



**EduLity**

**Elevate Education, Embrace Quality**

# **REPORT BOOK**

**Tugas Akhir Pemrograman Berorientasi Obyek**

**Program Studi Teknologi Informasi 2023**

Kelompok ASTERIA

Barbara Neanake A. (22/494495/TK/54238)

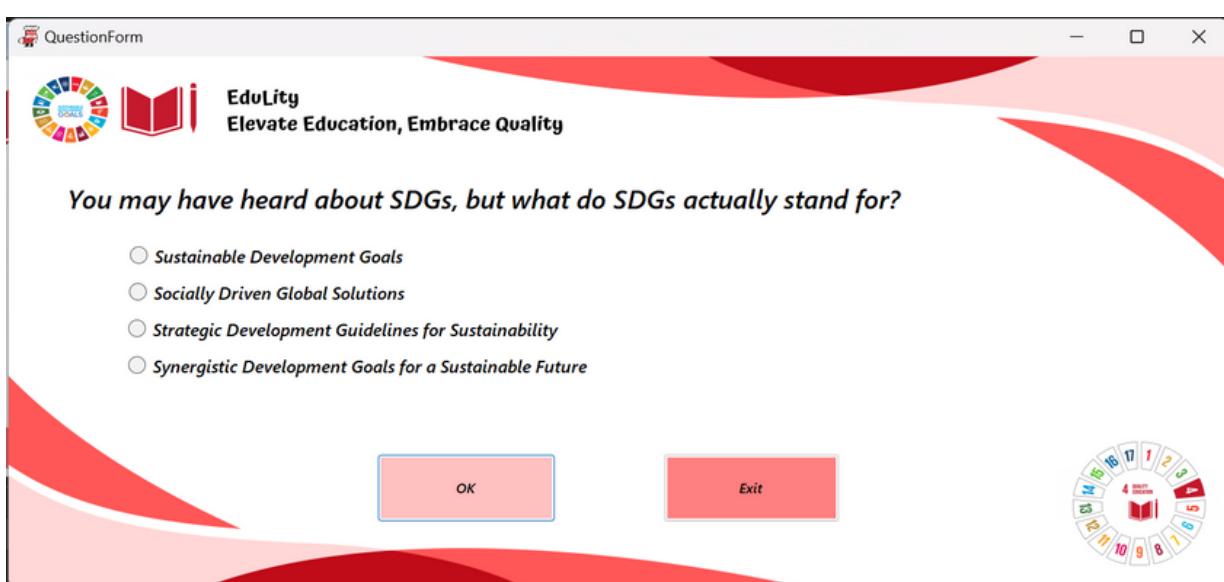
Christella Jesslyn D. (22/493149/TK/54003)



# Apa itu EduLity?

EduLity adalah aplikasi berbasis C# Windows Forms yang memberikan pengalaman pembelajaran inovatif dengan fokus pada Pendidikan Berkelanjutan (SDGs ke-4). Aplikasi ini dibentuk sebagai sarana peningkatan pengetahuan yang memberikan kesempatan pada pengguna untuk belajar melalui serangkaian kuis yang tidak hanya informatif tetapi juga menghibur.

EduLity, dikembangkan dengan manajemen data kuat melalui PostgreSQL, dirancang cermat dengan Microsoft Form dan diimplementasikan lancar melalui Visual Studio 2022. Dengan bahasa pemrograman C#, kami memastikan fungsionalitas efisien dan manajemen database tanpa hambatan. Sebagai representasi perpaduan teknologi dan pendidikan, EduLity bertujuan memfasilitasi lingkungan pembelajaran dinamis dan interaktif.





# Tujuan Hadirnya EduLity

## Empowering Quality Education Through Adaptive Quizzes

EduLity berkomitmen mendukung SDG 4, Pendidikan Berkualitas, melalui kuis adaptif yang informatif. Dengan fitur uniknya, EduLity bertujuan meningkatkan pemahaman pengguna terhadap tantangan dan peluang pendidikan, menjadikannya kontributor kunci dalam mencapai SDG4.

## Igniting Profound Awareness of SDGs

EduLity tidak hanya menyajikan kuis menghibur; tetapi juga menggali pemahaman tentang SDGs, khususnya SDG 4. EduLity mengajak pengguna berpartisipasi aktif & mendukung inisiatif peningkatan mutu pendidikan. EduLity bukan sekadar penyedia quiz, tapi kami penggerak perubahan.

## Innovative Program for OOP Final Project

Kami menciptakan EduLity untuk memenuhi tugas akhir Pemrograman Berorientasi Objek. Dengan pendekatan adaptif, desain kode yang unik, dan teknologi terkini, EduLity adalah jalan kami bereksplorasi di pemrograman berorientasi objek bersama pembelajaran global yang dinamis.





# Fitur Unggulan EduLity

## Quiz Berkualitas mengenai SDGs

- Kuis EduLity menawarkan pertanyaan berkualitas tinggi yang menyentuh inti Pendidikan Berkelanjutan (SDGs ke-4).
- Setiap pertanyaan didesain untuk merangsang pemikiran mendalam tentang isu-isu terkait SDGs.

## Feedback Interaktif

- Umpan balik instan dalam setiap pertanyaan, sehingga dapat meningkatkan pengalaman belajar.
- Memberikan motivasi dan bimbingan, mendorong pemahaman mendalam.

## Tampilan Rapi dan Menarik

- Interface yang rapi dan menarik untuk pengalaman pembelajaran yang menyenangkan.
- Desain intuitif untuk mempermudah navigasi dan memastikan pengguna dapat dengan mudah mengakses fitur pembelajaran.

## Recap, Scoring, & Explanation

- Fitur recap memberikan gambaran hasil kuis secara menyeluruh.
- Menampilkan peringkat pengguna, menambah elemen kompetitif.

## Kode Terstruktur & Mudah Dimodifikasi

- EduLity memprioritaskan kode yang terstruktur dan fleksibel, memudahkan pengembang dalam modifikasi tanpa mengurangi stabilitas aplikasi.

# Asteria Group

Tim Dibalik Lahirnya EduLity



**BARBARA Neanake Ajesti**  
22/494495/TK/54238



## 1. Creating UI/UX Design

- Crafting the complete UI/UX design for EduLity, ensuring a user-friendly experience.

## 2. Managing Database

- Overseeing the database operations using PostgreSQL for efficient data management.

## 3. Refactoring with Techniques

- Enhancing EduLity's codebase with advanced refactoring methods for improved structure.

## 4. Implementing Design Patterns

- Incorporating design patterns like Singleton, Observer, and Strategy for optimized functionality.

## 5. Developing Coding Entire Application

- (since the application is only available on Barbara's laptop). Creating FirstForm, RegistForm, LoginForm, MainForm, RecapForm, and ExplanationForm.

## 6. Creating User Control Elements

- Designing and implementing User Control elements for enhanced application behavior (sidebar, etc).

## 7. Creating Application Report

- Generating a comprehensive report with use cases, flowcharts, and detailed explanations.

## 8. Creating Presentation Video

- Crafting an engaging video showcasing EduLity's features and design approach.

## 9. Researching EduLity Question & Explanation

- Thoroughly researching and addressing questions to deepen understanding.

# Asteria Group

Tim dibalik lahirnya EduLity



Christella JESSLYN Dewantara  
22/493149/TK/54003

## 1. Coding Quiz Functionality (QuestionForm)

- Developing the codebase for EduLity's quiz module, focusing on the QuestionForm to enable seamless quiz execution.

## 2. Creating Group Name

- Generating meaningful group names, like "Asteria," inspired by themes or meanings relevant to each group's identity.

## 3. Creating Application Report

- Creating a concise report detailing EduLity's functionalities and technical aspects for user understanding and developer insights.



# EduLity

# Use Case Diagram

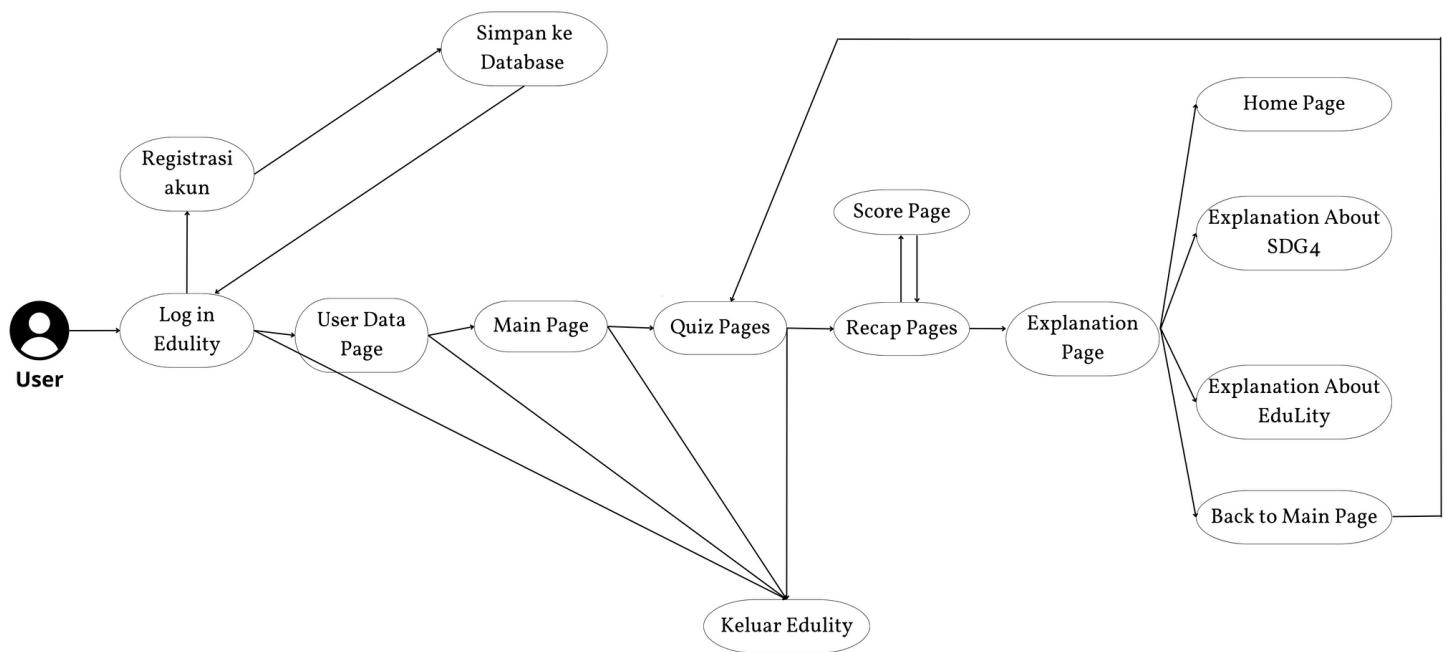


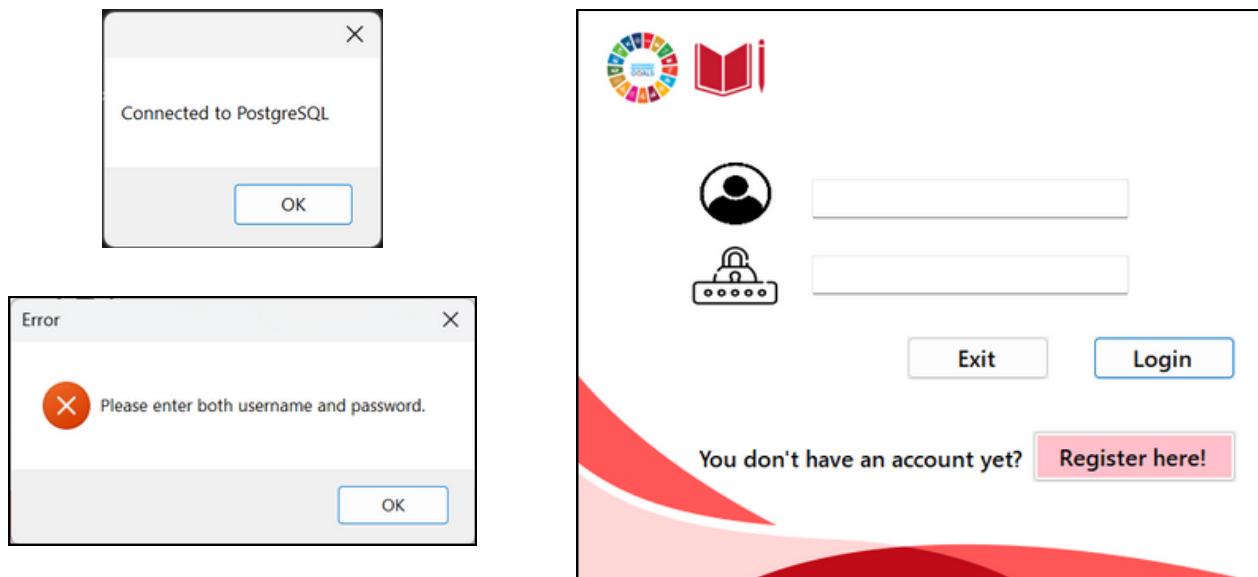
Diagram use case di atas menampilkan hubungan antara FirstForm (Login), RegistForm (Registrasi akun), LoginForm (User Data Page), MainForm (Main Page), QuestionForm (Quiz Pages), RecapForm (Recap Page), hingga ExplanationForm (Explanation Page), dan hubungan mereka dengan database. Diagram ini mencakup hubungan fungsional antar elemen-elemen tersebut dan kaitannya dengan basis data.



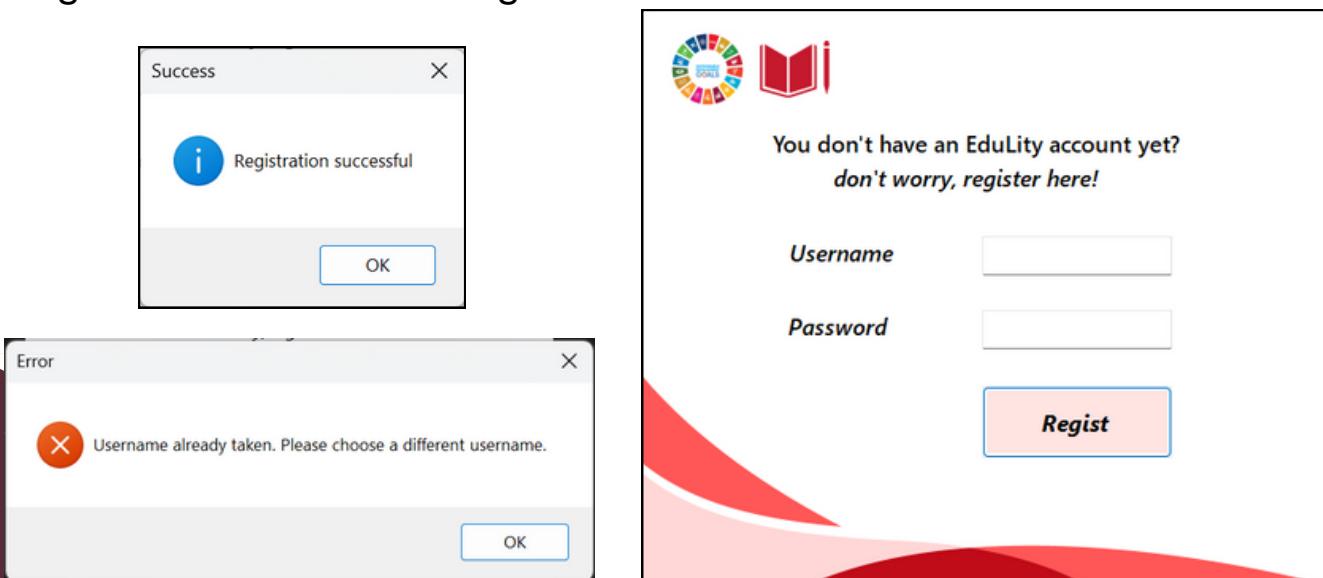
# EduLity

# User Scenario

1. Ketika code dijalankan, program menampilkan pop up konfirmasi telah terhubung ke PostgreSQL dan user dapat mengakses FirstForm untuk login dengan memasukkan username serta password.



2. Jika belum memiliki akun, user dapat melakukan registrasi melalui RegistForm dan kembali login.

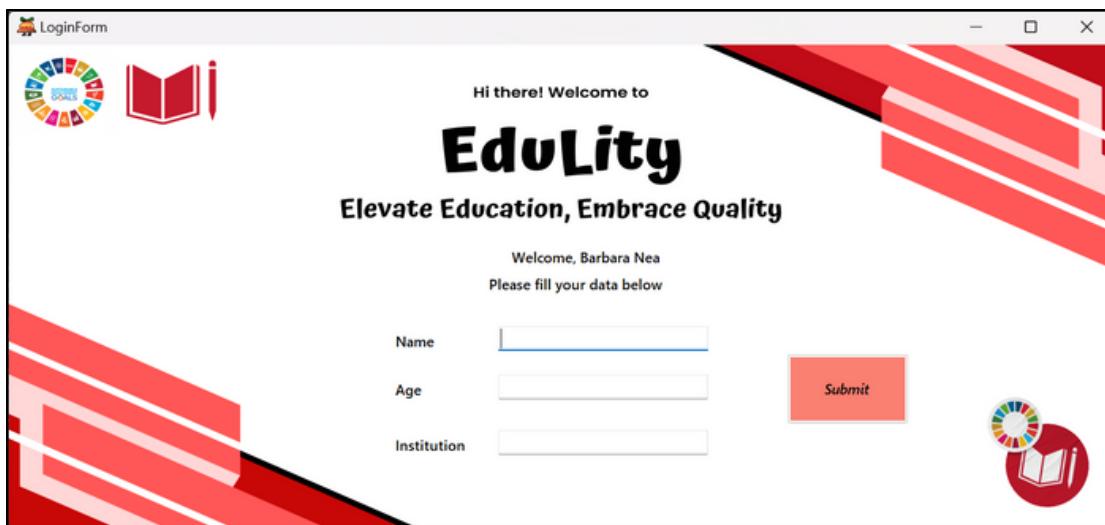




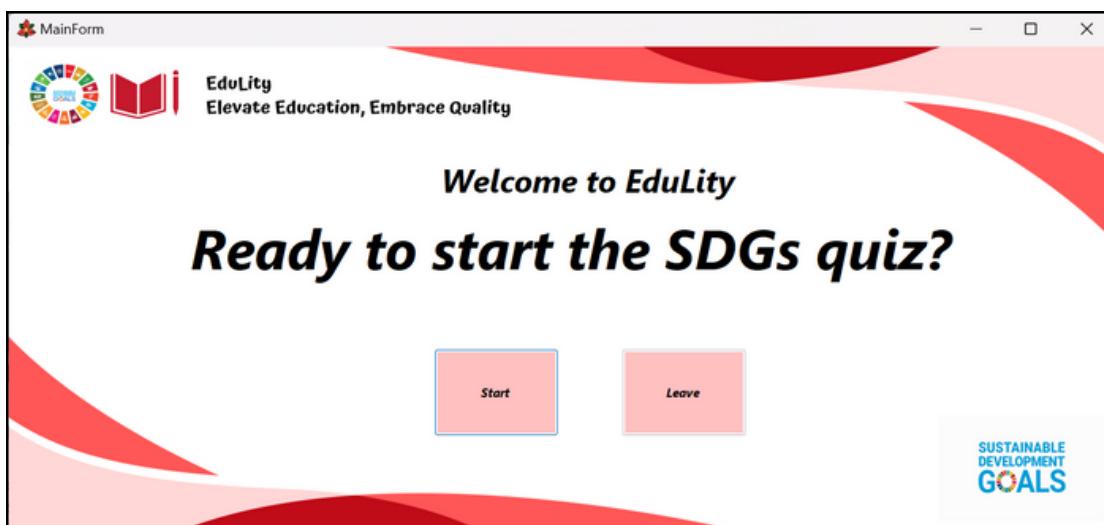
# EduLity

# User Scenario

3. Setelah berhasil login, user diarahkan ke LoginForm untuk mengisi data pribadi seperti nama, usia, dan instansi.



4. Saat menekan submit, MainForm muncul, meminta user untuk memastikan kesiapan mengikuti kuis.

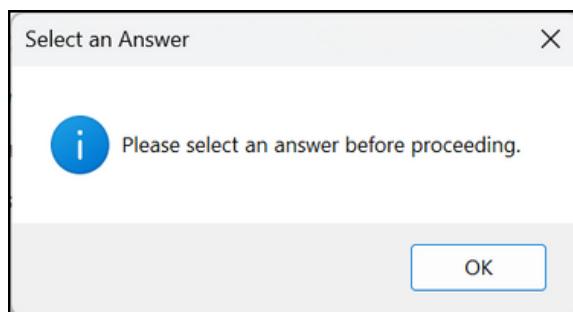
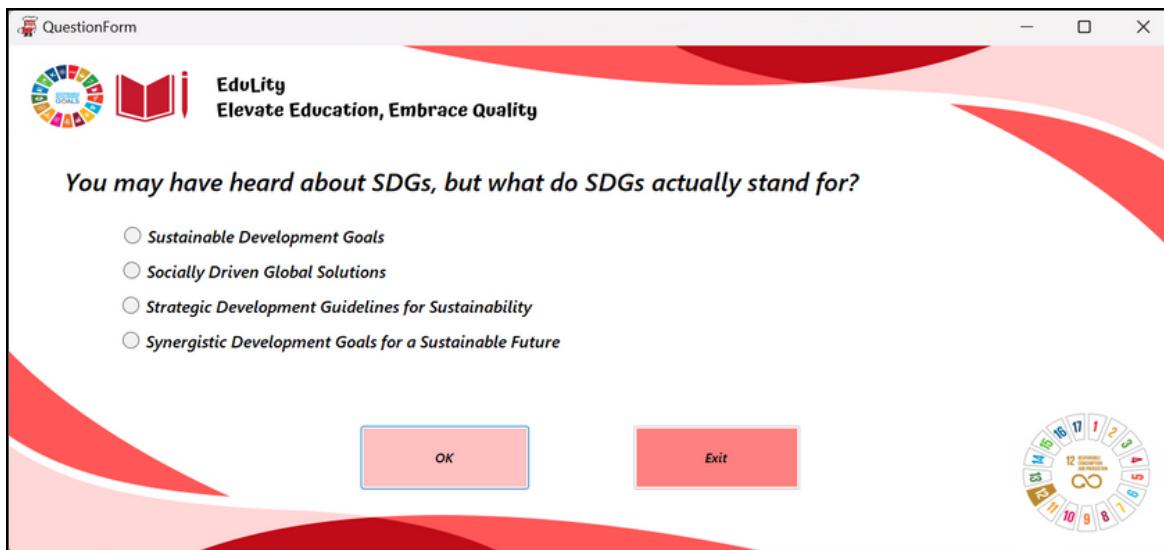




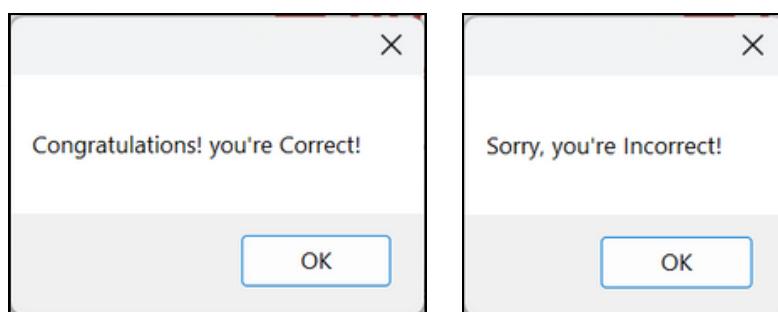
# EduLity

# User Scenario

5. Jika siap, user dapat menekan "Ready" untuk beralih ke QuestionForm dan menjawab pertanyaan kuis.



6. Feedback pop-up akan memberikan informasi setiap kali user memberikan jawaban.

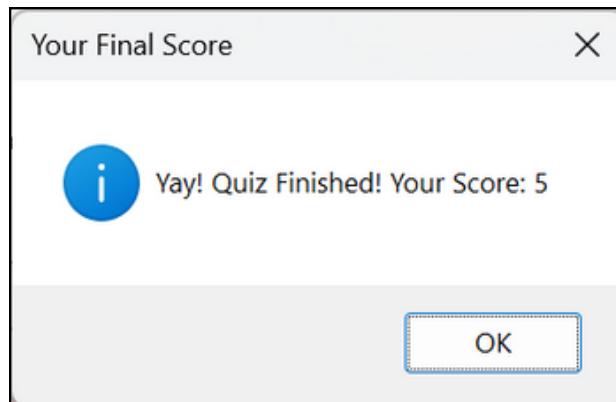
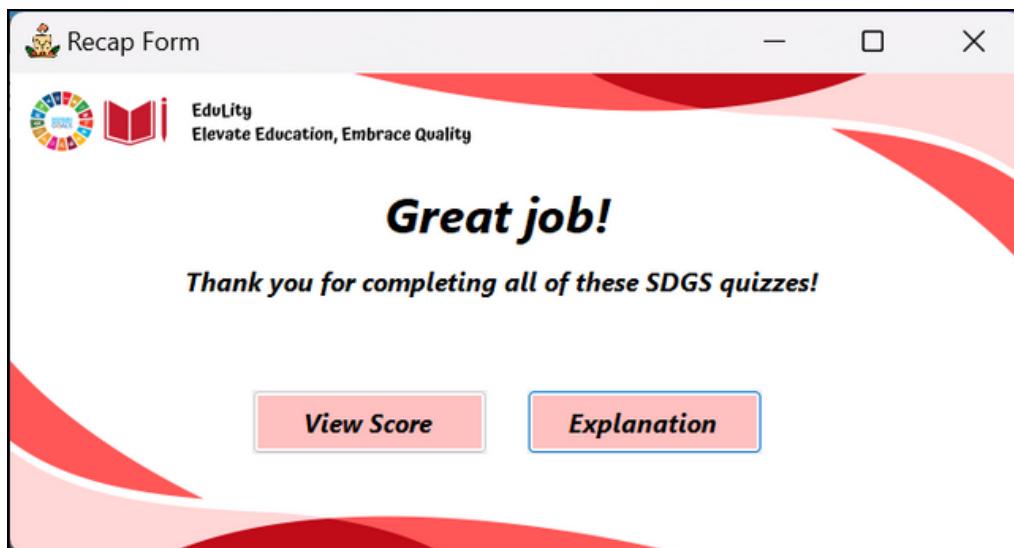




# EduLity

# User Scenario

- Setelah menyelesaikan QuestionForm, user otomatis dialihkan ke RecapForm untuk melihat skor quiz. Dari RecapForm, user dapat melihat skor akhir dengan menekan "View Score" dan membuka penjelasan dalam "Explanation".

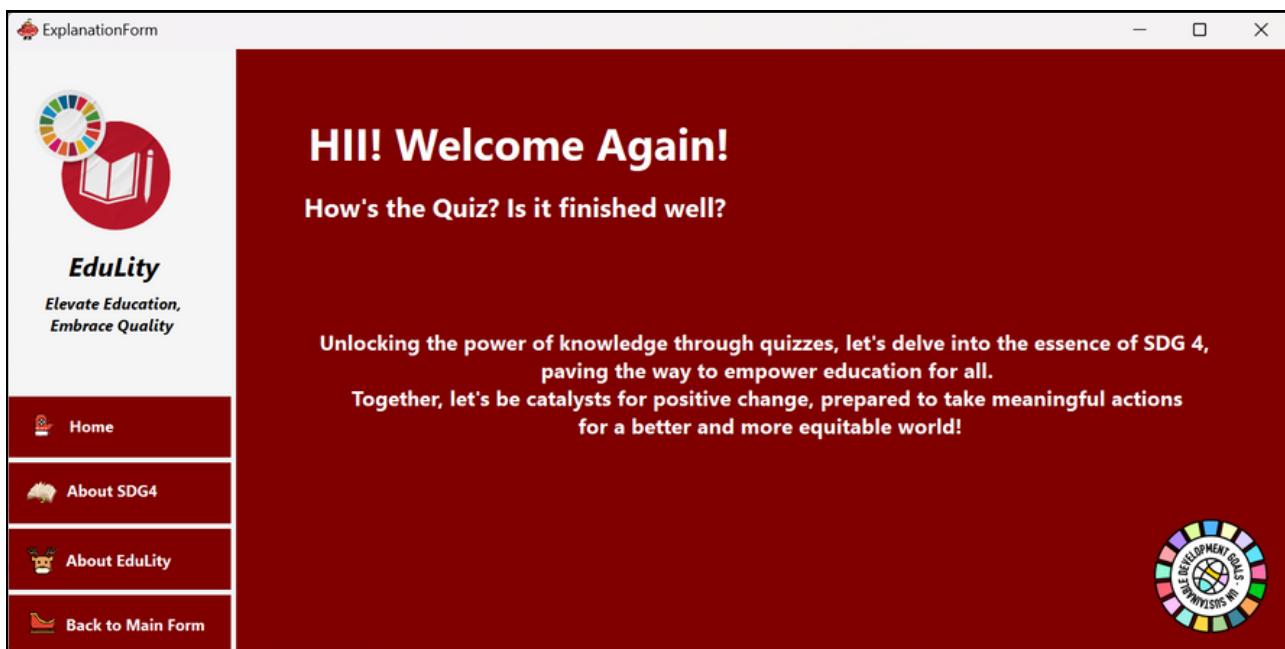




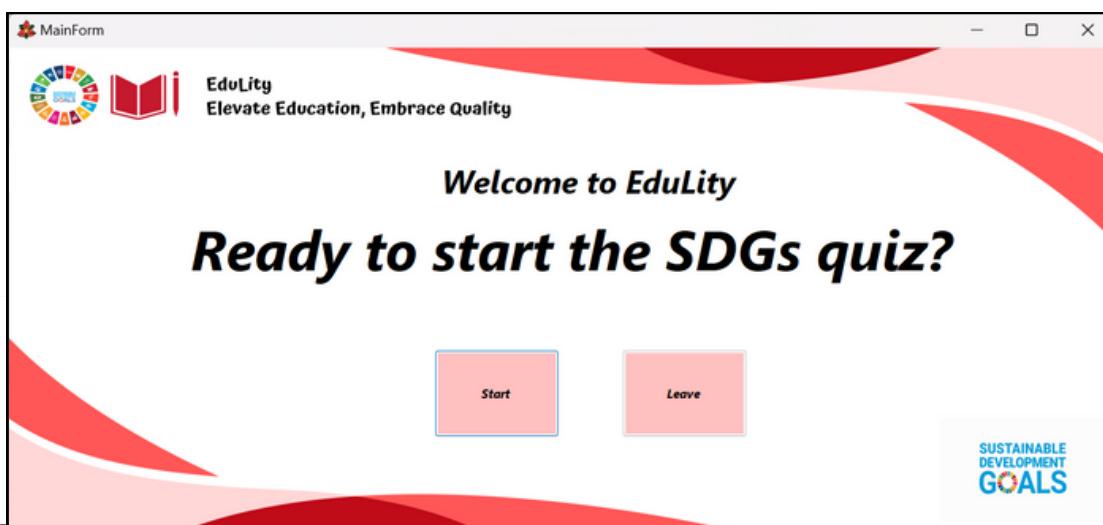
# EduLity

# User Scenario

8. ExplanationForm menyajikan sidebar dengan opsi untuk menjelajahi berbagai fitur seperti halaman utama, penjelasan SDG 4, penjelasan EduLity, dan opsi Quit untuk kembali ke MainForm.



9. Panel "Back to Main Form" pada ExplanationForm memungkinkan user untuk kembali ke MainForm, dan user dapat **keluar dari aplikasi** atau **mengulang kuis**.





# 3 Concept OOP

that we apply in EduLity

## 1 DATABASE

EduLity menggunakan PostgreSQL untuk menyimpan data pengguna, seperti username dan password. Struktur database melibatkan tabel pengguna, fungsi login, dan registrasi.

```
SQL Statistics Dependencies Dependents Processes DATAFIX.sql
postgres@PostgreSQL 11
No limit
History
drop existing function if it exists
FUNCTION IF EXISTS u_login(character varying, character varying)

drop existing table if it exists
TABLE IF EXISTS tbl_users;

create table contains data
TE TABLE tbl_users

username CHARACTER VARYING(150) NOT NULL PRIMARY KEY,
password CHARACTER VARYING(150)

insert some demo records
RT INTO tbl_users VALUES('admin', 'admin');
RT INTO tbl_users VALUES('user', '123');
RT INTO tbl_users VALUES('Barbara Nea', '2304');

elect all records from tbl_users
CT * FROM tbl_users;

re create function login
RE OR REPLACE FUNCTION u_login(_username CHARACTER VARYING, _password CHARACTER VARYING)
RETURNS INT AS
$0 of 0
```

## 2 REFACTORING

Kode EduLity disempurnakan dengan ekstraksi metode, properti, dan manajemen exception. Pemeliharaan dan kejelasan kode ditingkatkan melalui prinsip OOP.

```
RegistForm.cs [Design] QuestionForm.cs [Design]
EduLity.IQuestionObserver
namespace EduLity
{
    // Observer interface
    public interface IQuestionObserver
    {
        void UpdateSelectedAnswer(string selectedAnswer);
    }

    // Subject class
    public partial class QuestionForm : Form
    {
        private string selectedAnswer;
        private List<IQuestionObserver> observers = new List<IQuestionObserver>();

        public string SelectedAnswer
        {
            get { return selectedAnswer; }
            private set
            {
                if (selectedAnswer != value)
                    observers.ForEach(observer => observer.UpdateSelectedAnswer(value));
            }
        }
    }
}
```

## 3 DESIGN PATTERNS

Dalam EduLity, Design Pattern Observer digunakan pada QuestionForm untuk memisahkan subjek (form pertanyaan) dan pengamat (contoh pengamat), meningkatkan fleksibilitas dan pemeliharaan kode.

# 1 Database using PostgreSQL

## Implementasi Database pada EduLity menggunakan PostgreSQL

Dalam aplikasi EduLity, penggunaan database PostgreSQL diimplementasikan untuk menyimpan data pengguna, terutama informasi seperti username dan password. Berikut adalah penjelasan teknis mengenai sistem database PostgreSQL yang diimplementasikan:

### Struktur Tabel (tbl\_users)

```
7 -- Create table contains data
8 CREATE TABLE tbl_users
9 (
10     username CHARACTER VARYING(150) NOT NULL PRIMARY KEY,
11     password CHARACTER VARYING(150)
12 );
13
```

- username: kolom untuk menyimpan nama pengguna. Dijadikan PRIMARY KEY untuk memastikan keunikan setiap pengguna.
- password: kolom untuk menyimpan kata sandi / password pengguna.

### Fungsi Login (u\_login)

```
22 -- Create function login
23 CREATE OR REPLACE FUNCTION u_login(_username CHARACTER VARYING, _password CHARACTER VARYING)
24 RETURNS INT AS
25 $$
26 BEGIN
27     RETURN EXISTS (SELECT 1 FROM tbl_users WHERE username = _username AND password = _password)::INT;
28 END
29 $$
30 LANGUAGE plpgsql;
31
```

- Fungsi: Digunakan untuk memeriksa apakah kombinasi username dan password yang dimasukkan oleh pengguna valid.
- Return Value: Mengembalikan 1 jika login berhasil dan 0 jika gagal.

## Fungsi Registrasi (u\_register)

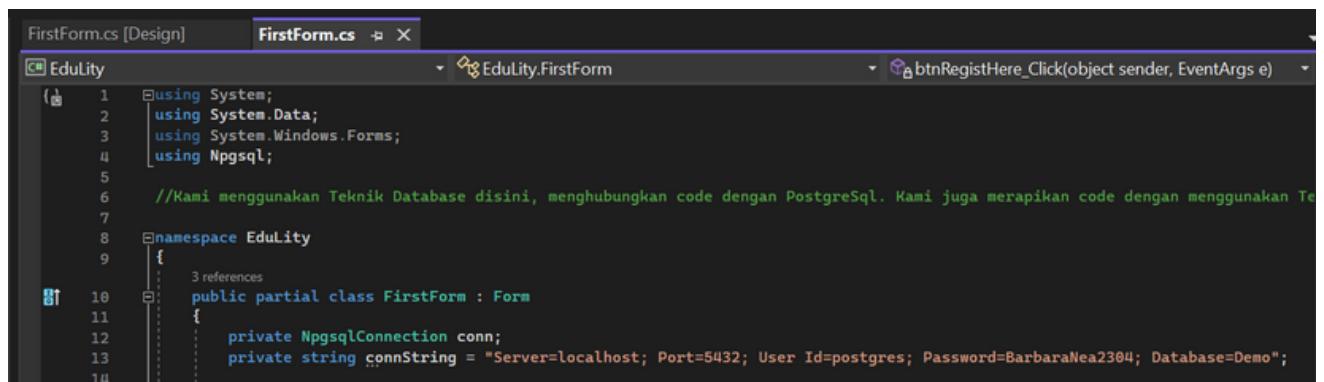
```

32 -- Create function for user registration
33 CREATE OR REPLACE FUNCTION u_register(_username CHARACTER VARYING, _password CHARACTER VARYING)
34 RETURNS VOID AS
35 $$
36 BEGIN
37     -- Check if the username already exists
38     IF NOT EXISTS (SELECT 1 FROM tbl_users WHERE username = _username) THEN
39         -- If username is available, insert the new user
40         INSERT INTO tbl_users(username, password) VALUES (_username, _password);
41     END IF;
42 END
43 $$;
44 LANGUAGE plpgsql;
45

```

- Fungsi: Digunakan untuk mendaftarkan pengguna baru ke dalam tabel `tbl_users`.
- Proses: Memeriksa apakah username sudah terdaftar. Jika belum, melakukan penyisipan data baru.

## Koneksi ke Database dalam FirstForm



```

1  using System;
2  using System.Data;
3  using System.Windows.Forms;
4  using Npgsql;
5
6  //Kami menggunakan Teknik Database disini, menghubungkan code dengan PostgreSQL. Kami juga merapikan code dengan menggunakan Te
7
8  namespace EduLity
9  {
10     public partial class FirstForm : Form
11     {
12         private NpgsqlConnection conn;
13         private string connString = "Server=localhost; Port=5432; User Id=postgres; Password=BarbaraNea2304; Database=Demo";
14     }

```

- Inisialisasi Koneksi: Membuat koneksi ke database PostgreSQL dengan parameter seperti server, port, user, password, dan nama database.

## Login Procedure dalam FirstForm

```

FirstForm.cs [Design] FirstForm.cs ✎ X
EduLity FirstForm btnRegistHere_Click
    72
    73     private int ExecuteLoginProcedure()
    74     {
    75         string sql = "select * from u_login(:_username, :_password)";
    76         using (NpgsqlCommand cmd = new NpgsqlCommand(sql, conn))
    77         {
    78             cmd.Parameters.AddWithValue("_username", txtUsername.Text);
    79             cmd.Parameters.AddWithValue("_password", txtPassword.Text);
    80             return (int)cmd.ExecuteScalar();
    81         }
    82     }
    83

```

- Eksekusi Login Procedure: Memanggil fungsi login pada database untuk memeriksa kevalidan login.

## Registrasi dalam RegistForm

```

RegistForm.cs [Design] FirstForm.cs [Design] RegistForm.cs ✎ X FirstForm.cs
EduLity RegistForm txtNewUsername
    40
    41
    42     private void btnRegist_Click(object sender, EventArgs e)
    43     {
    44         try
    45         {
    46             using (NpgsqlConnection conn = new NpgsqlConnection(connString))
    47             {
    48                 conn.Open();
    49
    50                 if (string.IsNullOrWhiteSpace(txtNewUsername.Text) || string.IsNullOrWhiteSpace(txtNewPassword.Text))...
    51
    52                 // Check if the username is already taken
    53                 string checkUsernameQuery = "SELECT COUNT(*) FROM tbl_users WHERE username = :_username";
    54                 using (NpgsqlCommand checkUsernameCmd = new NpgsqlCommand(checkUsernameQuery, conn))...
    55
    56                 // Insert new user into the database using the u_register function
    57                 string insertUserQuery = "SELECT u_register(:_username, :_password)";
    58                 using (NpgsqlCommand insertUserCmd = new NpgsqlCommand(insertUserQuery, conn))...
    59
    60             }
    61         }
    62         catch (Exception ex)
    63         {
    64             MessageBox.Show($"Error: {ex.Message}\nStackTrace: {ex.StackTrace}", "Something went wrong", MessageBoxButtons.OK, MessageBoxIcon.Error);
    65         }
    66     }
    67 }
    68
    69
    70
    71
    72
    73
    74
    75
    76
    77
    78
    79
    80
    81
    82
    83
    84
    85
    86
    87
    88
    89
    90
    91
    92
    93
    94
    95

```

- Registrasi Pengguna Baru: Memastikan koneksi dibuka sebelum melakukan proses registrasi pengguna baru menggunakan fungsi u\_register di database.

## Kesimpulan Implementasi Database dalam EduLity

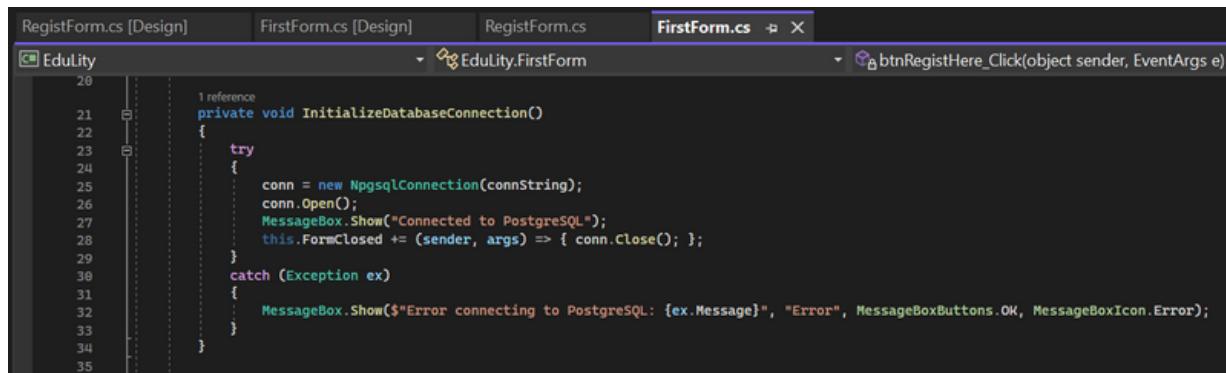
- Implementasi ini memastikan bahwa data pengguna tersimpan dengan aman dalam tabel PostgreSQL dan proses login/registrasi dapat dilakukan dengan baik melalui fungsi yang telah dibuat di database.

# Refactoring Implementations 2

## Refactoring dalam FirstForm

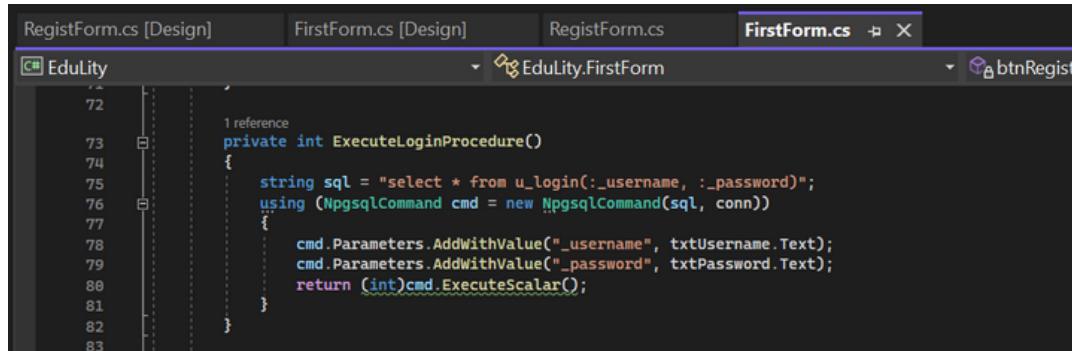
### Manajemen Koneksi Database:

Meningkatkan keterbacaan kode dengan mengenkapsulasi pengaturan dan penutupan koneksi database dalam metode terpisah (`InitializeDatabaseConnection`).



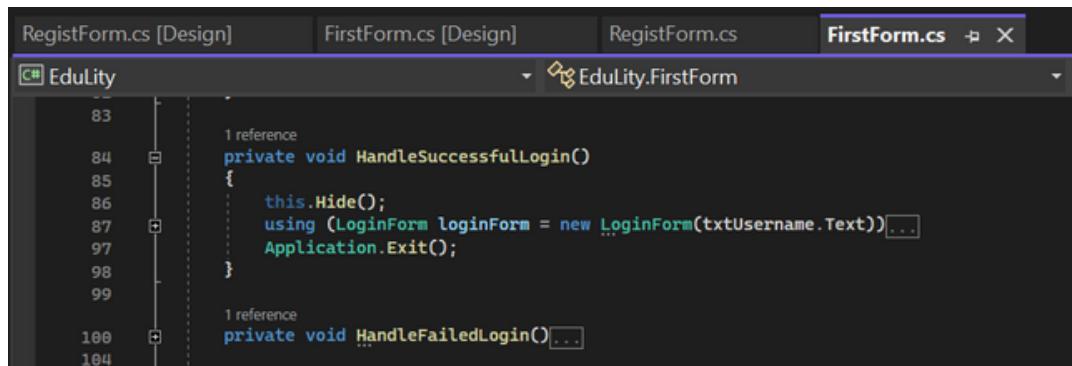
```
20
21     private void InitializeDatabaseConnection()
22     {
23         try
24         {
25             conn = new NpgsqlConnection(connString);
26             conn.Open();
27             MessageBox.Show("Connected to PostgreSQL");
28             this.FormClosed += (sender, args) => { conn.Close(); };
29         }
30         catch (Exception ex)
31         {
32             MessageBox.Show($"Error connecting to PostgreSQL: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
33         }
34     }
35 
```

**Eksekusi Login:** Mengekstrak logika eksekusi login ke dalam metode terpisah (`ExecuteLoginProcedure`) untuk kemudahan pemeliharaan.



```
72
73     private int ExecuteLoginProcedure()
74     {
75         string sql = "select * from u_login(:_username, :_password)";
76         using (NpgsqlCommand cmd = new NpgsqlCommand(sql, conn))
77         {
78             cmd.Parameters.AddWithValue("_username", txtUsername.Text);
79             cmd.Parameters.AddWithValue("_password", txtPassword.Text);
80             return (int)cmd.ExecuteScalar();
81         }
82     }
83 
```

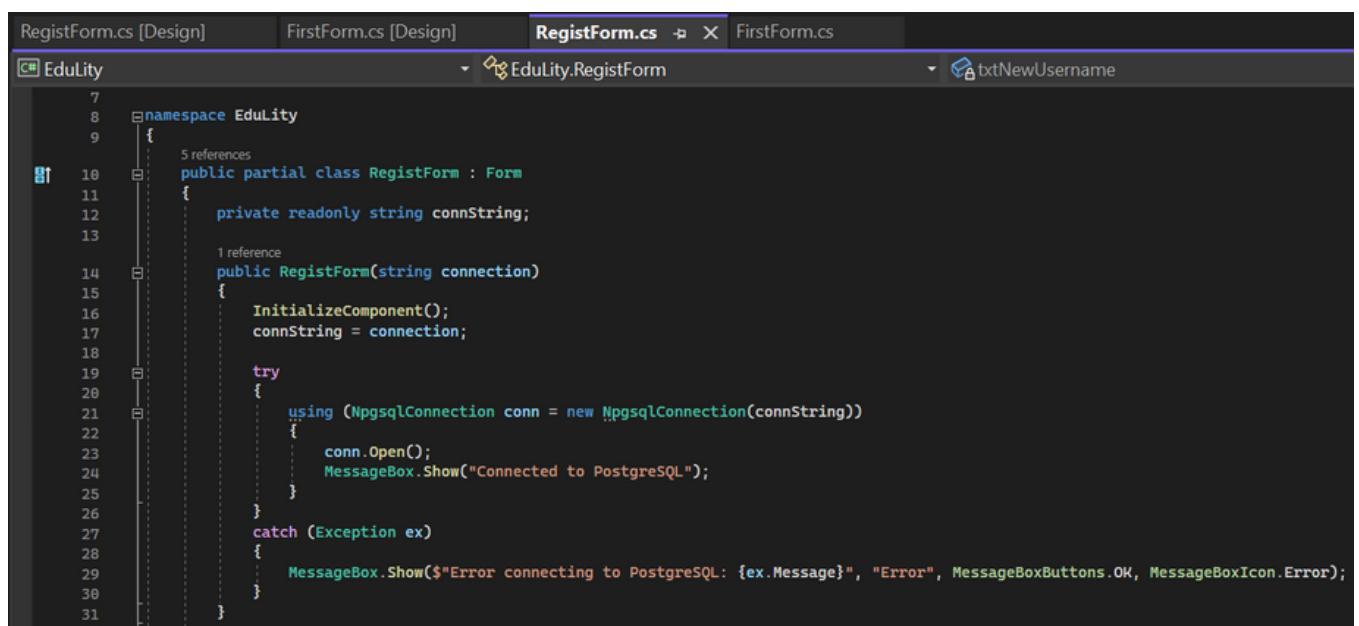
**Penanganan Login Sukses dan Gagal:** Modularisasi penanganan login sukses dan gagal ke dalam metode terpisah (`HandleSuccessfulLogin` dan `HandleFailedLogin`).



```
83
84     private void HandleSuccessfulLogin()
85     {
86         this.Hide();
87         using (LoginForm loginForm = new LoginForm(txtUsername.Text)...)
88             Application.Exit();
89     }
90
91     private void HandleFailedLogin()... 
```

## Refactoring dalam RegistForm

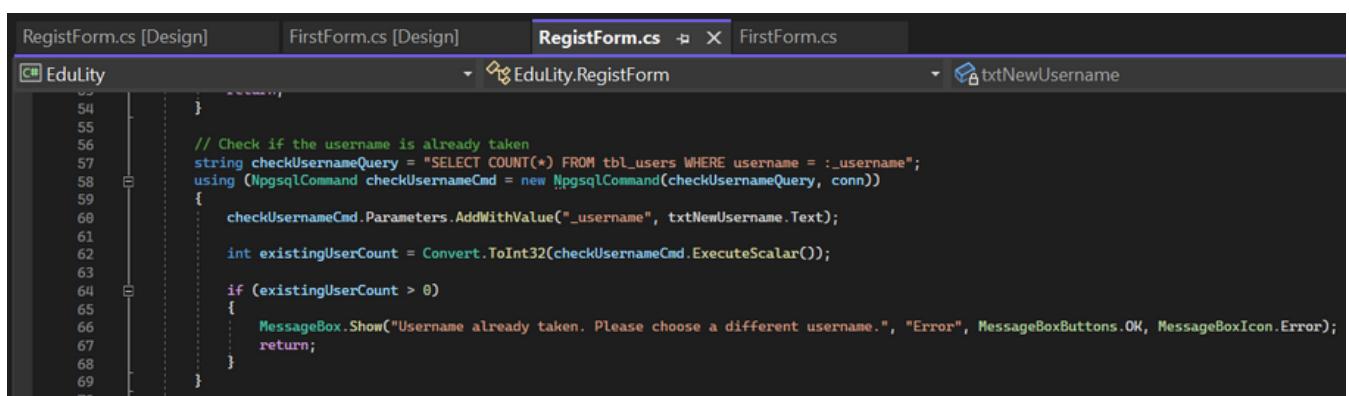
**Penanganan Koneksi Database:** Meningkatkan struktur kode dengan menggunakan pernyataan **using** untuk setup koneksi database, mengurangi redundansi, dan meningkatkan keterbacaan.



The screenshot shows the Visual Studio IDE with the 'RegistForm.cs' tab selected. A context menu is open over the connection string declaration, with the 'Refactor' option highlighted. The code editor displays the following C# code:

```
7
8     namespace EduLity
9     {
10         public partial class RegistForm : Form
11         {
12             private readonly string connString;
13
14             public RegistForm(string connection)
15             {
16                 InitializeComponent();
17                 connString = connection;
18
19                 try
20                 {
21                     using (NpgsqlConnection conn = new NpgsqlConnection(connString))
22                     {
23                         conn.Open();
24                         MessageBox.Show("Connected to PostgreSQL");
25                     }
26                 }
27                 catch (Exception ex)
28                 {
29                     MessageBox.Show($"Error connecting to PostgreSQL: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
30                 }
31             }
32         }
33     }
```

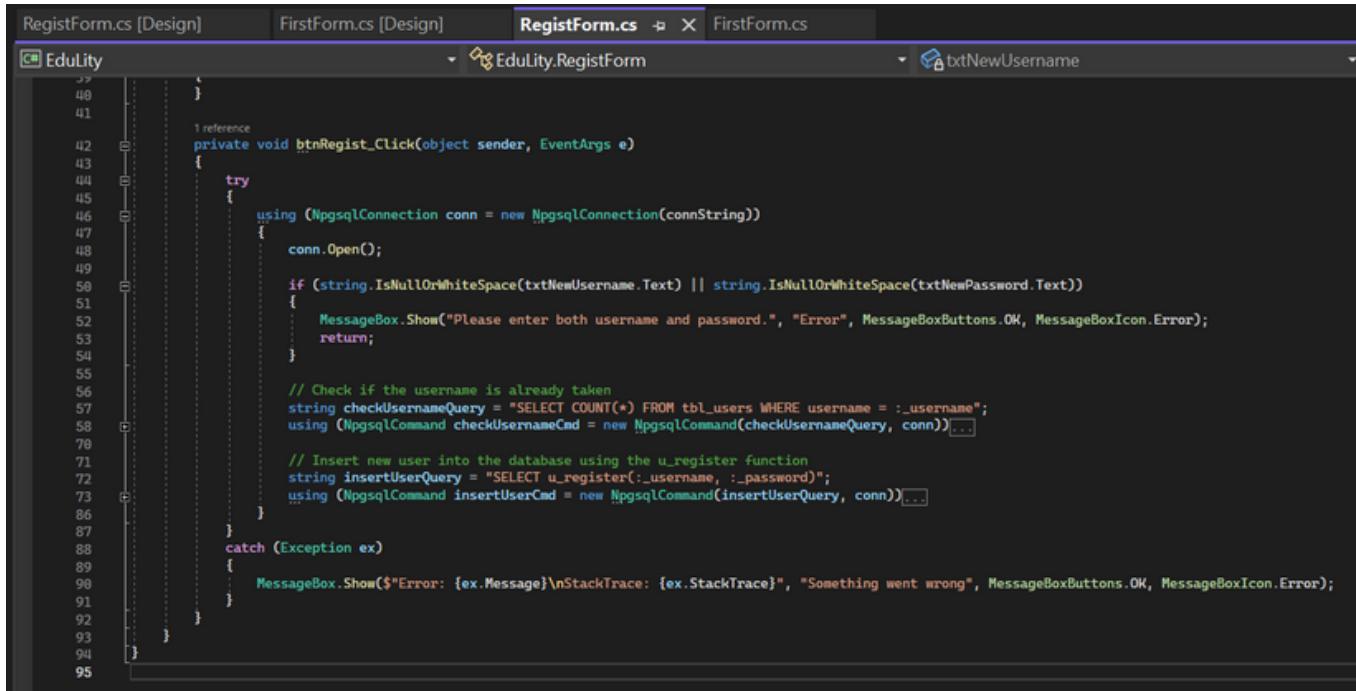
**Validasi Username:** Mengekstrak logika validasi username ke dalam query terpisah untuk meningkatkan keterbacaan.



The screenshot shows the Visual Studio IDE with the 'RegistForm.cs' tab selected. A context menu is open over the validation logic, with the 'Refactor' option highlighted. The code editor displays the following C# code with a separate SQL query:

```
54
55
56     // Check if the username is already taken
57     string checkUsernameQuery = "SELECT COUNT(*) FROM tbl_users WHERE username = @_username";
58     using (NpgsqlCommand checkUsernameCmd = new NpgsqlCommand(checkUsernameQuery, conn))
59     {
60         checkUsernameCmd.Parameters.AddWithValue("_username", txtNewUsername.Text);
61
62         int existingUserCount = Convert.ToInt32(checkUsernameCmd.ExecuteScalar());
63
64         if (existingUserCount > 0)
65         {
66             MessageBox.Show("Username already taken. Please choose a different username.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
67             return;
68         }
69     }
70 }
```

**Logika Registrasi Pengguna:** Modularisasi logika registrasi pengguna ke dalam metode terpisah (**btnRegist\_Click**) untuk organisasi kode yang lebih baik.



The screenshot shows the Visual Studio IDE with the code editor open. The tab bar at the top has three tabs: "RegistForm.cs [Design]", "FirstForm.cs [Design]", "RegistForm.cs", and "FirstForm.cs". The "RegistForm.cs" tab is currently selected. The code in the editor is for the **btnRegist\_Click** event handler. It starts with a try block, checks if both username and password are provided, then checks if the username already exists in the database using a stored procedure **u\_register**. If successful, it inserts the new user into the database. A catch block handles any exceptions and displays an error message box.

```
    39 }
    40 }
    41     1 reference
    42     private void btnRegist_Click(object sender, EventArgs e)
    43     {
    44         try
    45         {
    46             using (NpgsqlConnection conn = new NpgsqlConnection(connString))
    47             {
    48                 conn.Open();
    49
    50                 if (string.IsNullOrWhiteSpace(txtNewUsername.Text) || string.IsNullOrWhiteSpace(txtNewPassword.Text))
    51                 {
    52                     MessageBox.Show("Please enter both username and password.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    53                     return;
    54                 }
    55
    56                 // Check if the username is already taken
    57                 string checkUsernameQuery = "SELECT COUNT(*) FROM tbl_users WHERE username = :_username";
    58                 using (NpgsqlCommand checkUsernameCmd = new NpgsqlCommand(checkUsernameQuery, conn))...
    59
    60                 // Insert new user into the database using the u_register function
    61                 string insertUserQuery = "SELECT u_register(:_username, :_password)";
    62                 using (NpgsqlCommand insertUserCmd = new NpgsqlCommand(insertUserQuery, conn))...
    63             }
    64         }
    65         catch (Exception ex)
    66         {
    67             MessageBox.Show($"Error: {ex.Message}\nStackTrace: {ex.StackTrace}", "Something went wrong", MessageBoxButtons.OK, MessageBoxIcon.Error);
    68         }
    69     }
    70 }
```

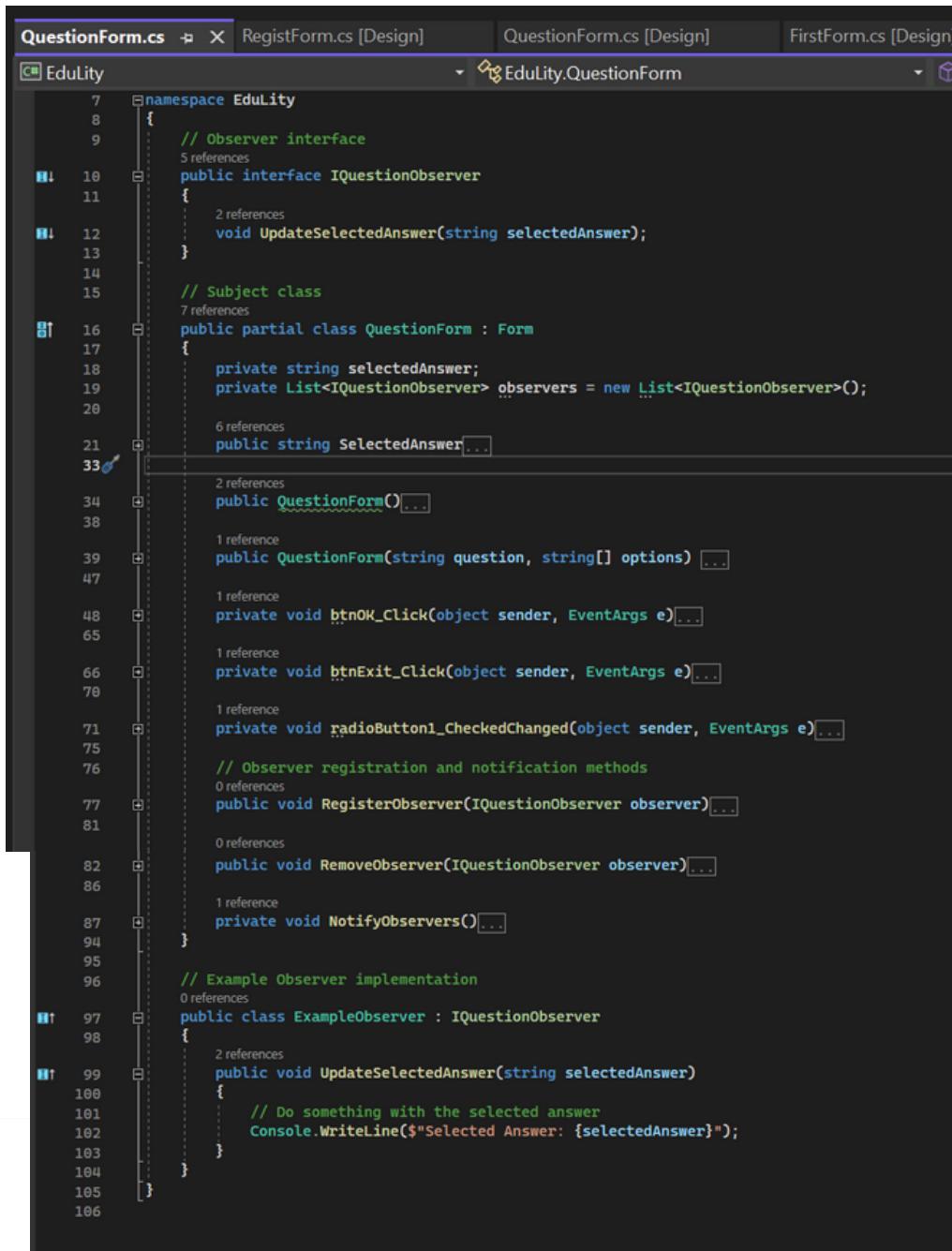
## Kesimpulan Implementasi Refactoring dalam EduLity

Refactoring pada FirstForm dan RegistForm di EduLity membawa sejumlah perbaikan signifikan. Refactoring pada FirstForm meningkatkan penanganan kesalahan koneksi database, sementara di RegistForm membawa perbaikan dalam manajemen sumber daya dengan penggunaan blok "using". Keduanya menghasilkan kode yang lebih bersih, mudah dibaca, dan mematuhi prinsip-prinsip desain.

# 3 Design Patterns Implementations

## Design Pattern pada QuestionForm: Observer

Pola Observer: Diterapkan untuk meningkatkan fleksibilitas dan keterbacaan dengan memisahkan subjek (QuestionForm) dan observer (ExampleObserver).



The screenshot shows a code editor with the file `QuestionForm.cs` open. The code implements the Observer design pattern. It defines an `IQuestionObserver` interface with a method `UpdateSelectedAnswer`. A `QuestionForm` class implements this interface, maintaining a list of observers and notifying them when the selected answer changes. An `ExampleObserver` class provides an example implementation of the observer interface.

```
namespace EduLity
{
    // Observer interface
    public interface IQuestionObserver
    {
        void UpdateSelectedAnswer(string selectedAnswer);
    }

    // Subject class
    public partial class QuestionForm : Form
    {
        private string selectedAnswer;
        private List<IQuestionObserver> observers = new List<IQuestionObserver>();

        public string SelectedAnswer
        {
            get { return selectedAnswer; }
            set
            {
                selectedAnswer = value;
                NotifyObservers();
            }
        }

        public QuestionForm()
        {
            InitializeComponent();
        }

        public QuestionForm(string question, string[] options)
        {
            InitializeComponent();
            LoadQuestion(question);
            LoadOptions(options);
        }

        private void btnOK_Click(object sender, EventArgs e)
        {
            if (selectedAnswer != null)
            {
                DialogResult = DialogResult.OK;
                Close();
            }
        }

        private void btnExit_Click(object sender, EventArgs e)
        {
            DialogResult = DialogResult.Cancel;
            Close();
        }

        private void radioButton1_CheckedChanged(object sender, EventArgs e)
        {
            selectedAnswer = ((RadioButton)sender).Text;
        }

        // Observer registration and notification methods
        public void RegisterObserver(IQuestionObserver observer)
        {
            observers.Add(observer);
        }

        public void RemoveObserver(IQuestionObserver observer)
        {
            observers.Remove(observer);
        }

        private void NotifyObservers()
        {
            foreach (IQuestionObserver observer in observers)
            {
                observer.UpdateSelectedAnswer(selectedAnswer);
            }
        }
    }

    // Example Observer implementation
    public class ExampleObserver : IQuestionObserver
    {
        public void UpdateSelectedAnswer(string selectedAnswer)
        {
            // Do something with the selected answer
            Console.WriteLine($"Selected Answer: {selectedAnswer}");
        }
    }
}
```

## **Design Pattern pada QuestionForm: Observer**

Subjek dan Pengamat: Subjek (QuestionForm) memberikan notifikasi kepada pengamat (ExampleObserver) setiap kali ada perubahan pada jawaban yang dipilih.

Registrasi dan Pemberitahuan: Metode RegisterObserver, RemoveObserver, dan NotifyObservers memungkinkan dinamika pengelolaan pengamat.

Pengamatan Contoh: Melalui implementasi ExampleObserver, kita dapat menyesuaikan tindakan yang diambil setelah pembaruan jawaban terpilih.

## **Kesimpulan Implementasi Design Pattern dalam QuestionForm EduLity**

Penerapan pola observer ini memberikan kemampuan untuk memperluas fungsionalitas dalam menanggapi perubahan pada subjek tanpa mengubah struktur subjek itu sendiri. Misalnya, kita dapat dengan mudah menambahkan pengamat tambahan dengan menerapkan antarmuka IQuestionObserver dan mendaftarkannya pada QuestionForm.



# UI/UX DESIGN

Design & Elements that we use in EduLity

## Forms Design

You don't have an account yet? [Register here!](#)

Regist

Hi there! Welcome to

**EduLity**

Elevate Education, Embrace Quality

Welcome, Barbara Nea

Please fill your data below

Name

Age

Institution

Submit

Welcome to EduLity

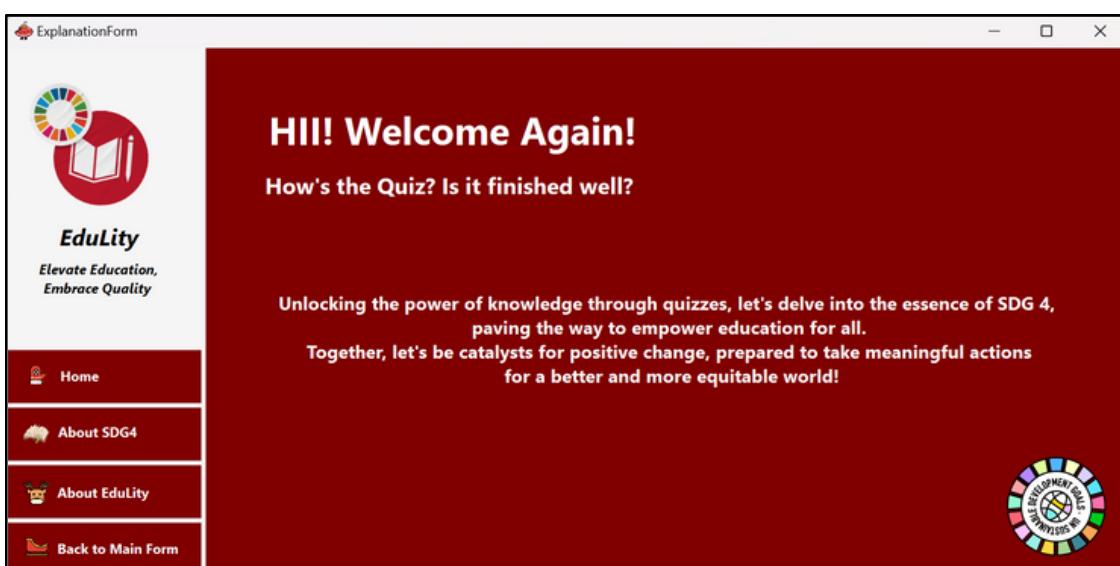
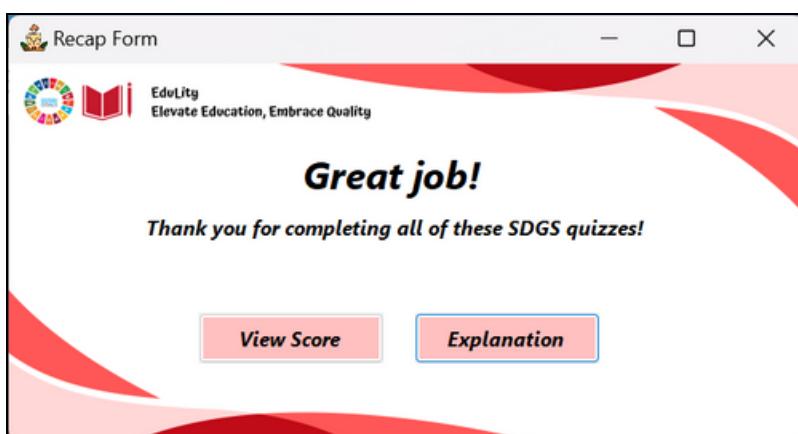
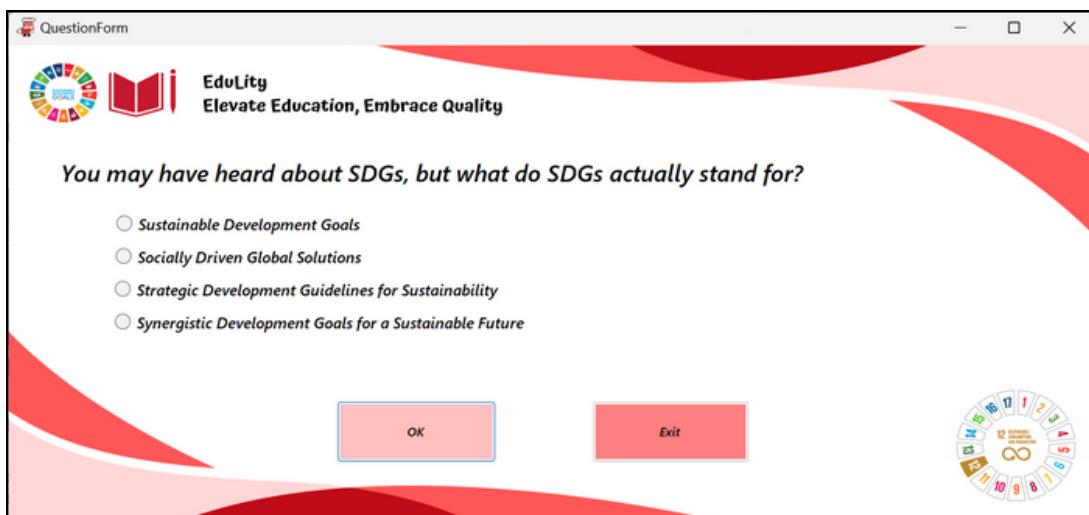
**Ready to start the SDGs quiz?**

Start

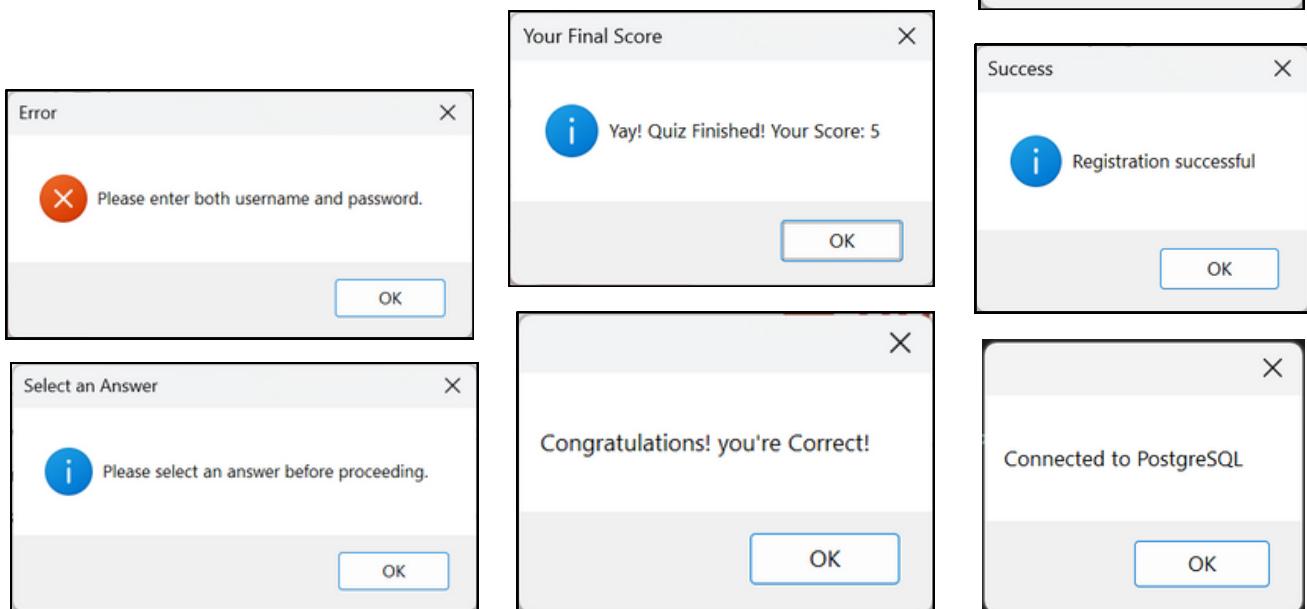
Leave

SUSTAINABLE DEVELOPMENT GOALS

# Forms Design



# Pop Up Notification



## Elements in EduLity

I added Christmas themed elements for decoration because EduLity was created during the Christmas season!

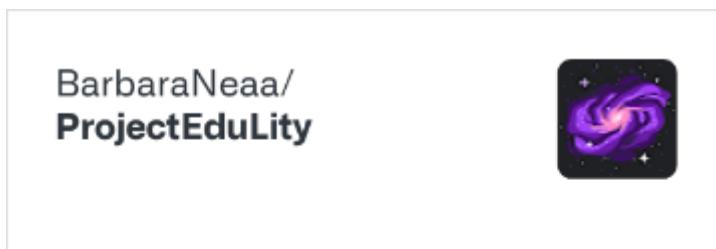




# Our Repository in **GITHUB**

<https://github.com/BarbaraNeaa/ProjectEduLity.git>

BarbaraNeaa/  
**ProjectEduLity**



A 1 Contributor    0 Issues    0 Stars    0 Forks

**BarbaraNeaa/ProjectEduLity**  
Contribute to BarbaraNeaa/ProjectEduLity development by creating an account on GitHub.

 GitHub





# Our Presentation on **YouTube**

<https://youtu.be/T75qnIH7v1I>

The screenshot shows a YouTube video thumbnail for 'EduLity'. The thumbnail features a large red and white abstract graphic at the top. Below it, the word 'EduLity' is written in a large, bold, black font. Underneath, the tagline 'Elevate Edu , Embrace Quality' is displayed in a smaller black font. At the bottom of the thumbnail, the text 'by Asteria Group' is visible. In the top left corner, there is a circular profile picture of a person and a small icon of an open book. To the right of the thumbnail, the video title 'Presentasi Project Akhir PBO - Edulity App by A...' is partially visible. On the far right, there is a 'Copy link' button with a clipboard icon.



# Thank You!

**EDULITY**  
**ELEVATE EDUCATION, EMBRACE QUALITY**

**PROJECT AKHIR PEMROGRAMAN BERORIENTASI OBJEK  
KELOMPOK ASTERIA**

# Member Information

## ASTERIA GROUP

### BARBARA Neanake Ajesti

[barbaraneanakeajesti@mail.ugm.ac.id](mailto:barbaraneanakeajesti@mail.ugm.ac.id)  
[instagram.com/neanakee](https://instagram.com/neanakee)  
[wa.me/6285875891040](https://wa.me/6285875891040)



### Christella JESSLYN Dewantara

[christellajesslyndewantara@mail.ugm.ac.id](mailto:christellajesslyndewantara@mail.ugm.ac.id)  
[instagram.com/jzz.jesslyn](https://instagram.com/jzz.jesslyn)  
[wa.me/6285803747376](https://wa.me/6285803747376)

