Івано-Франківський національний технічний університет нафти і газу

Кафедра інженерії програмного забезпечення

Лабораторна робота №3 Породжувальні шаблони

Виконав

Ст. гр. ІП-22-1

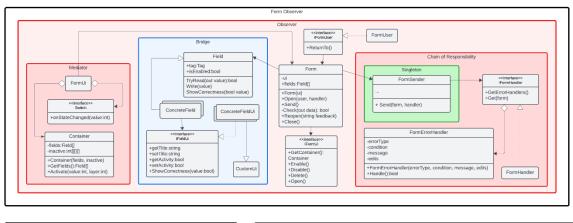
Хімій Денис

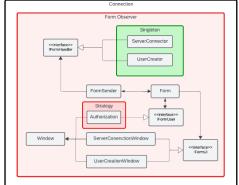
Перевірила

Піх М.М.

Івано-Франківськ 2024 **Мета**: продемонструвати реалізацію породжувальних шаблонів проєктування в коді проєкту

Хід виконання роботи





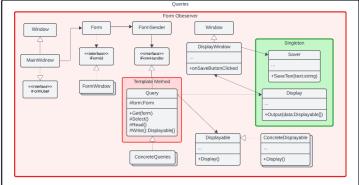


Рис. 1 – Загальна діаграма класів проєкту

Як випливає з рисунку 1 в роботі використано лише один породжувальний шаблон проєктування (позначено зеленим) — Синглтон. Нижче наведено всі 5 класів, які його реалізують.

ServerConnector

```
using CS.General;
using CS.General.Form.Logic;
using Npgsql;
using System.Net.Sockets;
using Input = CS.General.Form.Field.Logic;
namespace CS.Authorization
      public class ServerConnector : IFormHandler
            public FormErrorHandler[] ErrorHandlers => errorHandlers;
            private static ServerConnector singletone;
            private static FormErrorHandler[] errorHandlers;
            public static ServerConnector Instant()
                  return singletone;
            private ServerConnector() { }
            static ServerConnector()
                  _errorHandlers = [
                        new FormErrorHandler(typeof(SocketException), ex => $"Не вдалося
під'єднатися до сервера", (Input.Tag.Host, "...")),
                        new FormErrorHandler(typeof(PostgresException),
((PostgresException)ex).SqlState == "28P01", ex => $"Неправильне ім'я користувача або пароль",
(Input.Tag.Password, string.Empty)),
                                FormErrorHandler(typeof(PostgresException),
                                                                                    =>
                        new
                               == "3D000", ex => $"Базу
((PostgresException)ex).SqlState
                                                                            знайдено",
                                                                даних не
(Input.Tag.Database, string.Empty)),
                        new FormErrorHandler(typeof(PostgresException),
((PostgresException)ex).SqlState == "28000", ex => $"Помилка авторизації"),
                        new FormErrorHandler(typeof(PostgresException), ex => $"Помилка
при під'єднанні до сервера:\n {((PostgresException)ex).ErrorCode}"),
                        new FormErrorHandler(typeof(Exception), ex => true, ex =>
$"Помилка:\n {ex.Message}")];
            public void Get(Form form)
            {
                  Database.Load(new NpgsqlConnection(
                        $"Host={form[Input.Tag.Host]};" +
                         $"Username={form[Input.Tag.Name]};" +
                         $"Password={form[Input.Tag.Password]};" +
                         $"Database={form[Input.Tag.Database]}"));
                  Database.Status = ConnectionStatus.Connected;
            }
      }
```

UserCreator

```
using CS.General;
using CS.General.Form.Logic;
using Npgsql;
using Input = CS.General.Form.Field.Logic;
namespace CS.Authorization
{
      public class UserCreator : IFormHandler
             public FormErrorHandler[] ErrorHandlers => errorHandlers;
             private static UserCreator singleton;
             private static FormErrorHandler[] _errorHandlers;
             private UserCreator() { }
             static UserCreator()
             {
                   errorHandlers = [
                                 FormErrorHandler(typeof(PostgresException),
                         new
                                                                                ex
((PostgresException)ex).SqlState == "42710", "Користувач з таким ім'ям вже існує",
(Input.Tag.Name, string.Empty)),
                         new FormErrorHandler(typeof(Exception), ex => true, ex =>
$"Помилка:\n {ex.Message}")];
             }
             public static UserCreator Instant()
             {
                   if (_singleton == null) _singleton = new UserCreator();
                   return singleton;
             }
             public void Get(Form form)
             {
                   string createUserQuery = $"create role {form[Input.Tag.Name]} superuser
login password '{form[Input.Tag.Password]}'";
                   string grantPrivilegesQuery = $"grant all privileges on database
{Database.Connection.Database} to {form[Input.Tag.Name]};";
                   new
                                                             NpgsqlCommand(createUserQuery,
Database.Connection).ExecuteNonQuery();
                                                        NpgsqlCommand(grantPrivilegesQuery,
                   new
Database.Connection).ExecuteNonQuery();
      }
}
```

FormSender

Display

```
using CS.Output.Items;
using System.Windows;
namespace CS.Output
    public static class Display
      {
             private static DisplayUi? _ui;
             public static void Output(List<Displayable> data)
                    if (_ui == null)
                    {
                          _ui = new DisplayUi();
                          _ui.Closed += (s, e) => _ui = null;
                          _ui.Show();
                    }
                    else
                    {
                          if (_ui.WindowState == WindowState.Minimized)
                                 _ui.WindowState = WindowState.Normal;
                    }
                    _ui.GetData([.. data.Select(e => e.Display())]);
             }
     }
```

Saver

```
using Microsoft.Win32;
using System.IO;
using System.Windows;
namespace CS.Output
      internal static class Saver
             public static void SaveText(string text)
                    var dialogue = new SaveFileDialog
                           Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*",
                           FilterIndex = 1,
                          RestoreDirectory = true
                    } ;
                    if (dialogue.ShowDialog() == true)
                           try
                           {
                                 File.WriteAllText(dialogue.FileName, text);
                                 MessageBox.Show("Дані збережено");
                           catch (Exception ex)
                                 MessageBox.Show($"Помилка при збереженні ({ex.Message})");
                    }
             }
      }
}
```

Висновок

На цій лабораторній роботі продемонструвати реалізацію погоджувального шаблону проєктування Синглтон в коді проєкту.