

## LAMPIRAN

### Lampiran 1: Source Code

```
Asset > Scripts > Entity > Singletons > Authentication > AuthenticationManager.cs > {} Authentication > A
1 using System.Collections;
2 using System.Collections.Generic;
3 using System.Text.RegularExpressions;
4 using System.Threading.Tasks;
5 using UnityEngine;
6 using UnityEngine.UI;
7 using Firebase.Auth;
8 using TMPro;
9 using Persistence;
10
11 namespace Authentication
12 {
13     1 reference
14     public class AuthenticationManager : Singleton<AuthenticationManager>
15     {
16         [Header("Login")]
17         1 reference
18         [SerializeField] private TMP_InputField loginEmailText;
19         1 reference
20         [SerializeField] private TMP_InputField loginPasswordText;
21         1 reference
22         [SerializeField] private Button loginButton;
23         [Header("Register")]
24         1 reference
25         [SerializeField] private TMP_InputField registerEmailText;
26         1 reference
27         [SerializeField] private TMP_InputField registerPasswordText;
28         1 reference
29         [SerializeField] private TMP_InputField registerVerifyPasswordText;
30         1 reference
31         [SerializeField] private Button registerButton;
32
33         3 references
34         private FirebaseAuth auth;
35
36         // Start is called before the first frame update
37         0 references
38         void Start()
39         {
40             auth = FirebaseAuth.DefaultInstance;
41             loginButton.onClick.AddListener(Login);
42             registerButton.onClick.AddListener(Register);
43         }
44     }
45 }
```



Hak cipta © Institut Bisnis dan Informatika Kwik Kian Gie



KWIK KIAN GIE  
SCHOOL OF BUSINESS

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik dan tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar IBIKKG.
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin IBIKKG.



```
1 reference
void Login()
{
    string email = loginEmailText.text;
    string password = loginPasswordText.text;

    // Validate text
    if (!IsValidEmail(email))
    {
        Debug.LogError("Email is invalid");
        return;
    }

    if (!IsValidPassword(password))
    {
        Debug.LogError("Password is invalid");
        return;
    }

    StartCoroutine(LoginCoroutine(email, password));
}
```

```
1 reference
void Register()
{
    string email = registerEmailText.text;
    string password = registerPasswordText.text;
    string verifyPassword = registerVerifyPasswordText.text;

    // Validate text
    if (!IsValidEmail(email))
    {
        Debug.LogError("Email is invalid");
        return;
    }

    if (!IsValidPassword(password))
    {
        Debug.LogError("Password is invalid");
        return;
    }

    if (verifyPassword != password)
    {
        Debug.LogError("Verify Password must match with Password");
        return;
    }

    StartCoroutine(RegisterCoroutine(email, password));
}
```

1. Ditarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik dan tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar IBIKKG.
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin IBIKKG.



```
1 reference
IEnumerator RegisterCoroutine(string email, string password)

    Task<FirebaseUser> task = auth.CreateUserWithEmailAndPasswordAsync(email, password);
    yield return new WaitUntil(() => task.IsCompleted);

    if (task.Exception != null)
    {
        yield break;
    }

    Debug.Log("Successfully registered");
}

2 references
bool IsValidEmail(string email)
{
    return Regex.IsMatch(email, @"^[^@\s]+@^[^@\s]+\.[^@\s]+$");
}

2 references
bool IsValidPassword(string password)
{
    return password.Length >= 8 && Regex.IsMatch(password, @"^[a-zA-Z0-9]+$");
}

using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Newtonsoft.Json;

namespace Persistence
{
    [Serializable]
    5 references
    public class PlayerData
    {
        2 references
        public string userID;
        1 reference
        public string username;
        2 references
        public int plasmids;
        3 references
        public Dictionary<string, bool> subjects;
    }

    [JsonConstructor]
    1 reference
    public PlayerData(string userID, string username, int plasmids, Dictionary<string, bool> subjects)
    {
        this.userID = userID;
        this.username = username;
        this.plasmids = plasmids;
        this.subjects = subjects;
    }
}

Hak Cipta Dilindungi Undang-Undang
```

Hak Cipta milik IBI KKG (Institut Bisnis dan Informatika Kwik Kian Gie)

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik dan tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar IBIKKG.
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin IBIKKG.



```
using Newtonsoft.Json;
using System;
using System.Collections;
using System.Collections.Generic;
using System.Threading.Tasks;
using UnityEngine;
using Firebase.Database;

namespace Persistence
{
    public class PersistenceManager : Singleton<PersistenceManager>
    {
        private DatabaseReference databaseReference;

        protected override void Awake()
        {
            base.Awake();
            DontDestroyOnLoad(gameObject);
        }

        // Start is called before the first frame update
        void Start()
        {
            // Set up the database reference
            databaseReference = FirebaseDatabase.DefaultInstance.GetReference("users");
        }

        public IEnumerator SavePlayerData()
        {
            PlayerData playerData = Player.Instance.playerData;

            // Convert the PlayerData object to a JSON string
            string json = JsonConvert.SerializeObject(playerData);

            // Save the JSON string to the database
            Task task = databaseReference.Child(playerData.userID).SetRawJsonValueAsync(json);

            // Wait for the database operation to complete
            yield return new WaitUntil(() => task.IsCompleted);

            // Check if the operation was successful
            if (task.Exception != null)
            {
                yield break;
            }
        }
    }
}
```

1. Ditarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik dan tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar IBIKKG.
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin IBIKKG.



```
1 reference
public IEnumerator LoadPlayerData(string userID)
{
    // Get the player data in json string from the Realtime Database
    Task<DataSnapshot> task = databaseReference.Child(userID).GetValueAsync();
    // Wait for the database operation to complete
    yield return new WaitUntil(() => task.IsCompleted);
    // Check if the operation was successful
    if (task.Exception != null)
        yield break;
    DataSnapshot snapshot = task.Result;
    // Check if the result of the operation is null (no data was found)
    if (snapshot.Value != null)
    {
        // Deserialize the snapshot's value as a PlayerData object using JsonConvert
        string json = snapshot.GetRawJsonValue();
        Player.Instance.playerData = JsonConvert.DeserializeObject<PlayerData>(json);
    }
    else
    {
        // Create new player data if there is none (user doesn't have player data in Realtime Database)
        string usernameTest = "New Player";
        int plasmidsTest = 100;
        Dictionary<string, bool> subjectsTest = new Dictionary<string, bool>
        {
            {"A001", true},
            {"D001", true},
            {"S001", true}
        };
        PlayerData playerData = new PlayerData(userID, usernameTest, plasmidsTest, subjectsTest);
        Player.Instance.playerData = playerData;
        StartCoroutine(SavePlayerData());
    }
    // Load to Menu Scene
    LoadingManager.Instance.LoadScene((int)SceneIndices.Menu);
}
```

Hak Cipta Dilindungi Undang-Undang

Hak Cipta Dilindungi IBI/KKG (Institusi Bisnis dan Informatika Kwik Kian Gie)

Institusi Bisnis dan Informatika Kwik Kian Gie

1. Ditarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik dan tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar IBI/KKG.
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin IBI/KKG.