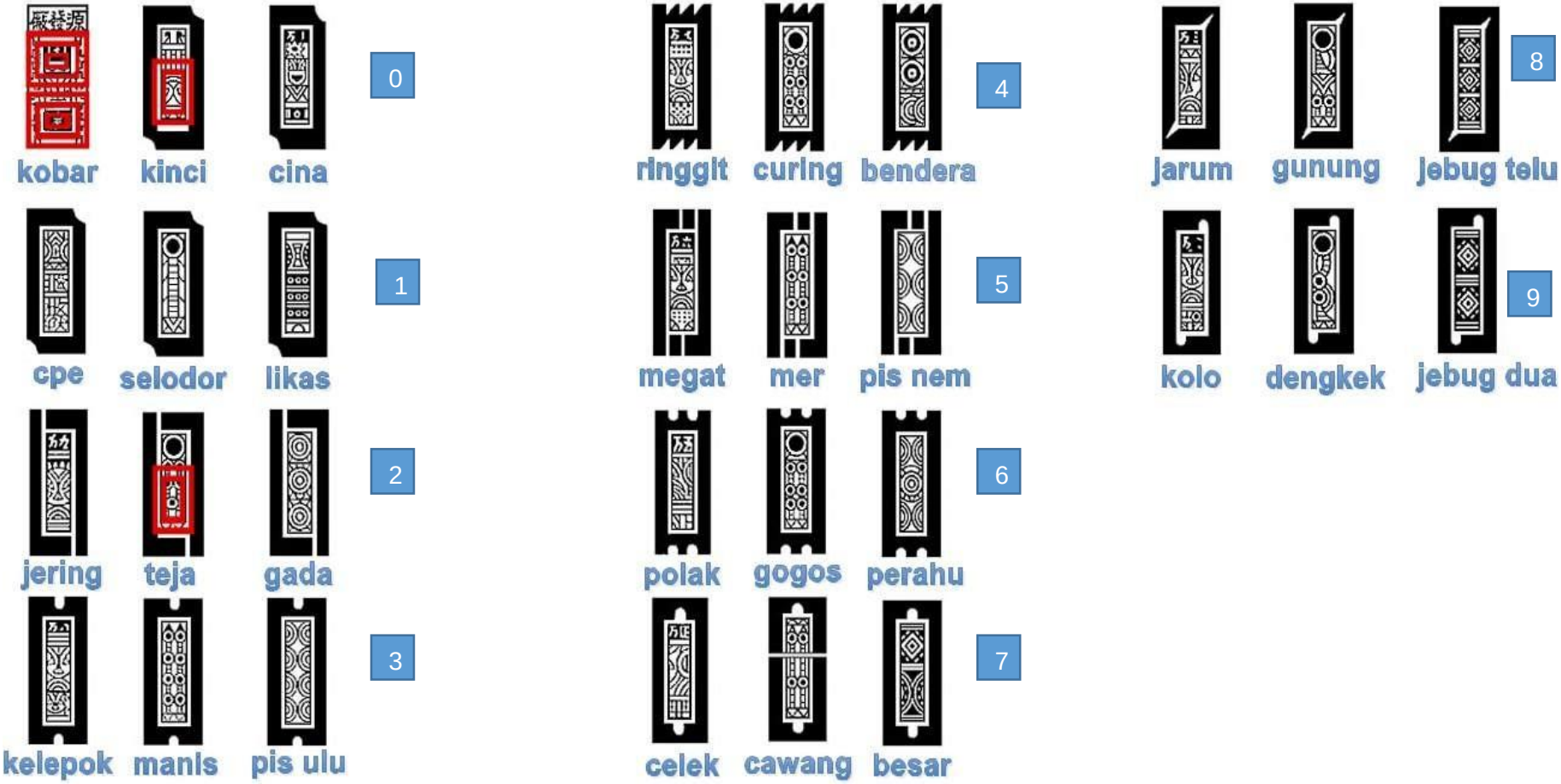
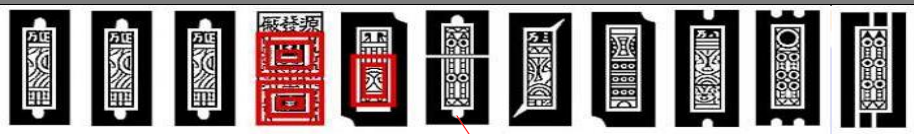
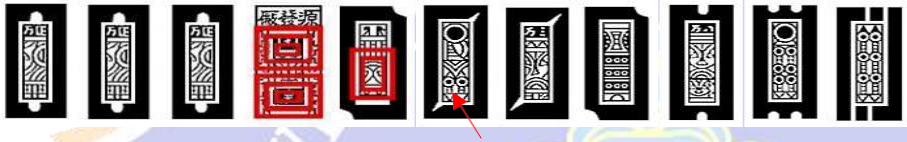
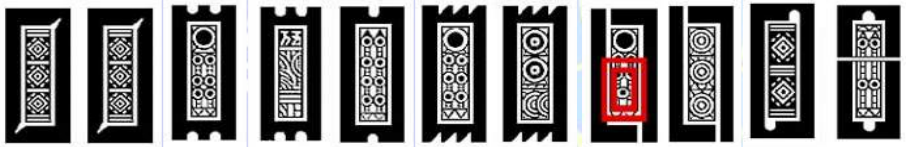
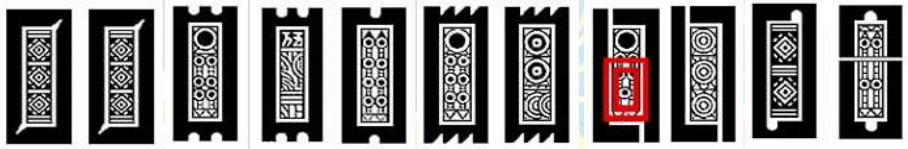
LAMPIRAN

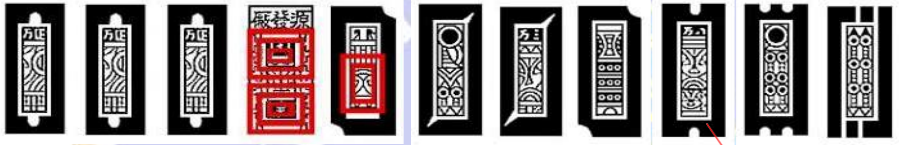
Lampiran 1 nama-nama kartu ceki versi Bali

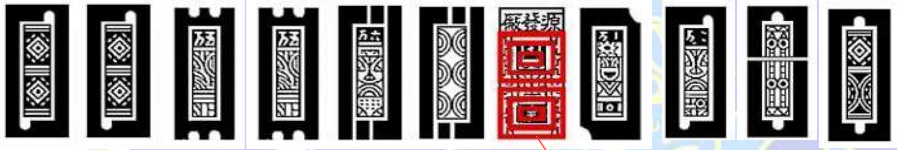


Lampiran 2 Proses permainan ceki 5 player

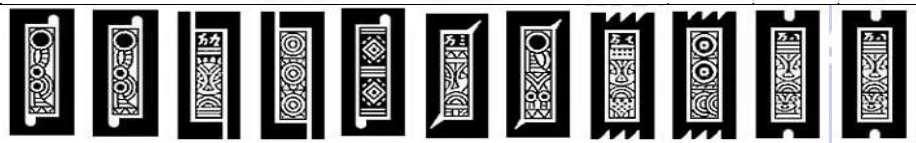
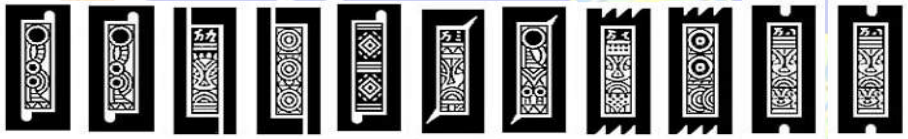
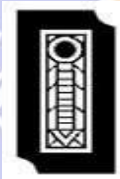
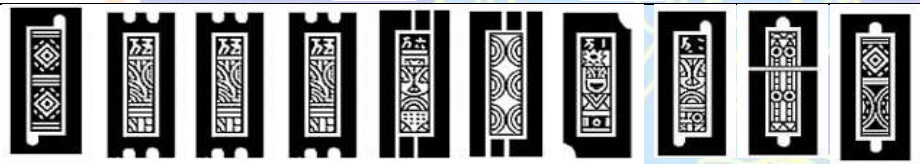
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
1	1	 Sehingga kombinasi kartu player menjadi : 				0
	2	 Sehingga kombinasi kartu player menjadi : 				0

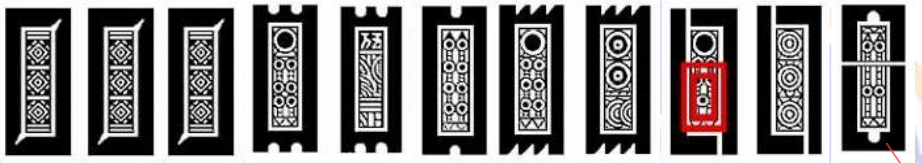
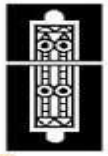
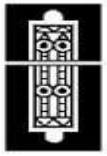
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	3	 Sehingga kombinasi kartu player menjadi : 				0
	4	 Sehingga kombinasi kartu player menjadi : 				0
	5					0

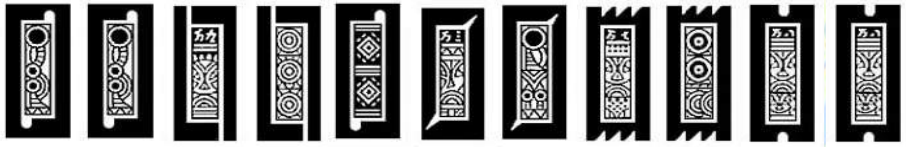
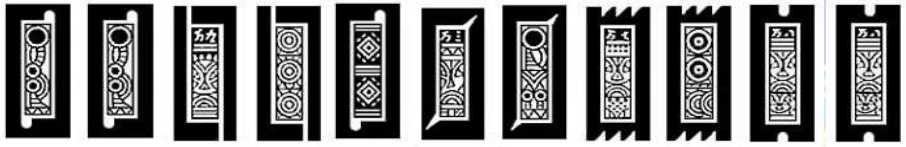
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
		Sehingga kombinasi kartu player menjadi : 				
2	1	 Sehingga kombinasi kartu player menjadi : 				0
	2					0

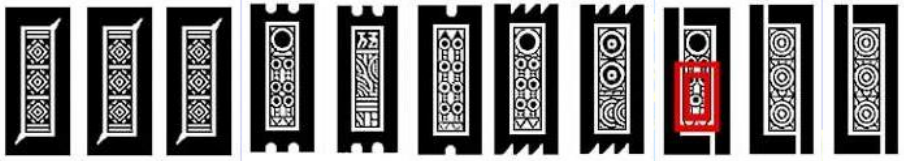
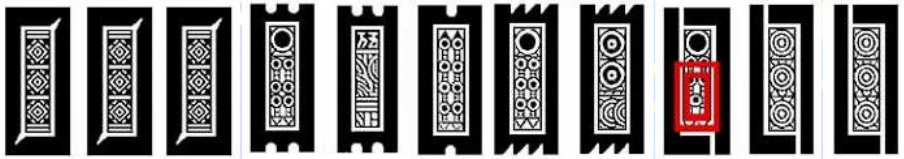
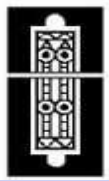
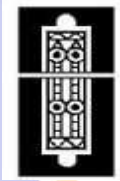
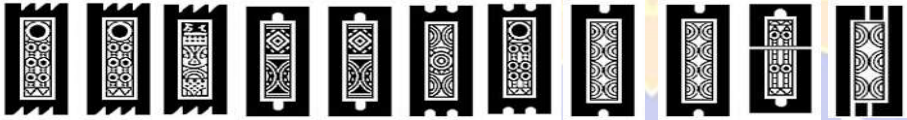
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
		Sehingga kombinasi kartu player menjadi : 				
	3	 Sehingga kombinasi kartu player menjadi : 				0
	4					0

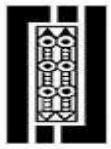
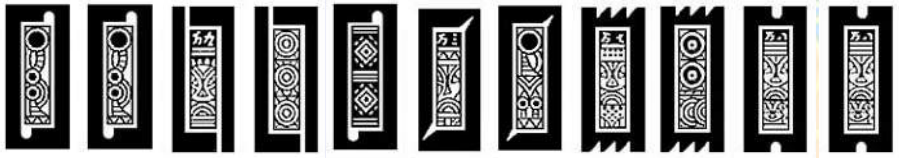
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
		Sehingga kombinasi kartu player menjadi : 				
	5	 Sehingga kombinasi kartu player menjadi : 				0
3	1	 Sehingga kombinasi kartu player menjadi :				0

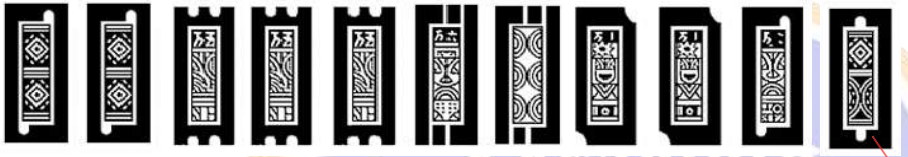
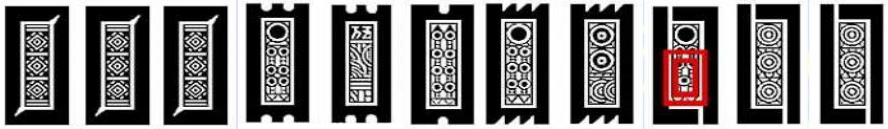
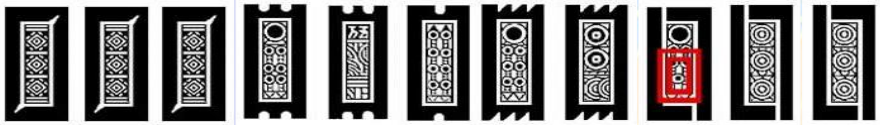
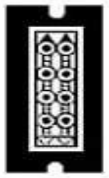
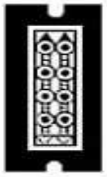
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
						
	2	 Sehingga 				0
	3	 Sehingga kombinasi kartu player menjadi : 				0

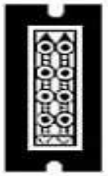
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	4	 Sehingga kombinasi kartu player menjadi : 				0
	5	 Sehingga kombinasi kartu player menjadi : 				0

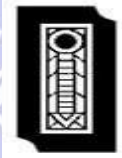
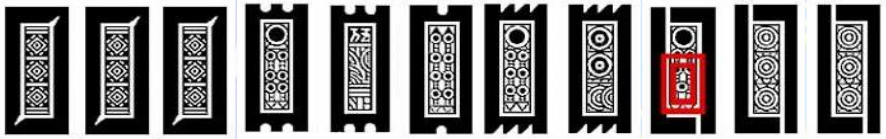
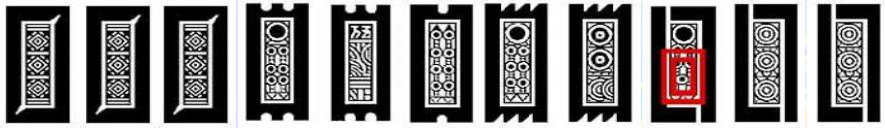
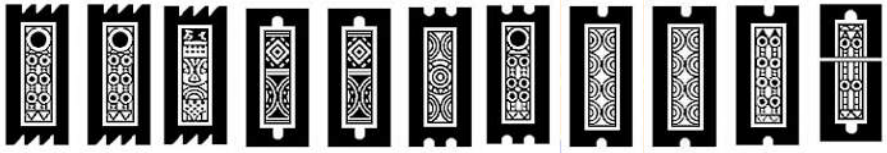
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
4	1	 Sehingga kombinasi kartu player menjadi : 				0
	2	 Sehingga kombinasi kartu player menjadi : 				0
	3					0

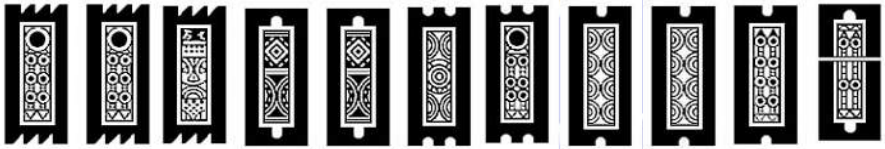
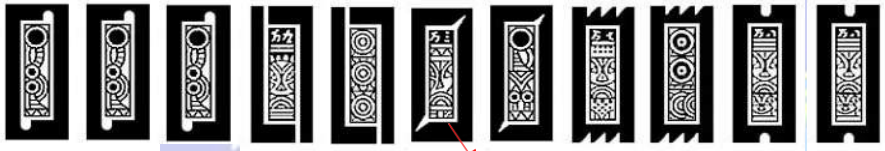
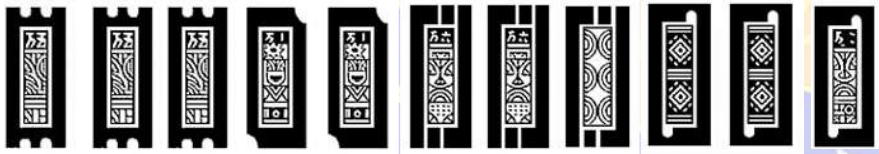
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
		Sehingga kombinasi kartu player menjadi : 				
	4	 Sehingga kombinasi kartu player menjadi : 				0
	5	 Sehingga kombinasi kartu player menjadi : 				0

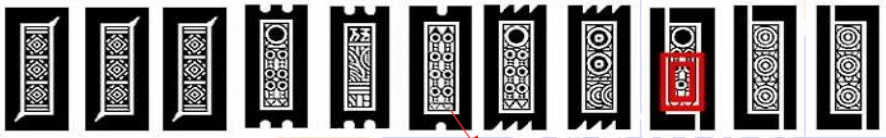
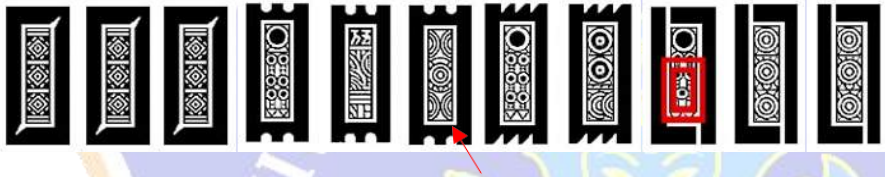
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
5	1	 Sehingga player 1 dalam posisi kunci (kancing) yang artinya tinggal menunggu 1 pasang kartu. dalam posisi kancing, setiap pemain wajib memperlihatkan kartu yang telah diambil dari deck. 				1
	2	 Sehingga kombinasi kartu player menjadi : 				0

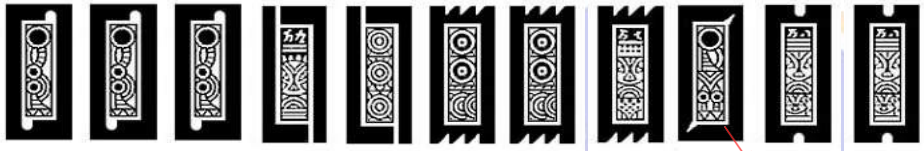
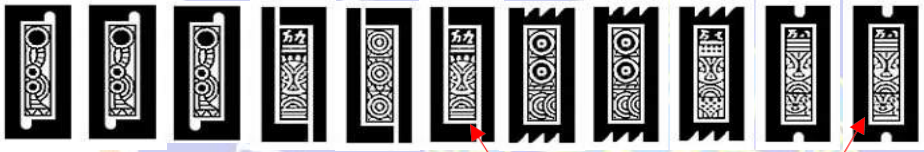
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	3	 Sehingga player 3 dalam posisi kunci (kancing) yang artinya tinggal menunggu 3 pasang kartu. dalam posisi kancing, setiap pemain wajib memperlihatkan kartu yang telah diambil dari deck. 				1
	4	 Sehingga kombinasi kartu player menjadi : 				0

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	5	 Sehingga kombinasi kartu player menjadi : 				0
6	1					0
	2					0

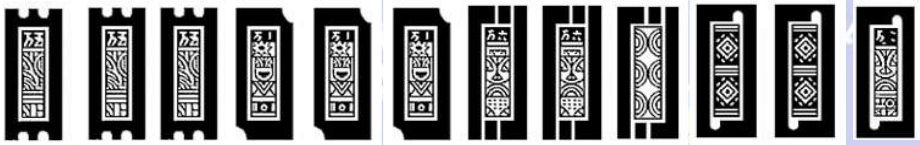
Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
		Sehingga kombinasi kartu player menjadi : 				
	3					1
	4	 Sehingga kombinasi kartu player menjadi : 				0
	5	 Sehingga kombinasi kartu player menjadi :				0

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
						
7	1					1
	2	 Sehingga kombinasi kartu player menjadi : 				0
	3					1

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	4	 Sehingga kombinasi kartu player menjadi : 				0
	5	 Sehingga kombinasi kartu player menjadi : 				0
8	1					1

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	2	 Sehingga player 3 dalam posisi kunci (kancing) yang artinya tinggal menunggu 3 pasang kartu. dalam posisi kancing, setiap pemain wajib memperlihatkan kartu yang telah diambil dari deck. 				1
	3					1
	4	 Sehingga kombinasi kartu player menjadi :				0

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
						
	5	 Sehingga kombinasi kartu player menjadi : 				0
9	1					1
	2					1

Putaran	Player	Kombinasi kartu	Deck	Pit	Mengambil dari pit lawan	Kunci (0/1)
	3	 Sehingga kombinasi kartu player menjadi :  Player 3 memenangkan permainan dengan mendapatkan langsung kartu pasangan terakhir dari deck. Kombinasi kartu yang di dapat oleh player 3 adalah 2 soca, 2 lawang.				1

Lampiran 3 *penjelasan source code dan unit fungsi dari setiap code

source code Mcki.cs

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

source code Mcki.cs

```
using System.Linq;

public class Mcki : MonoBehaviour {

    public Sprite[] cardFaces; // konstruktor untuk menampilkan pola gambar dan tanda kartu ceki
    public GameObject cardPrefab; // gameobject prefab kartu
    public GameObject deckButton; // gameobject tombol deck
    public GameObject[] playerPos; // gamobject untuk posisi player
    public GameObject[] pitPos; //gameobject untuk posisi pit


    public static string[] values = new string[] { "KOBAR", "KINCI", "CINA", "CPE", "SELODOR",
"LIKAS", "JERING",
                                                   "TEJA", "GADA", "KELEPOK", "MANIS",
"PISULU",
                                                   "RINGGIT", "CURING", "BENDERA", "MEGAT",
"MER", "PISNEM",
                                                   "POLAK", "GOGOS", "PERAHU", "CELEK",
"CAWANG", "BESAR",
                                                   "JARUM", "GUNUNG", "JEBUGTELU", "KOLO",
"DENGKEK", "JEBUGDUA" }; //konstruktor nama dari kartu ceki
    public static string[] suits = new string[] { "0", "1", "2", "3" }; //konstruktor untuk
index dari kartu untuk mencapai 120 kartu [kobar0][kobar1]...

    public List<string>[] players; //konstruktor list player
    public List<string>[] pits; //konstruktor pit player
    public List<string> tripsOnDisplay = new List<string>(); //konstruktor untuk menampilkan
gambar kartu dari deck
```

source code Mcki.cs

```
    public List<List<string>> deckTrips = new List<List<string>>(); //konstruktor untuk
mengambil kartu dari deck

    private List<string> player0 = new List<string>();
    private List<string> player1 = new List<string>();
    private List<string> player2 = new List<string>(); //konstruktor list dengan jumlah 5 player
    private List<string> player3 = new List<string>();
    private List<string> player4 = new List<string>();

    public List<string> deck; //konstruktor list deck
    public List<string> discardPile = new List<string>(); //konstruktor list untuk
    private int deckLocation; //kontrukstor int untuk lokasi deck
    private int trips;
    private int tripsRemainder;

    // Use this for initialization
      void Start ()
    {
        players = new List<string>[] {player0 , player1, player2, player3, player4};
//inisialisasi konstruktor list player pada saat mulai game
        PlayCards();// memanggil fungsi memulai kartu
      }

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

      }
```

source code Mcki.cs

```
    public void PlayCards() //fungsi memanggil kartu untuk di deck
    {
        foreach (List<string> list in players)
        {
            list.Clear();
        }
        deck = GenerateDeck();
        Shuffle(deck);

        // test card in deck
        foreach (string card in deck)
        {
            print(card);
        }
        MecekiSort();
        StartCoroutine(MecekiDeal());
        SortDeckIntoTrips();
    }

    public static List<string> GenerateDeck() //fungsi untuk generate deck dengan kartu. contoh
[kobar]+[0]=[kobar0][kobar]+[1]=[kobar1]...
    {
        List<string> newDeck = new List<string>();
        foreach (string s in suits)
        {
            foreach (string v in values)
            {
                newDeck.Add(s+v);
            }
```

source code Mcki.cs

```
        }
        return newDeck;
    }

    void Shuffle<T>(List<T> list) //fungsi untuk mengacak kartu
    {
        System.Random random = new System.Random();
        int n = list.Count;
        while (n > 1)
        {
            int k = random.Next(n);
            n--;
            T temp = list[k];
            list[k] = list[n];
            list[n] = temp;
        }
    }

    IEnumerator MecekiDeal()
    {
        for (int i = 0; i < 1; i++) //fungsi untuk menentukan letak posisi player dan membagikan
kartu player
        {
            float xOffset = -2.5f;
            float yOffset = 0;
            float zOffset = 0.03f;
            foreach (string card in players[i])
            {
                yield return new WaitForSeconds(0.05f);
```

source code Mcki.cs

```
                //perbaiki tampilan
                GameObject newCard = Instantiate(cardPrefab, new
Vector3(playerPos[i].transform.position.x + xOffset, playerPos[i].transform.position.y -
yOffset, playerPos[i].transform.position.z-zOffset), Quaternion.identity,
playerPos[i].transform);
                newCard.name = card;
                newCard.GetComponent<Selectable>().baris = i; // belum bisa
                if (players[i]==players[0])
                {
                    newCard.GetComponent<Selectable>().faceUp = true;
                    xOffset = xOffset + 0.6f;
                    yOffset = 0f;
                    zOffset = zOffset + 0.03f;
                }
                else
                {
                    newCard.GetComponent<Selectable>().faceUp = false;
                }

                discardPile.Add(card);
            }
        }

        foreach (string card in discardPile) //fungsi untuk membuang kartu kartu
        {
            if (deck.Contains(card))
            {
                deck.Remove(card);
            }
```

source code Mcki.cs

```
        }
        discardPile.Clear();
    }

    void MecekiSort() // fungsi untuk membagikan 11 kartu ke player
    {
        for (int i = 0; i < 11; i++)
        {
            for (int j = 0; j < 1; j++)
            {
                players[j].Add(deck.Last<string>());
                deck.RemoveAt(deck.Count - 1);
            }
        }
    }

    public void SortDeckIntoTrips() // fungsi untuk mengambl 1 kartu dari deck
    {
        trips = deck.Count / 1;
        tripsRemainder = deck.Count % 1;
        deckTrips.Clear();

        int modifier = 0;
        for (int i = 0; i < trips; i++)
        {
            List<string> myTrips = new List<string>();
            for (int j = 0; j < 1; j++)
            {
                myTrips.Add(deck[j + modifier]);
```

source code Mcki.cs

```
            }
            deckTrips.Add(myTrips);
            modifier = modifier + 1;

            if(tripsRemainder != 0)
            {
                List<string> myRemainders = new List<string>();
                modifier = 0;
                for (int k = 0; k < tripsRemainder; k++)
                {
                    myRemainders.Add(deck[deck.Count - tripsRemainder + modifier]);
                    modifier++;
                }
                deckTrips.Add(myRemainders);
                trips++;
            }
            deckLocation = 0;
        }
    }

    public void DealFromDeck() //fungsi apabila mengambil kartu deck, maka deck akan berkurang
jumlahnya
    {
        // add cards to discardpile

        foreach (Transform child in deckButton.transform)
        {
            if (child.CompareTag("Card"))
            {
```

source code Mcki.cs

```
                deck.Remove(child.name);
                discardPile.Add(child.name);
                Destroy(child.gameObject);
            }
        }

        if (deckLocation < trips)
        {
            // mengambil 1 kartu baru
            tripsOnDisplay.Clear();
            float xOffset = 1f;
            float zOffset = -0.2f;

            foreach (string card in deckTrips[deckLocation])
            {
                GameObject newpitCard = Instantiate(cardPrefab, new
Vector3(deckButton.transform.position.x + xOffset, deckButton.transform.position.y,
deckButton.transform.position.z + zOffset), Quaternion.identity, deckButton.transform);

                xOffset = xOffset + 0.5f;
                zOffset = zOffset - 0.2f;

                newpitCard.name = card;
                tripsOnDisplay.Add(card);
                newpitCard.GetComponent<Selectable>().faceUp = true;
                newpitCard.GetComponent<Selectable>().inDeckPile = true;
            }
            deckLocation++;
        }
```

source code Mcki.cs

```
        else
        {
            //RestackTopDeck();
        }
    }

    /*void RestackTopDeck()
    {
        foreach (string card in discardPile)
        {
            deck.Add(card);
        }
        discardPile.Clear();
        SortDeckIntoTrips();
    }*/
}
```

Penjelasan:
Source code diatas menjalankan seluruh fungsi konstruktor kartu, konstruktor player, konstruktor pit, konstruktor kartu, fungsi mengacak kartu, mengambil jumlah kartu yang diambil dari deck, membuang kartu ke pit,pembagian kartu ke masing-masing player, penentuan kamera kartu.

Source code UpdateSprite.cs

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

Source code UpdateSprite.cs

```
public class UpdateSprite : MonoBehaviour {

    public Sprite cardFace; //konstruktor sprite untuk menampilkan pola gambar dan tanda kartu
ceki
    public Sprite cardBack; //konstruktor sprite untuk menampilkan bagian belakang kartu
    private SpriteRenderer spriteRenderer; //menyisipkan konstruktor spriterender, untuk
memanggil fungsi sprite renderer dari controller unity
    private Selectable selectable; //menyisipkan script selectable untuk melakukan fungsi di
update sprite
    private Mcki mcki; //menyisipkan script mcki untuk melakukan fungsi di update sprite
    private InputUser inputUser; //menyisipkan script inputuser untuk melakukan fungsi di update
sprite

      // Use this for initialization
      void Start () {
        List<string> deck = Mcki.GenerateDeck(); //memanggil fungsi generate deck pada script
Mcki.cs
        mcki = FindObjectOfType<Mcki>();
        inputUser = FindObjectOfType<InputUser>();//inisialisasi script inputuser

        int i = 0;
        foreach (string card in deck)
        {
            if (this.name == card)
            {
                cardFace = mcki.cardFaces[i]; //mengkombinasikan kartu dengan pola gambar dan
kartu
```

Source code UpdateSprite.cs

```
                break;
            }
            i++;
        }
        spriteRenderer = GetComponent<SpriteRenderer>(); // mendapatkan sprite renderer dari
controller unity
        selectable = GetComponent<Selectable>(); //mendapatkan script selectable
    }

    // Update is called once per frame
    void Update () { //update tampilan kartu dengan pola gambar dan belakang kartu

        if (selectable.faceUp == true)
        {
            spriteRenderer.sprite = cardFace;
        }
        else
        {
            spriteRenderer.sprite = cardBack;
        }

        if (inputUser.slot1)
        {

            if (name == inputUser.slot1.name)
            {
                spriteRenderer.color = Color.yellow;
            }
```

Source code UpdateSprite.cs

```
            else
            {
                spriteRenderer.color = Color.white;
            }
        }
        /*else if (inputUser.slot2)
        {
            if (name == inputUser.slot2.name)
            {
                spriteRenderer.color = Color.yellow;
            }
            else
            {
                spriteRenderer.color = Color.white;
            }
        }
        else if (inputUser.slot3)
        {
            if (name == inputUser.slot3.name)
            {
                spriteRenderer.color = Color.yellow;
            }
            else
            {
                spriteRenderer.color = Color.white;
            }
        }*/
    }
```

Source code UpdateSprite.cs

```
}
```

Penjelasan:
Source code diatas berfungsi untuk mengubah konstruktor menjadi kartu dengan pola gambar dan tanda gambar serta gambar belakang kartu ceki.

Source code Selectable.cs

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

public class Selectable : MonoBehaviour {

    public bool pit = false;
    public string nomor; //konstruktor no kartu [kobar0]--->0
    public int kartu; //  konstruktor kartu untuk kartu yang bersaudara misal kobar,kinci,cina
adalah satu saudara
    public int family;// konstruktor family kartu 0,1,2,3,4,5,6,7,8,9
    public int baris; //belum bisa
    public bool faceUp = false;
    public bool inDeckPile = false;

    private string ValueString; //konstruktor untuk nama kartu
    private string FamilyString; //konstruktor untuk family kartu

      // Use this for initialization
```

Source code Selectable.cs

```
 void Start () //pendefenisian kartu dengan family yang dimiliki
{
    if (CompareTag("Card"))
    {
        nomor = transform.name[0].ToString();

        for (int i=1; i < transform.name.Length; i++)
        {
            char c = transform.name[i];
            ValueString = ValueString + c.ToString();
            FamilyString = FamilyString + c.ToString();
        }

        if (ValueString == "KOBAR")
        {
            kartu = 1;
        }
        if (ValueString == "KINCI")
        {
            kartu = 2;
        }
        if (ValueString == "CINA")
        {
            kartu = 3;
        }
        if (ValueString == "CPE")
        {
            kartu = 1;
```

Source code Selectable.cs

```
            }
            if (ValueString == "SELODOR")
            {
                kartu = 2;
            }
            if (ValueString == "LIKAS")
            {
                kartu = 3;
            }
            if (ValueString == "JERING")
            {
                kartu = 1;
            }
            if (ValueString == "TEJA")
            {
                kartu = 2;
            }
            if (ValueString == "GADA")
            {
                kartu = 3;
            }
            if (ValueString == "KELEPOK")
            {
                kartu = 1;
            }
            if (ValueString == "MANIS")
            {
                kartu = 2;
```

Source code Selectable.cs

```
            }
            if (ValueString == "PISULU")
            {
                kartu = 3;
            }
            if (ValueString == "RINGGIT")
            {
                kartu = 1;
            }
            if (ValueString == "CURING")
            {
                kartu = 2;
            }
            if (ValueString == "BENDERA")
            {
                kartu = 3;
            }
            if (ValueString == "MEGAT")
            {
                kartu = 1;
            }
            if (ValueString == "MER")
            {
                kartu = 2;
            }
            if (ValueString == "PISNEM")
            {
                kartu = 3;
```

Source code Selectable.cs

```
            }
            if (ValueString == "POLAK")
            {
                kartu = 1;
            }
            if (ValueString == "GOGOS")
            {
                kartu = 2;
            }
            if (ValueString == "PERAHU")
            {
                kartu = 3;
            }
            if (ValueString == "CELEK")
            {
                kartu = 1;
            }
            if (ValueString == "CAWANG")
            {
                kartu = 2;
            }
            if (ValueString == "BESAR")
            {
                kartu = 3;
            }
            if (ValueString == "JARUM")
            {
                kartu = 1;
```

Source code Selectable.cs

```
            }
            if (ValueString == "GUNUNG")
            {
                kartu = 2;
            }
            if (ValueString == "JEBUGTELU")
            {
                kartu = 3;
            }
            if (ValueString == "KOLO")
            {
                kartu = 1;
            }
            if (ValueString == "DENGKEK")
            {
                kartu = 2;
            }
            if (ValueString == "JEBUGDUA")
            {
                kartu = 3;
            }
            //-------------------------------------------------------------------------------------------------------------//

            if (FamilyString == "KOBAR")
            {
                family = 0;
            }
            if (FamilyString == "KINCI")
```

Source code Selectable.cs

```
            {
                family = 0;
            }
            if (FamilyString == "CINA")
            {
                family = 0;
            }
            if (FamilyString == "CPE")
            {
                family = 1;
            }
            if (FamilyString == "SELODOR")
            {
                family = 1;
            }
            if (FamilyString == "LIKAS")
            {
                family = 1;
            }
            if (FamilyString == "JERING")
            {
                family = 2;
            }
            if (FamilyString == "TEJA")
            {
                family = 2;
            }
            if (FamilyString == "GADA")
```

Source code Selectable.cs

```
            {
                family = 2;
            }
            if (FamilyString == "KELEPOK")
            {
                family = 3;
            }
            if (FamilyString == "MANIS")
            {
                family = 3;
            }
            if (FamilyString == "PISULU")
            {
                family = 3;
            }
            if (FamilyString == "RINGGIT")
            {
                family = 4;
            }
            if (FamilyString == "CURING")
            {
                family = 4;
            }
            if (FamilyString == "BENDERA")
            {
                family = 4;
            }
            if (FamilyString == "MEGAT")
```

Source code Selectable.cs

```
            {
                family = 5;
            }
            if (FamilyString == "MER")
            {
                family = 5;
            }
            if (FamilyString == "PISNEM")
            {
                family = 5;
            }
            if (FamilyString == "POLAK")
            {
                family = 6;
            }
            if (FamilyString == "GOGOS")
            {
                family = 6;
            }
            if (FamilyString == "PERAHU")
            {
                family = 6;
            }
            if (FamilyString == "CELEK")
            {
                family = 7;
            }
            if (FamilyString == "CAWANG")
```

Source code Selectable.cs

```
            {
                family = 7;
            }
            if (FamilyString == "BESAR")
            {
                family = 7;
            }
            if (FamilyString == "JARUM")
            {
                family = 8;
            }
            if (FamilyString == "GUNUNG")
            {
                family = 8;
            }
            if (FamilyString == "JEBUGTELU")
            {
                family = 8;
            }
            if (FamilyString == "KOLO")
            {
                family = 9;
            }
            if (FamilyString == "DENGKEK")
            {
                family = 9;
            }
            if (FamilyString == "JEBUGDUA")
```

Source code Selectable.cs

```
            {
                family = 9;
            }
        }
    }

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

    }
}
```

Penjelasan:
Source code diatas berfungsi untuk mendefinisikan family kartu, index kartu tiap masing-masing kartu[0]..[3]. Definisi disini dibutuhkan untuk kedepannya dapat menentukan kombinasi kartu seperti soca,lawang,sarigat.

Source code InputUser.cs

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

public class InputUser : Photon.PunBehaviour
{
    public GameObject slot1, slot2, slot3; //konstruktor gameobject untuk mendapatkan kombinasi
kartu "soca, lawang, sarigat"
    private Mcki mcki;// menyisipkan script mcki untuk melakukan fungsi
```

Source code InputUser.cs

```
    public int countSoca = 0, countLawang = 0, countSarigat = 0; //konstruktor untuk perhitungan
jumlah kombinasi yang didapatkan

    //private float timer;
    // Use this for initialization
    void Start()
    {
        mcki = FindObjectOfType<Mcki>();//inisialisasi untuk dapat memanggil fungsi yang ada
pada script Mcki
        slot1 = this.gameObject;//slot kartu 1 untuk menempatkan kartu yang akan di kompare
        slot2 = this.gameObject;//slot kartu 2 untuk menempatkan kartu yang akan di kompare
        slot3 = this.gameObject;//slot kartu 3 untuk menempatkan kartu yang akan di kompare
    }

    // Update is called once per frame
    void Update()
    {
        if (photonView.isMine == false && PhotonNetwork.connected == true) //fungsi untuk
menjalankan multiplayer dengan plugin photon namun belum bisa digunakan
        {
            return;
        }

        GetMouseClick();//mendapatkan fungsi getmouse click
    }

    void GetMouseClick() //fungsi untuk seleksi gameobject yang dipilih dengan cara mengklik
    {
```

Source code InputUser.cs

```
        if (Input.GetMouseButtonDown(0))
        {
            Vector3 mousePosition = Camera.main.ScreenToWorldPoint(new
Vector3(Input.mousePosition.x, Input.mousePosition.y));
            RaycastHit2D hit =
Physics2D.Raycast(Camera.main.ScreenToWorldPoint(Input.mousePosition), Vector2.zero);
            if (hit)
            {
                // what has been hit? Deck/Card/EmptySlot...
                if (hit.collider.CompareTag("Deck"))
                {
                    //clicked deck
                    Deck();
                }
                else if (hit.collider.CompareTag("Card"))
                {
                    // clicked card
                    Card(hit.collider.gameObject);
                }
                else if (hit.collider.CompareTag("Pit"))
                {
                    // clicked pit
                    Pit(hit.collider.gameObject);
                }
                else if (hit.collider.CompareTag("Player"))
                {
                    // clicked player
                    Player(hit.collider.gameObject);
                }
```

Source code InputUser.cs

```
            }
        }
    }

    /*void OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)
    {
        if (stream.isWriting)
        {
            stream.SendNext(transform.position);
        }
        else
        {
            syncPosCard = (Vector3)stream.ReceiveNext();
        }
    }*/

    void Deck() //fungsi deck
    {
        mcki.DealFromDeck();//memanggil fungsi dealfromdeck yang ada pada script mcki
        slot1 = this.gameObject;//saat mengklik deck makan akan mendapatkan kartu yang akan
menjadi slot1
    }

    public void Card(GameObject selected) //fungsi gameobject yang di seleksi
    {
        if (slot1 == this.gameObject) //jika player menyeleksi object yang pertama maka ->slot1
        {
            slot1 = selected;
        }
```

Source code InputUser.cs

```
        else if (slot2 == this.gameObject) //jika player menyeleksi object yang kedua maka -
>slot2
        {
            slot2 = selected;
        }
        else if (slot3 == this.gameObject)//jika player menyeleksi object yang ketiga maka -
>slot3
        {
            slot3 = selected;
        }
        else
        {
            StartCoroutine("Start");//melakukan secara berulang
        }

        if (slot1 && slot2 && slot3 == selected)//fungsi jika kartu pertama, kedua, diseleksi
dengan nilai yang ditentukan dari fungsi komparekartu
        {
            if (KompareKartu(selected) == 1)
            {
                print("Soca");
                countSoca++;
                //KumpulKombinasi(selected);
            }
            else if (KompareKartu(selected) == 2)
            {
                print("Lawang");
                countLawang++;
                //KumpulKombinasi(selected);
```

Source code InputUser.cs

```
            }

            else if (KompareKartu(selected) == 3)
            {
                print("Sarigat");
                countSarigat++;
                //KumpulKombinasi(selected);
            }

            else if (KompareKartu(selected) == 0)
            {
                print("tidak memenuhi kombinasi kartu");
            }
        }
    }

    void Pit(GameObject selected) //fungsi jika kartu ingin dipindahkan/dibuang ke pit
    {
        if (slot1.CompareTag("Card"))
        {
            if (slot1.GetComponent<Selectable>())
            {
                PindahKartuPit(selected);
            }
        }
    }

    void Player(GameObject selected)//fungsi jika kartu ingin diambil player
    {
```

Source code InputUser.cs

```
        if (slot1.CompareTag("Card"))
        {
            if (slot1.GetComponent<Selectable>())
            {
                PindahKartuPlayer(selected);
            }
        }
    }

    public int KompareKartu(GameObject selected) //fungsi melakukan kompare kartu untuk
mendapatkan kombinasi kartu "soca,lawang,sarigat"
    {
        Selectable s1 = slot1.GetComponent<Selectable>();
        Selectable s2 = slot2.GetComponent<Selectable>();
        Selectable s3 = slot3.GetComponent<Selectable>();
        // compare them to see if they stack
        if (s1.family == s2.family && s1.family == s3.family)
        {
            if (s1.kartu == s2.kartu && s1.kartu == s3.kartu)
            {
                return 1;
            }
            else if (s1.kartu != s2.kartu && s1.kartu != s3.kartu && s2.kartu != s3.kartu)
            {
                return 3;
            }
            else
            {
                return 2;
```

Source code InputUser.cs

```
            }
        }
        else
        {
            return 0;
        }
    }


    /*void KumpulKombinasi(GameObject selected)
    {
        Selectable s1 = slot1.GetComponent<Selectable>();
        Selectable s2 = slot2.GetComponent<Selectable>();
        Selectable s3 = slot3.GetComponent<Selectable>();
        float yOffset = 0.3f;

        slot1.transform.position = new Vector3(selected.transform.position.x,
selected.transform.position.y - yOffset, selected.transform.position.z - 0.03f);
        slot1.transform.SetParent(selected.transform);
        slot2.transform.position = new Vector3(selected.transform.position.x,
selected.transform.position.y - yOffset, selected.transform.position.z - 0.03f);
        slot2.transform.SetParent(selected.transform);
        slot3.transform.position = new Vector3(selected.transform.position.x,
selected.transform.position.y + -0.3f, selected.transform.position.z - 0.03f);
        slot3.transform.SetParent(selected.transform);

        //InvokeRepeating("Start", 5.0f, 7);

        if (s1.inDeckPile)
```

Source code InputUser.cs

```
        {
            mcki.tripsOnDisplay.Remove(slot1.name);
        }

        s1.inDeckPile = false;

        slot1 = this.gameObject;
    }*/

    void PindahKartuPlayer(GameObject selected) //fungsi untuk mengambil kartu pit pemain namun
belum bisa dijalankan karena mode multiplayer tidak dapat berjalan
    {
        Selectable s1 = slot1.GetComponent<Selectable>();
        float xOffset = -2.5f;
        float yOffset = 1.5f;
        float zOffset = 0.03f;

        slot1.transform.SetParent(selected.transform);
        slot1.transform.position = new Vector3(selected.transform.position.x + xOffset,
selected.transform.position.y + yOffset, selected.transform.position.z - zOffset);
        //slot1.transform.position = syncPosCard;
        print("Kartu di ambil player");
        if (s1.inDeckPile)
        {
            mcki.tripsOnDisplay.Remove(slot1.name);
            mcki.deck.Remove(slot1.name);
        }

        slot1 = this.gameObject;
```

Source code InputUser.cs

```
    }

    void PindahKartuPit(GameObject selected)//fungsi untuk membuang kartu ke pit
    {
        Selectable s1 = slot1.GetComponent<Selectable>();
        float yOffset = 0.1f;
        float zOffset = 0.1f;

        slot1.transform.SetParent(selected.transform);
        slot1.transform.position = new Vector3(selected.transform.position.x,
selected.transform.position.y + yOffset, selected.transform.position.z-zOffset);
        //slot1.transform.position = syncPosCard;
        print("Kartu di Buang Ke Pit");

        if (s1.inDeckPile)
        {
            mcki.tripsOnDisplay.Remove(slot1.name);
            mcki.deck.Remove(slot1.name);
        }

        slot1 = this.gameObject;
    }

    public bool KondisiMenang() //fungsi untuk menentukan pemenang dengan kombinasi kartu yang
sudah ditentukan
    {
        if (countSoca == 4)
        {
            return true;
```

Source code InputUser.cs

```
        }
        else if (countSoca == 3 && countLawang == 1)
        {
            return true;
        }
        else if (countSoca == 2 && countLawang == 2)
        {
            return true;
        }
        else if (countSoca == 2 && countLawang == 1 && countSarigat == 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
```

Penjelasan:
Source code diatas berfungsi untuk pengecekan inputan player, gameobject yang mana dipilih player(deck, pit, seleksi kartu dll), melakukan compare kartu, mendapatkan kombinasi kartu dan pengecekan untuk jumlah kombinasi kartu yang telah dikumpulkan.

Source code GameManager.cs

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

Source code GameManager.cs

```
public class GameManager : Photon.PunBehaviour
{
    public GameObject lobbyCam;
    public GameObject lobbyUI;
    public Transform[] SpawnPoints;
    public Text playerCountText;
    public Text statusText;
    //public Button startGameButton;

    void Start()
    {
        PhotonNetwork.ConnectUsingSettings("1.0");
        //startGameButton.onClick.AddListener(StartGame);
        //startGameButton.interactable = false;
    }

    void Update()
    {
        statusText.text = PhotonNetwork.connectionStateDetailed.ToString();
    }

    public override void OnConnectionFail(DisconnectCause cause)
    {
        print("Connection Failed !" + cause.ToString());
    }

    public override void OnPhotonPlayerConnected(PhotonPlayer newPlayer)
    {
```

Source code GameManager.cs

```
        playerCountText.text = PhotonNetwork.playerList.Length + "Player(s) Connected.";
    }

    public override void OnPhotonPlayerDisconnected(PhotonPlayer otherPlayer)
    {
        playerCountText.text = PhotonNetwork.playerList.Length + "Player(s) Connected.";
    }

    public override void OnJoinedLobby()
    {
        RoomOptions roomOptions = new RoomOptions() { isVisible = false, maxPlayers = 5 };
        PhotonNetwork.JoinOrCreateRoom("Room", roomOptions, TypedLobby.Default);
    }

    public override void OnJoinedRoom()
    {
        playerCountText.text = PhotonNetwork.playerList.Length + "Player(s) Connected.";

        if (PhotonNetwork.isMasterClient)
        {
            //startGameButton.interactable = true;
        }
    }

    void StartGame()
    {
        PhotonPlayer[] players = PhotonNetwork.playerList;
        for (int i = 0; i < players.Length; i++)
        {
```

Source code GameManager.cs

```
            photonView.RPC("RPCStartGame", players[i], SpawnPoints[i].position,
SpawnPoints[i].rotation);
        }
    }

    [PunRPC]
    void RPCStartGame(Vector3 spawnPos, Quaternion spawnRot)
    {
        lobbyCam.SetActive(false);
        lobbyUI.SetActive(false);
        PhotonNetwork.Instantiate("Player", spawnPos, spawnRot, 0);
    }
}
```

Penjelasan:
Script diatas merupakan controller dari mode multiplayer. Controller ini berfugnsi untuk perhitungan jumlah player yang join ke dalam room, player yang keluar dari room, letak spawn posisi player.

Source code SettingMultiplayer.cs

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

Source code SettingMultiplayer.cs

```
public class SettingMultiplayer : Photon.PunBehaviour {
    //setup pengaturan skor = jumlah kombinasi kartu !
    public Text namaPlayer;
    public Text statusAktifitas;
    public Text namaRoom;
    public Text skorPlayer;
    public Transform[] SpawnPoints;
    //public Transform[] PitPoints;
    //public Text kalahSkor;
    public Text jumlahPlayer;
    public Text textDaftarMenangPlayer;
    public Text kalahNamaPlayer;

    public Button tombolKeluar;
    Button TombolKeluar;

    public Transform player;
    public string tagPlayer = "Player";
    public Transform pit;
    public string tagPit = "Pit";
    public string namaSceneKeluar;
    public string sceneSelanjutnya;

    public GameObject panelMenang;
    public GameObject panelKalah;

    public float lamaPanelMenang = 10f;
```

Source code SettingMultiplayer.cs

```
    public int skor;
    private bool sudahMenang = false;
    public float radiusPlayer = 7f;

    PhotonView pv;

     // Use this for initialization
     void Start () {
        pv = this.GetComponent<PhotonView>();
        if (PhotonNetwork.connected)
        {
            MulaiPlayer();
            statusAktifitas.text += "<color=yellow> Mulai Bergabung </color>\n";
            PhotonNetwork.player.SetScore(0);
        }

        if (pv.isMine)
        {
            skor = (int)PhotonNetwork.player.GetScore();
            namaPlayer.text = PhotonNetwork.player.NickName;
            kalahNamaPlayer.text = "<color=lime>" + PhotonNetwork.player.NickName + "</color>";
        }

        textDaftarMenangPlayer.text = "";

        panelMenang.SetActive(false);

        skorPlayer.text = "Kartu" + skor.ToString();
```

Source code SettingMultiplayer.cs

```
        //kalahSkor.text = "<color=cyan>" + skor.ToString() + "</color>";
        namaRoom.text = PhotonNetwork.room.Name;

        TombolKeluar = tombolKeluar.GetComponent<Button>();
        TombolKeluar.onClick.AddListener(() => MeninggalkanRoom());
    }

    // Update is called once per frame
    void Update () {
        if (!PhotonNetwork.connected)
        {
            PhotonNetwork.offlineMode = true;
        }

        jumlahPlayer.text = PhotonNetwork.countOfPlayersInRooms.ToString() +  "Player";
        skor = (int)PhotonNetwork.player.GetScore();

        if (pv.isMine)
        {
            //kalahSkor.text = "<color=cyan>Skor" + skor.ToString() + "</color>";
            skorPlayer.text = "Kartu"  + skor.ToString();
        }

        if(Menang() && sudahMenang == false)
        {

            pv.RPC("KamuMenang", PhotonTargets.AllBuffered);
            if (panelMenang.activeInHierarchy)
```

Source code SettingMultiplayer.cs

```
            {
                List<PhotonPlayer> players =
PhotonNetwork.playerList.OrderByDescending(p=>p.GetScore()).ToList();
                foreach (var player in players)
                {
                    textDaftarMenangPlayer.text += "Player : <b><color=lime>" + player.NickName
+ "</color></b>, Skor : <b><color=cyan>" + player.GetScore().ToString() + "</color></b>\n";
                }
            }
        }
      }

    public void MeninggalkanRoom()
    {
        if (PhotonNetwork.connected)
        {
            PhotonNetwork.DestroyPlayerObjects(PhotonNetwork.player.ID);
            PhotonNetwork.LeaveRoom();
            PhotonNetwork.LeaveLobby();
        }
    }

    public override void OnLeftRoom()
    {
        pv.RPC("KirimPesan", PhotonTargets.All, "<color=red>" + PhotonNetwork.player.NickName +
"Meninggalkan Room </color>\n");
        Scene sceneIni = SceneManager.GetActiveScene();
        if (sceneIni.name != namaSceneKeluar)
```

Source code SettingMultiplayer.cs

```
        {
            PhotonNetwork.LoadLevel(namaSceneKeluar);
        }
    }

    public void MulaiPlayer()
    {
        int id1 = PhotonNetwork.AllocateViewID();
        //PhotonPlayer[] players = PhotonNetwork.playerList;
        Vector3 posisiRandom = new Vector3(Random.Range(-radiusPlayer, radiusPlayer), 0f,
Random.Range(-radiusPlayer, radiusPlayer));
        pv.RPC("SpawnPlayer", PhotonTargets.All, posisiRandom, transform.rotation, id1);
        pv.RPC("KirimPesan", PhotonTargets.All, "<color=cyan>" + PhotonNetwork.player.NickName +
"Bergabung </color>\n");
        /*for ( int i = 0; i < players.Length; i++)
        {
            //pv.RPC("SpawnPlayer", players[i], SpawnPoints[i].transform.position,
SpawnPoints[i].transform.rotation, id1);
            //pv.RPC("KirimPesan", PhotonTargets.All, "<color=cyan>" +
PhotonNetwork.player.NickName + "Bergabung </color>\n");
            //pv.RPC("SpawnPlayer", PhotonTargets.All, posisiRandom, transform.rotation, id1);
        }*/
    }

    [PunRPC]
    void SpawnPlayer(Vector3 posisi, Quaternion rotasi, int id1)
    {
        if (PhotonNetwork.connected)
```

Source code SettingMultiplayer.cs

```
        {
            Transform Player = Instantiate(player, posisi, rotasi) as Transform;
            Player.name = player.name;
            Transform Pit = Instantiate(pit, posisi, rotasi) as Transform;
            Pit.name = pit.name;
            PhotonView[] nViews = Player.GetComponentsInChildren<PhotonView>();
            nViews[0].viewID = id1;
        }
    }


    [PunRPC]
    void KirimPesan(string pesan)
    {
        statusAktifitas.text += pesan;
    }

    [PunRPC]
    IEnumerator KamuMenang()
    {
        sudahMenang = true;
        panelKalah.SetActive(false);
        panelMenang.SetActive(true);
        yield return new WaitForSeconds(lamaPanelMenang);
        panelKalah.SetActive(false);
        panelMenang.SetActive(false);
        yield return new WaitForSeconds(lamaPanelMenang);
        Scene sceneIni = SceneManager.GetActiveScene();
```

Source code SettingMultiplayer.cs

```
        if (sceneIni.name != sceneSelanjutnya)
        {
            PhotonNetwork.LoadLevel(sceneSelanjutnya);
        }
    }

    public bool Menang()
    {
        if (FindObjectOfType<InputUser>().KondisiMenang())
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
```

Penjelasan:
Source code diatas menjalankan fungsi pengaturan multiplayer, yang terdiri dari inisialisasi koneksi, inisialisasi room, inisialisasi player, sampai penentuan pemenang. Script ini sudah sesuai semestinya. Namun gagal dalam melakukan inisialisasi rpc(player) yang akan bermain.

Source Code Backsound Terus

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

public class BacksoundTerus : MonoBehaviour {
        public static BacksoundTerus obyek = null;

        void Awake(){
                if (obyek == null)
                        obyek = this;
                else if (obyek != null)
                        Destroy(gameObject);

                DontDestroyOnLoad (this.gameObject);
        }
}
```

Penjelasan:
Source code diatas berfungsi untuk menjalankan looping backsound secara terus menerus.

Source Code PindahScene.cs

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

public class PindahScene : MonoBehaviour {

        public AudioSource ButtonSound;
        public string namaScene;

        public void PindahKeScene(){
                AudioSource buttonSound =
ButtonSound.GetComponent<AudioSource> ();
                buttonSound.PlayOneShot
(buttonSound.clip);

                Scene sceneIni =
SceneManager.GetActiveScene ();
                if (sceneIni.name != namaScene)
                        SceneManager.LoadScene
(namaScene);
        }
}
```

Penjelasan:
Fungsi dari source code diatas adalah untuk berpindah dari satu scene ke scene yang lainnya.

Source Code Setting Backsound

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

public class BacksoundSet : MonoBehaviour {

        public string SourceBacksound;

        public static bool BacksoundStatus = true;
        void Start () {
                Debug.Log ("Start Backsound : " +
BacksoundStatus);
                AudioSource backsound =
GameObject.Find
(SourceBacksound).GetComponent<AudioSource> ();

                if(BacksoundSet.BacksoundStatus ==
true){
                        backsound.mute = false;
                }
                else{
                        backsound.mute = true;
                }
        }

}
```

Penjelasan:
Source code diatas berfungsi untuk pengaturan backsound.

Source Code Pindah Panel

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

public class PindahPanel : MonoBehaviour {

        public AudioSource buttonSound;
        public GameObject PanelAwal;
        public GameObject PanelTujuan;

        public void GantiKePanelBaru(){
                buttonSound.PlayOneShot
(buttonSound.clip);
                PanelAwal.SetActive (false);
                PanelTujuan.SetActive (true);
        }
}
```

Penjelasan:

Source code diatas menjalankan fungsi untuk berpindah panel.

Source Code Pengaturan Awal Terus

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

public class PengaturanAwalTerus : MonoBehaviour {
        public static PengaturanAwalTerus obyek =
null;

        void Awake(){
                if (obyek == null)
                        obyek = this;
                else if (obyek != null)
                        Destroy(gameObject);

                DontDestroyOnLoad (this.gameObject);
        }
}
```

Penjelasan:
Sourcode diatas berfungsi untuk pengaturan scene dan gameobject backsound dan yang agar tidak berhenti.

Source Code Inisial Game

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

public class InisialGame : MonoBehaviour {

        public static AudioSource UtamaSound;
        public static GameObject UtamaPanel;
        public static GameObject CaramPanel;
        public static GameObject SettingPanel;
        public static GameObject CreditPanel;
        public static GameObject KeluarPanel;

        public AudioSource utamaSound;
        public GameObject utamaPanel;
        public GameObject caramPanel;
        public GameObject settingPanel;
        public GameObject creditPanel;
        public GameObject keluarPanel;

        // Use this for initialization
        void Start () {
                Screen.sleepTimeout =
SleepTimeout.NeverSleep;
                UtamaSound = utamaSound;
                UtamaPanel = utamaPanel;
                CaramPanel = caramPanel;
                SettingPanel = settingPanel;
                CreditPanel = creditPanel;
                KeluarPanel = keluarPanel;
        }


}
```

Penjelasan:

Source code diatas digunakan untuk inisialisasi audio, panel dan screen game.

Source Code Load Awal

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

public class LoadAwal : MonoBehaviour {

        public GameObject pengaturanAwal;
        public GameObject backsound;

        void Awake(){
                if(PengaturanAwalTerus.obyek ==
null){
                        GameObject baruPengaturan =
(GameObject)Instantiate (pengaturanAwal) as
GameObject;
                        baruPengaturan.name =
pengaturanAwal.name;
                }

                if(BacksoundTerus.obyek == null){
                        GameObject baruBacksound =
(GameObject)Instantiate (backsound) as GameObject;
                        baruBacksound.name =
backsound.name;
                }
        }
}
```

Penjelasan:
Source code diatas berfungsi untuk melakukan load pada script pengaturanAwal Terus dan gameobject backsound.

Lampiran 3 contoh angket pengujian Black Box

ANGKET PENGUJIAN PENGEMBANGAN *GAME* MECEKI BERBASIS ANDROID

Nama :……

Tipe Perangkat Android :……

Total RAM :……

Versi Android :……

CPU :……

PETUNJUK PENGISIAN ANGKET

Berikan tanda centang (✓) pada salah satu kolom pilihan jawaban yang tersedia.

Contoh 1:

No	Pernyataan	Kesesuaian	
		Sesuai	Tidak Sesuai
1.	Aplikasi dapat dijalankan pada perangkat android yang digunakan	✓	

Lampiran 4 contoh angket pengujian ahli isi

ANGKET PENGUJIAN AHLI ISI PENGEMBANGAN *GAME* MECEKI BERBASIS ANDROID

Nama :……

Pendidikan :……

Status :……

PETUNJUK PENGISIAN

Untuk mengisi daftar penilaian dibawah ini, dimohon memilih salah satu kesesuaian yang telah disediakan dengan memberi tanda centang (✓) pada kolom pilihan jawaban yang tersedia.

Contoh 1:

No	Pernyataan	Alternatif Jawaban	
		Sesuai	Tidak Sesuai
1.	Kesesuaian jalannya aplikasi dengan keadaan sebenarnya.	✓	
2.	Saat kondisi ceki/mecari/kancing pemain dapat melihat kartu yang diambil dari pemain lawan.		

Lampiran 5 contoh angket pengujian ahli media

ANGKET PENGUJIAN AHLI MEDIA PENGEMBANGAN *GAME* MECEKI BERBASIS ANDROID

Nama :……

Pendidikan :……

Status :……

PETUNJUK PENGISIAN

Untuk mengisi daftar penilaian dibawah ini, dimohon memilih salah satu kesesuaian yang telah disediakan dengan memberi tanda centang (✓) pada kolom pilihan jawaban yang tersedia.

Contoh 1:

No	Pernyataan	Alternatif Jawaban	
		Sesuai	Tidak Sesuai
Kesesuaian Audio			
1.	Kesesuaian music (back sound).		
Kesesuaian Visual			
2.	Ketepatan pemilihan jenis huruf.		
3.	Ketepatan pemilihan ukuran huruf.		
4.	Kesesuaian komposisi warna huruf dengan latar.		
5.	Kesesuaian objek 2D dengan gambar latar pada aplikasi.		

Lampiran 6 contoh angket respon pengguna

ANGKET PENGUJIAN RESPON PENGGUNA PENGEMBANGAN *GAME* MECEKI BERBASIS ANDROID

Nama :……

Pendidikan :……

Umur :……

Status :……

PETUNJUK PENGISIAN

Untuk mengisi pernyataan dibawah ini, dimohon memilih salah satu jawaban yang paling sesuai dari jawaban-jawaban yang telah disediakan, dengan memberi tanda centang (✓) pada kolom yang tersedia. Untuk keterangan jawaban bisa dilihat dibawah ini.

SS = Sangat Setuju

S = Setuju

TS = Tidak Setuju

STS = Sangat Tidak Setuju

Contoh 1:

No	Kriteria	Alternatif Jawaban			
		SS	S	TS	STS
1.	Saya merasa senang memainkan *Game* Meceki Berbasis Android karena lebih menarik dan seru.				
2.	Saya kesulitan memainkan *game* meceki.				