4. IMPLEMENTASI SISTEM

Bab ini akan membahas tentang implementasi sistem sesuai dengan yang analisis dan desain sistem yang telah dibuat. Bahasa pemrograman yang digunakan untuk pengimlementasian ini adalah C#. *Software* dan *resource* yang digunakan adalah *unity*, *adobe photoshop*, mono *developer*, firebase SDK. Langkah langkah pembuatan dan pengimplementasian pertama mengkoneksikan *Unity* dengan firebase SDK seperti pada Tabel 4.1. Setelah terkoneksi ke database firebase SDK dengan *unity* step berikutnya membuat bahasa pemrograman menggunakan mono *developer* dimana mono *developer* merupakan aplikasi untuk menuliskan bahasa pemrograman dari *unity*. Sambil berjalannya pengerjaan program pembuat juga mengerjakan image yang digunakan untuk mengisi program dengan menggunakan *adobe photoshop*. Dan selanjutnya memasukkan *sound* untuk melengkapi *game*.

Tabel 4.1 Menyambungkan *Unity* dengan Firebase SDK

Menambahkan Firebase ke program yang digunakan

1. Buat firebase project di firebase console.
2. Klik Add Firebase ke program yang di gunakan.
3. Dan yang terakhir, download “Google-Services.json” file di https://support.google.com/firebase/answer/7015592.

Menambahkan SDK ke *Unity*

1. Buat project di *unity*.
2. Download Firebase *Unity* SDK di https://dl.google.com/firebase/sdk/*unity*/firebase_*unity*_sdk_4.5.1.zip.
3. Buka *Unity* Project di *Unity* Project Editor.
   - Pilih “File > *Open* Project”.

- Klik “*Open*”.
- Arahkan ke tempat aplikasi dikotak dialog dan klik”*Open*”.

4. Memasukan Plugin Fitur Firebase, Sebagai Contoh “Firebase Analytics.

- Pilih “Assets > Import Package > Custom Package”.
- Masukkan “FirebaseAnalytics.*unit*ypackage” dari “Firebase *Unit*y SDK” yang telah didownload.
- Klik “Import” ketika “Import *Unit*y Package” menu terbuka.

5. Menambahkan “Google-Services.json” ke file project.

- Tarik “Google-Services.json” yang telah di download ke Firebase konsol di *Unit*y Projek.

6. Build for android

- Pilih “File > Build Settings” dimenu option.
- Select “Android” di “Platform” list.
- Klik “Switch Platform” untuk memilih “Android” sebagai target platform.
- Tunggu icon sampai berhenti di bawah kanan.

Klik “Build and Run”.

Untuk implementasi code pada *game battleship* akan dibagi menjadi segmentasi sebagai berikut:

Tabel 4.2 Tabel *Segment*asi

Nama Program	Keterangan	Segmen Program	Flowchart
Koneksi *database* Pengambilan Data Dari *database*	Koneksi dengan *database*	4.3, 4.4	3.4.2
Register	Pembuatan akun baru	4.5	3.6.1
Login	Masuk dengan akun yang sudah ada	4.6	3.6.1
Metode BFS	Metode pencarian jalur	4.7	3.4.3
Map	Fungsi pemanggilan *Map*	4.8	3.4.4
Unit	Fungsi pemanggilan Status *unit*	4.9	-
Single *Player*	Fungsi yang menangani *Singleplayer*	4.10	3.4.5
Multi *Player*	Fungsi yang menangani *Multiplayer*	4.11	3.4.6

4.1 Koneksi *Database*

Koneksi *database* diperlukan untuk dapat mengakses informasi *user* dan mengelola data *user*. Koneksi dengan *database* menghubungkan *database* Firebase dengan *game battleship* online.

Segmen Program 4.3 *Source code* untuk connect ke *database*

```
  FirebaseApp.DefaultInstance.SetEditorDatabaseUrl("https://skripsi-
40fce.firebaseio.com/");
  reference = FirebaseDatabase.DefaultInstance.RootReference;
```

Segmen Program 4.4 *Source code* untuk pengambilan data dari *database*

```
FirebaseDatabase.DefaultInstance
            .GetReference("User")
            .ValueChanged += (object sender2, ValueChangedEventArgs e2
) =>
            {
                if (e2.DatabaseError != null)
                {

                }
                //Debug.Log("Isi Test : " +e2.Snapshot.Value);

                if (e2.Snapshot != null && e2.Snapshot.ChildrenCount >
 0)
                {
                    foreach (var childSnapshot in e2.Snapshot.Children
)
                    {
                        var key = childSnapshot.Key;

                        IDictionary<string, object> iuser = (IDictiona
ry<string, object>)childSnapshot.Value;
                        if (iuser["Username"].Equals(checkMenuBegin.us
erlogin))
                        {
                            dataPlayer = iuser;
                            keyBattle = iuser["Battle"] + "";
                        }
                    }
                    if (dataPlayer != null)
                    {
                        foreach (var childSnapshot in e2.Snapshot.Chil
dren)
                        {
                            IDictionary<string, object> iuser = (IDict
ionary<string, object>)childSnapshot.Value;

                            if (iuser.ContainsKey("Key"))
                            {
                                if (iuser["Key"].Equals(dataPlayer["En
emy"]))
                                {
                                    dataEnemy = iuser;
                                }
                            }
```

```
                        }
                    }
                }
            };
```

4.2 *Register*

Fungsi *Register* dijalankan ketika *user* ingin memulai permainan tetapi tidak memiliki account. Pada proses ini aplikasi akan menjalankan fungsi *register* untuk menambah data *user* baru kedalam *database* dan mengecek apakah *user* tersebut sudah ada dalam *database* atau belum dan apakah password confirmasinya sudah sama atau belum.

Segmen Program 4.5 *Source code Register* dan cek *User*

```
bool checkDataInput()
    {
        string username=tbUsernameRegister.text;// = tbUsername.text;
        string password=tbPasswordRegister.text;// = tbPassword.value;
        string password2=tbPasswordKonfirmasi.text;

        string warningmsg = "";
        if (!password.Equals (password2)) {
            warningmsg = warningmsg + "Password dan kornfirmasinya har
us sama\n";
        }
        for (int i = 0; i < listUser.Count; i++) {
            IDictionary iUser=listUser[i];
            if (iUser ["Username"].Equals (username)) {
                warningmsg = warningmsg + "Username anda sudah digunak
an\n";
            }
        }
        if (warningmsg.Equals ("")) {
            return true;
        } else {
            tbWarningRegister.text = warningmsg;

            return false;
        }
        //return returnVal;
    }
    void RegisterData()
    {
        if (checkDataInput ()) {
```

Segmen Program 4.5 *Source code Register* dan cek *User*(Lanjutan)

```
            string username=tbUsernameRegister.text;// = tbUsername.te
xt;
            string password=tbPasswordRegister.text;// = tbPassword.va
lue;
            string password2=tbPasswordKonfirmasi.text;

            int ctr=1;
            string key="User"+ctr;
            for (int i = 0; i < listKey.Count; i++) {
                if (key.Equals (listKey [i])) {
                    ctr++;
                    key = "User" + ctr;
                }
            }
            User nu = new User (username,password);
            string json = JsonUtility.ToJson (nu);
            reference.Child ("User").Child (key).SetRawJsonValueAsync
(json);

            setInvisiblePanelRegister ();
            setVisibleLogin ();
            AudioSource ar = (AudioSource)suara.GetComponent<AudioSour
ce> ();
            ar.Play ();
        }

    }
```

4.3 *Login*

Fungsi *login* dilakukan ketika *user* ingin memulai permainan dan *user* tersebut sudah memiliki akun. Pada proses ini aplikasi akan menjalankan fungsi untuk mengecek apakah *user*name dan password yang dimasukan apakah sudah benar. Setelah benar program akan membawanya ke menu lanjutan

Segmen Program 4.6 *Source code Login*

```
void PindahMenuUtama()
    {

        AudioSource ar = (AudioSource)suara.GetComponent<AudioSource>
();
        ar.Play ();

        string username=tbUsername.text;
        string password=tbPassword.text;
        //Debug.Log ("input user "+username);
        //Debug.Log("input password : "+password);
```

Segmen Program 4.6 *Source code Login* (lanjutan)

```
        bool loginOk = false;
        for (int i = 0; i < listUser.Count; i++) {
            IDictionary iUser=listUser[i];
            //Debug.Log ("User "+iUser["Username"]+","+iUser["Password
"]);
            if (iUser ["Username"].Equals(username) && iUser ["Passwor
d"].Equals(password)) {
                loginOk = true;
            }
        }

        if (loginOk) {
            tbWarning.text = "";
            checkMenuBegin.userlogin = username;
            loadScene = 0;
            //Application.LoadLevel ("sceneMenuUtama");
        } else {
            tbWarning.text = "Username dan password yang anda masukkan
 tidak terdaftar";
            //MessageBox.Show ("Login gagal dilakukan");
        }
        //Application.LoadLevel ("sceneMenuUtama");
    }
```

4.4 Metode BFS

Fungsi ini dilakukan ketika *user* bermain *singleplayer* dan ingin mengarahkan *unit* mereka ke arah musuh. Program akan mencari jalur pendek untuk mencapai titik tersebut.

Segmen Program 4.7 Metode BFS

```
public bool checkAda(PosM pm)
    {
        int i = 0;
        bool found = false;
        while (i < aMove.Count && !found) {
            PosM pmc=(PosM)aMove[i];
            if (pm.x == pmc.x && pm.y == pmc.y) {
                //if (pmc.move.Length < pm.move.Length) {
                    found = true;
                //}
            }
            i++;
        }
```

Segmen Program 4.7 Metode BFS (lanjutan)

```
        if (solusi != null) {
            if (pm.move.Length > solusi.move.Length) {
                found = true;
            }
        }
        return found;
    }

PosM solusi;

public void startMove(PosM pmB)
    {
        int x = pmB.x;
        int y = pmB.y;
        Debug.Log ("check x : "+x+", check y : "+y);
        string move = pmB.move;
        //kalau titik sudah ketemu disimpan, pergerakannya

        if (x == xTujuan && y == yTujuan) {
            if (solusi == null) {
                solusi = pmB;
            }
            else if (pmB.move.Length < solusi.move.Length) {
                solusi = pmB;
            }
        }
        else {
            //gerak atas
            if (y > 0) {
                int yNew = y - 1;
                PosM pm = new PosM (x,yNew,move+"1");
                if (!checkAda (pm)) {
                    if (cmap [pm.y, pm.x] == 1 || cmap [pm.y, pm.x] ==
 3) {
                        aMove.Add (pm);
                    }
                }
            }

            //gerak bawah
            if (y < sizeMapY-1) {
                int yNew = y + 1;
                PosM pm = new PosM (x,yNew,move+"2");
                if (!checkAda (pm)) {
                    if (cmap [pm.y, pm.x] == 1 || cmap [pm.y, pm.x] ==
 3) {
                        aMove.Add (pm);
                    }
                }
            }

            //gerak kiri
```

Segment Program 4.7 Metode BFS (Lanjutan)

```
            if (x > 0) {
                int xNew = x - 1;
                PosM pm = new PosM (xNew,y,move+"3");
                if (!checkAda (pm)) {
                    if (cmap [pm.y, pm.x] == 1 || cmap [pm.y, pm.x] ==
3) {
                        aMove.Add (pm);
                    }
                }
            }

            //gerak kanan
            if (x<sizeMapX-1) {
                int xNew = x + 1;
                PosM pm= new PosM (xNew,y,move+"4");
                if (!checkAda (pm)) {
                    if (cmap [pm.y, pm.x] == 1 || cmap [pm.y, pm.x] ==
3) {

                       aMove.Add (pm);
                    }
                }
            }
        }
        if (aMove.Count > 0) {
            PosM pmA = (PosM)aMove [0];
            aMove.RemoveAt (0);

            startMove (pmA);
        }
        else {
            if (solusi == null) {
                sedangGerak = false;
                Debug.Log ("Tidak ditemukan");
            } else {
                tcheck = 0;
                checkmp ();
                Debug.Log (solusi.move);

            }

        }
    }
```

4.5 *Map*

Fungsi ini digunakan ketika *user* memilih *map* yang akan digunakan untuk bertarung. Fungsi ini digunakan untuk membedakan *map* mana yang akan di panggil ketika *player* telah menentukan *map* yang akan dilawan.

Segmen Program 4.8 *Source code Map*

```
void Start () {
        ar = GetComponent<AudioSource> ();
        float lebarTileX = Mathf.Abs((topX2 - topX) / sizeMapX);
        float lebarTileY = Mathf.Abs((topY2 - topY) / sizeMapY);

        Debug.Log("Lebar "+lebarTileX +", tinggi "+lebarTileY);

        float scaleX = 3.1f;
        float scaleY = 3f;
        tile.transform.localScale = new Vector3(scaleX,scaleY,tile.tra
nsform.localScale.z);

        if (tipe.Equals ("1")) {
            goMap= new GameObject[sizeMapY, sizeMapX];
            cmap = new int [5,10] {
                {1,1,1,1,1,8,1,1,1,1} ,   /*  initializers for row ind
exed by 0 */
                {1,1,1,1,1,8,1,1,1,1}  ,   /*  initializers for row in
dexed by 1 */
                {1,5,4,1,3,8,1,5,4,1} ,
                {1,7,6,1,1,8,1,7,6,3},
                {1,1,3,1,1,1,1,1,9,1}
            };
        } else if (tipe.Equals ("2")) {
            topX = -5.1f;
            topX2 = 8.8f;
            topY = 4.21f;
            topY2 = -5.1f;

            sizeMapX = 8;
            sizeMapY = 6;
            goMap= new GameObject[sizeMapY, sizeMapX];
            cmap = new int [6,8] {
                {1,1,1,1,1,1,5,4} ,
                {1,1,1,1,1,1,7,6} ,
                {3,9,1,1,1,1,1,1}  ,
                {1,1,8,8,9,1,1,1} ,
                {5,4,3,1,1,8,9,1},
                {7,6,1,1,1,1,1,3}
            };
        } else if (tipe.Equals ("3")) {
            goMap= new GameObject[sizeMapY, sizeMapX];
            posCharUtamaX = 2;
            posCharUtamaY = 2;
            cmap = new int [5,10] {
                {1,5,4,1,1,1,1,1,1,1} ,
                {1,7,6,3,1,1,9,1,5,4}  ,
                {1,1,1,8,8,8,8,1,7,6} ,
                {1,1,1,1,1,9,1,1,1,3},
                {1,1,3,1,1,1,1,1,1,1}
            };
        }
```

Segmen Program 4.8 *Source code Map*(lanjutan)

```
        float startX = topX;
        float startY = topY;
        for (int i=0;i<sizeMapY;i++)
        {
            startX = topX;
            for (int j=0;j<sizeMapX;j++)
            {

                GameObject go = Instantiate(tile);
                goMap [i, j] = go;
                go.transform.position = new Vector3(startX, startY, ti
le.transform.position.z);

                checktouch ct = go.GetComponent<checktouch> ();
                ct.x = j;
                ct.y = i;
                startX += lebarTileX;
                /*
                if (i == j) {
                    SpriteRenderer sr = go.GetComponent<SpriteRenderer
> ();

                    sr.sprite = Resources.Load<Sprite> ("tile2");
                }*/
            }
            startY -= lebarTileY;

            //Debug.Log("i "+i+", go : "+go.transform.position.x);

        }
```

4.6 *Unit*

Fungsi ini digunakan untuk pembuatan *unit*, menentukan *skill unit* dan juga digunakan untuk mengeset status awal dari *unit* setiap *unit*. Fungsi ini dipanggil ketika *player* memilih *unit* yang akan digunakan.

Segmen Program 4.9 *Source code Unit*

```
public character GetData(int tipe,int level)
    {
        character c = new character ();
        if (tipe == 0) {
            c.nama = "Kapal Api";
            c.hp = 100;
            c.agi = 10;
            c.def = 20;
            c.atk = 300;
```

Segmen Program 4.9 *Source code Unit* (lanjutan)

```
            c.skill1 = "Normal Attack";
            c.skill2 = "Torpedo Attack";
        }
        else if (tipe == 1) {
            c.nama = "Kapal B";
            c.hp = 75;
            c.agi = 20;
            c.def = 10;
            c.atk = 50000;
            c.skill1 = "Normal Attack";
            c.skill2 = "Raise Attack";
            c.skill3 = "Heal Random";
        }
        else if (tipe == 2) {
            c.nama = "Kapal C";
            c.hp = 150;
            c.agi = 5;
            c.def = 25;
            c.atk = 50000;

            c.skill1 = "Raise Agility";
            c.skill2 = "Raise Defend";
            c.skill3 = "Normal Attack";
        }
        else if (tipe == 3) {
            c.hp = 125;
            c.agi = 15;
            c.def = 5;
            c.atk = 308;

            c.skill1 = "Normal Attack";
            c.skill2 = "Fire Bomb";
        }
        else if (tipe == 4) {
            c.hp = 175;
            c.agi = 15;
            c.def = 5;
            c.atk = 250;

            c.skill1 = "Defend Position";
        }

    }
```

4.7 *Singleplayer*

Fungsi ini digunakan ketika *user* memanggil menu *singleplayer*. Dimana fungsi ini yang mengatur semua jalannya *battle* dari single *player*. Fungsi ini

mengatur dari pemanggilan *skill* saat *battle*, pengecekan attack ke arah musuh, dan masih banyak lagi.

Segment Program 4.10 *Source code Singleplayer*

```
public void checkSkill(string skill)
    {
        if (skill.Equals ("Heal All")) {
            for (int i = 0; i < lPlayer.Count; i++) {
                character c = lPlayer [i];
                c.currentHP += 10;
            }
            keteranganBattle.text = caktif.nama + " using heal all";
            finishturn ();
            /*
            int damage = caktif.atk - lEnemy [musuh].def;
            keteranganBattle.text = caktif.nama +" attacking "+ lEnemy
[musuh].nama+ " damage "+damage;
            lEnemy [musuh].currentHP = lEnemy [musuh].currentHP - dama
ge;
            lEnemy [musuh].cekDead ();
            lEnemy [musuh].displayInfo ();
            inputcommand = false;
            choosemusuh = false;
            caktif.turnNow = 0;

            caktif = null;*/
        } else if (skill.Equals ("Heal Random")) {
            List<character> lifeCharacter = getLifeCharacter ();
            if (lifeCharacter.Count > 0) {

                int randomPlayer = Random.Range (0, lifeCharacter.Coun
t);
                keteranganBattle.text = lifeCharacter [randomPlayer].n
ama + " healed";
                lifeCharacter [randomPlayer].currentHP += 3000;
                finishturn ();
            }

        }
        else if (skill.Equals ("Raise Attack")) {
            for (int i = 0; i < lPlayer.Count; i++) {
                //character c = lPlayer [i];
                lPlayer [i].atk += 30;
                //c.atk += 30;;
            }
            keteranganBattle.text = caktif.nama + " using raise attack
";
            finishturn ();
        }
        else if (skill.Equals ("Fire Bomb")) {
```

Segment Program 4.10 *Source code Singleplayer* (lanjutan)

```
            for (int i = 0; i < lEnemy.Count; i++) {
                lEnemy [i].currentHP=    lEnemy [i].currentHP- 100;
                lEnemy [i].displayInfo ();
            }
            keteranganBattle.text = caktif.nama + " using firebomb";
            finishturn ();
        }
        else if (skill.Equals ("Raise Defend")) {

            for (int i = 0; i < lPlayer.Count; i++) {
                lPlayer [i].def+=30;
                lPlayer [i].displayInfo ();
            }
            keteranganBattle.text = caktif.nama + " using firebomb";
            finishturn ();
        }
        else if (skill.Equals ("Defend Position")) {

            //for (int i = 0; i < lPlayer.Count; i++) {
                caktif.def*=2;
            //    lPlayer [i].displayInfo ();
            //}
            keteranganBattle.text = caktif.nama + " using firebomb";
            finishturn ();
        }
        else {
            skillUsage = skill;
            choosemusuh = true;
            setInvisibleButton ();
            keteranganBattle.text="Pilih musuh anda";
        }
    }

public void setMusuh(int musuh)
    {
        if (choosemusuh) {
            int damage = caktif.atk - lEnemy [musuh].def;
            if (damage < 0) {
                damage = 10;
            }
            if (skillUsage.Equals ("Dual Cannon Shot")) {
                damage = damage * 2;
            } else if (skillUsage.Equals ("Torpedo Attack")) {
                damage = damage * 3;
                caktif.currentHP = caktif.currentHP - 100;
                caktif.displayInfo ();
            }
            keteranganBattle.text = caktif.nama +" attacking "+ lEnemy
[musuh].nama+ " damage "+damage;
            lEnemy [musuh].currentHP = lEnemy [musuh].currentHP - dama
ge;
```

Segment Program 4.10 *Source code Singleplayer*(lanjutan)

```
            lEnemy [musuh].cekDead ();
            lEnemy [musuh].displayInfo ();
            inputcommand = false;
            choosemusuh = false;
            caktif.turnNow = 0;
            caktif = null;
        }
```

4.8 *Multiplayer*

Fungsi *Multiplayer* ini digunakan Saat *user* memilih menu *multiplayer* dimana fungsi ini akan memproses semua langkah yang terjadi dibagian *multiplayer*. Seperti mengatur kondisi pengecekan menang, mengatur *turn*, mengatur serangan musuh.

Segment Program 4.11 *Source code Multiplayer*

```
void setDataBattle(IDictionary<string, object> ibattle)
    {
        if (ibattle.ContainsKey("winner"))
        {
            if (ibattle["winner"].Equals(dataPlayer["Key"]))
            {
                checkmenangs = "ok";
            }
            else
            {
                checkmenangs = "no";
            }
        }
        if (dataPlayer["Server"].Equals("Ok"))
        {
            string commandInput = ibattle["commandInput"] + "";
            if (!commandInput.Equals(""))
            {
                string inputEnemy = ibattle["commandInput"] + "";
                string[] dataInput = inputEnemy.Split(',');
                ibattle["commandInput"] = "";

                inputcommand = false;
                caktif = null;

                int skill = int.Parse(dataInput[0]);
                int idx = int.Parse(dataInput[1]);

                int damage = lEnemy[idxEnemy].atk - lPlayer[idx].def;

                ibattle["keterangan"] = lEnemy[idxEnemy].nama + " atta
cking " + lPlayer[idx].nama + " damage " + damage;
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
lPlayer[idx].currentHP -= damage;
                lEnemy[idxEnemy].turnNow = 0;

                sinkronData();
            }
        }
        else
        {
            if (!inputcommand)
            {
                string turns = ibattle["turnNow"] + "";
                if (!turns.Equals(""))
                {
                    int turn = int.Parse(turns);
                    lturnx = turn;
                }
            }
        }
        if (ibattle["user1"].Equals(dataPlayer["Key"]))
        {
            lPlayer[0].nama = "Unit " + ibattle["charUsera1"];
            lPlayer[0].tipe = ibattle["charUsera1"] + "";

            lPlayer[1].tipe = ibattle["charUserb1"] + "";
            lPlayer[2].tipe = ibattle["charUserc1"] + "";

            lEnemy[0].tipe = ibattle["charUsera2"] + "";
            lEnemy[1].tipe = ibattle["charUserb2"] + "";
            lEnemy[2].tipe = ibattle["charUserc2"] + "";
            if (!initData)
            {
                if (dataPlayer["Server"].Equals("Ok"))
                {
                    initData = true;
                    setDataPlayer(0, ibattle["charUsera1"] + "", dataP
layer["level"] + "");
                    setDataPlayer(1, ibattle["charUserb1"] + "", dataP
layer["level"] + "");
                    setDataPlayer(2, ibattle["charUserc1"] + "", dataP
layer["level"] + "");

                    setDataEnemy(0, ibattle["charUsera2"] + "", dataEn
emy["level"] + "");
                    setDataEnemy(1, ibattle["charUserb2"] + "", dataEn
emy["level"] + "");
                    setDataEnemy(2, ibattle["charUserc2"] + "", dataEn
emy["level"] + "");

                    lPlayer[0].currentHP = int.Parse(ibattle["hpa1"] +
 "");
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
                    lPlayer[0].hp = int.Parse(ibattle["mhpa1"] + "");
                    lPlayer[1].currentHP = int.Parse(ibattle["hpb1"] +
 "");
                    lPlayer[1].hp = int.Parse(ibattle["mhpb1"] + "");
                    lPlayer[2].currentHP = int.Parse(ibattle["hpc1"] +
 "");
                    lPlayer[2].hp = int.Parse(ibattle["mhpc1"] + "");

                    lEnemy[0].currentHP = int.Parse(ibattle["hpa2"] +
"");
                    lEnemy[0].hp = int.Parse(ibattle["mhpa2"] + "");
                    lEnemy[1].currentHP = int.Parse(ibattle["hpb2"] +
"");
                    lEnemy[1].hp = int.Parse(ibattle["mhpb2"] + "");
                    lEnemy[2].currentHP = int.Parse(ibattle["hpc2"] +
"");
                    lEnemy[2].hp = int.Parse(ibattle["mhpc2"] + "");

                    initSprite = true;
                }
            }
            if (!dataPlayer["Server"].Equals("Ok"))
            {
                lPlayer[0].currentHP = int.Parse(ibattle["hpa1"]+ "");
                lPlayer[0].hp = int.Parse(ibattle["mhpa1"] + "");
                lPlayer[0].turnNow = int.Parse(ibattle["speeda1"]+"");

                lPlayer[1].currentHP = int.Parse(ibattle["hpb1"]+ "");
                lPlayer[1].hp = int.Parse(ibattle["mhpb1"] + "");
                lPlayer[1].turnNow = int.Parse(ibattle["speedb1"]+"");

                lPlayer[2].currentHP = int.Parse(ibattle["hpc1"]+ "");
                lPlayer[2].hp = int.Parse(ibattle["mhpc1"] + "");
                lPlayer[2].turnNow = int.Parse(ibattle["speedc1"]+"");

                lEnemy[0].currentHP = int.Parse(ibattle["hpa2"] + "");
                lEnemy[0].hp = int.Parse(ibattle["mhpa2"] + "");
                lEnemy[0].turnNow = int.Parse(ibattle["speeda2"]+ "");

                lEnemy[1].currentHP = int.Parse(ibattle["hpb2"] + "");
                lEnemy[1].hp = int.Parse(ibattle["mhpb2"] + "");
                lEnemy[1].turnNow = int.Parse(ibattle["speedb2"]+ "");

                lEnemy[2].currentHP = int.Parse(ibattle["hpc2"] + "");
                lEnemy[2].hp = int.Parse(ibattle["mhpc2"] + "");
                lEnemy[2].turnNow = int.Parse(ibattle["speedc2"]+ "");
            }
            lPlayer[1].nama = "Unit " + ibattle["charUserb1"];
            lPlayer[2].nama = "Unit " + ibattle["charUserc1"];
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
            lEnemy[0].nama = "Unit " + ibattle["charUsera2"];
            lEnemy[1].nama = "Unit " + ibattle["charUserb2"];
            lEnemy[2].nama = "Unit " + ibattle["charUserc2"];
        }
        else
        {
            lEnemy[0].tipe = ibattle["charUsera1"] + "";
            lEnemy[1].tipe = ibattle["charUserb1"] + "";
            lEnemy[2].tipe = ibattle["charUserc1"] + "";
            lPlayer[0].tipe = ibattle["charUsera2"] + "";
            lPlayer[1].tipe = ibattle["charUserb2"] + "";
            lPlayer[2].tipe = ibattle["charUserc2"] + "";
            if (!initData)
            {
                if (dataPlayer["Server"].Equals("Ok"))
                {
                    initData = true;
                    setDataEnemy(0, ibattle["charUsera1"] + "", dataPl
ayer["level"] + "");
                    setDataEnemy(1, ibattle["charUserb1"] + "", dataPl
ayer["level"] + "");
                    setDataEnemy(2, ibattle["charUserc1"] + "", dataPl
ayer["level"] + "");


                    setDataPlayer(0, ibattle["charUsera2"] + "", dataE
nemy["level"] + "");
                    setDataPlayer(1, ibattle["charUserb2"] + "", dataE
nemy["level"] + "");
                    setDataPlayer(2, ibattle["charUserc2"] + "", dataE
nemy["level"] + "");


                    lEnemy[0].currentHP = int.Parse(ibattle["hpa1"] +
"");
                    lEnemy[0].hp = int.Parse(ibattle["mhpa1"] + "");
                    lEnemy[1].currentHP = int.Parse(ibattle["hpb1"] +
"");
                    lEnemy[1].hp = int.Parse(ibattle["mhpb1"] + "");
                    lEnemy[2].currentHP = int.Parse(ibattle["hpc1"] +
"");
                    lEnemy[2].hp = int.Parse(ibattle["mhpc1"] + "");
                    lPlayer[0].currentHP = int.Parse(ibattle["hpa2"] +
 "");
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
                    lPlayer[0].hp = int.Parse(ibattle["mhpa2"] + "");
                    lPlayer[1].currentHP = int.Parse(ibattle["hpb2"] +
 "");
                    lPlayer[1].hp = int.Parse(ibattle["mhpb2"] + "");
                    lPlayer[2].currentHP = int.Parse(ibattle["hpc2"] +
 "");
                    lPlayer[2].hp = int.Parse(ibattle["mhpc2"] + "");
                }
            }
            if (!dataPlayer["Server"].Equals("Ok"))
            {
                lEnemy[0].currentHP = int.Parse(ibattle["hpa1"] + "");
                lEnemy[0].hp = int.Parse(ibattle["mhpa1"] + "");
                lEnemy[0].turnNow = int.Parse(ibattle["speeda1"] + "")
;

                lEnemy[1].currentHP = int.Parse(ibattle["hpb1"] + "");
                lEnemy[1].hp = int.Parse(ibattle["mhpb1"] + "");
                lEnemy[1].turnNow = int.Parse(ibattle["speedb1"]+ "");

                lEnemy[2].currentHP = int.Parse(ibattle["hpc1"] + "");
                lEnemy[2].hp = int.Parse(ibattle["mhpc1"] + "");
                lEnemy[2].turnNow = int.Parse(ibattle["speedc1"]+ "");

                lPlayer[0].currentHP = int.Parse(ibattle["hpa2"]+ "");
                lPlayer[0].hp = int.Parse(ibattle["mhpa2"] + "");
                lPlayer[0].turnNow = int.Parse(ibattle["speeda2"]+"");

                lPlayer[1].currentHP = int.Parse(ibattle["hpb2"]+ "");
                lPlayer[1].hp = int.Parse(ibattle["mhpb2"] + "");
                lPlayer[1].turnNow = int.Parse(ibattle["speedb2"]+"");
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
                lPlayer[2].currentHP = int.Parse(ibattle["hpc2"]+ "");
                lPlayer[2].hp = int.Parse(ibattle["mhpc2"] + "");
                lPlayer[2].turnNow = int.Parse(ibattle["speedc2"]+"");
            }
            if (!dataPlayer["Server"].Equals("Ok"))
            {
                lEnemy[0].currentHP = int.Parse(ibattle["hpa1"] + "");
                lEnemy[0].hp = int.Parse(ibattle["mhpa1"] + "");
                lEnemy[0].turnNow = int.Parse(ibattle["speeda1"] + "")
;

                lEnemy[1].currentHP = int.Parse(ibattle["hpb1"] + "");
                lEnemy[1].hp = int.Parse(ibattle["mhpb1"] + "");
                lEnemy[1].turnNow = int.Parse(ibattle["speedb1"]+ "");

                lEnemy[2].currentHP = int.Parse(ibattle["hpc1"] + "");
                lEnemy[2].hp = int.Parse(ibattle["mhpc1"] + "");
                lEnemy[2].turnNow = int.Parse(ibattle["speedc1"]+ "");

                lPlayer[0].currentHP = int.Parse(ibattle["hpa2"]+ "");
                lPlayer[0].hp = int.Parse(ibattle["mhpa2"] + "");
                lPlayer[0].turnNow = int.Parse(ibattle["speeda2"]+"");

                lPlayer[1].currentHP = int.Parse(ibattle["hpb2"]+ "");
                lPlayer[1].hp = int.Parse(ibattle["mhpb2"] + "");
                lPlayer[1].turnNow = int.Parse(ibattle["speedb2"]+"");

                lPlayer[2].currentHP = int.Parse(ibattle["hpc2"]+ "");
                lPlayer[2].hp = int.Parse(ibattle["mhpc2"] + "");
                lPlayer[2].turnNow = int.Parse(ibattle["speedc2"]+"");
            }


            lEnemy[0].nama = "Unit " + ibattle["charUsera1"];
```

Segment Program 4.11 *Source code Multiplayer*(lanjutan)

```
            lEnemy[1].nama = "Unit " + ibattle["charUserb1"];
            lEnemy[2].nama = "Unit " + ibattle["charUserc1"];
            lPlayer[0].nama = "Unit " + ibattle["charUsera2"];
            lPlayer[1].nama = "Unit " + ibattle["charUserb2"];
            lPlayer[2].nama = "Unit " + ibattle["charUserc2"];
        }
        ctriterasi++;
        if (ibattle.ContainsKey("keterangan"))
        {
            keteranganA = ibattle["keterangan"] + "";
        }
```