Buổi 4
Mô hình 3 lớp

I. Mô hình 3 lớp

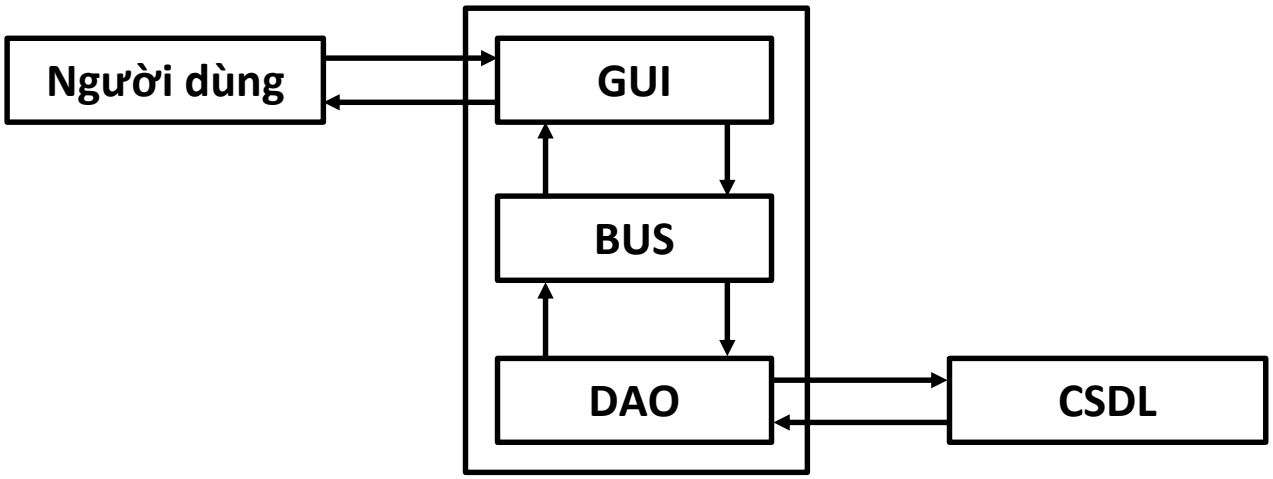
Nhược điểm của mô hình hiện tại (tạm gọi là mô hình 1 lớp):

- Mỗi khi cần thao tác với CSDL, phải thực hiện lại các thao tác tạo kết nối, mở kết nối, xử lí, sau đó đóng kết nối.
- Mã nguồn sẽ bị trùng lặp, dài dòng.
- Chuỗi kết nối phải viết đi viết lại nhiều lần. Nếu thay đổi chuỗi kết nối thì phải sửa rất nhiều chỗ.

Do đó, mô hình 3 lớp ra đời để đơn giản hóa việc kết nối và thực thi truy vấn với CSDL.

1. Khái niệm

Mô hình 3 lớp (*3-layer model*) chia ứng dụng thành 3 phần, mỗi phần đảm nhiệm 1 chức năng khác nhau:



- GUI – *Graphical User Interface*: Chứa giao diện người dùng.
- BUS – *Business Logic Layer*[1]: Đảm nhiệm việc xử lí các thao tác nghiệp vụ sau khi nhận dữ liệu từ GUI gửi xuống cũng như xử lí, kiểm tra dữ liệu từ DAO gửi lên.
- DAO – *Data Access Layer*[2]: Đảm nhiệm việc giao tiếp với CSDL, thực thi truy vấn. Trong lớp DAO có 1 lớp đối tượng có tên là DataProvider cung cấp các phương thức để kết nối và thực thi truy vấn.

Quy trình thực hiện 1 tác vụ như sau:

- Bước 1: Người dùng nhập dữ liệu vào giao diện
- Bước 2: Lớp GUI kiểm tra dữ liệu ở mức giao diện, sau đó gửi dữ liệu xuống lớp BUS.
- Bước 3: Lớp BUS kiểm tra, tính toán và xử lí dữ liệu ở mức nghiệp vụ, sau đó gửi dữ liệu xuống lớp DAO.
- Bước 4: Lớp DAO thực hiện truy vấn vào CSDL, lấy ra kết quả và gửi trả lên cho lớp BUS.

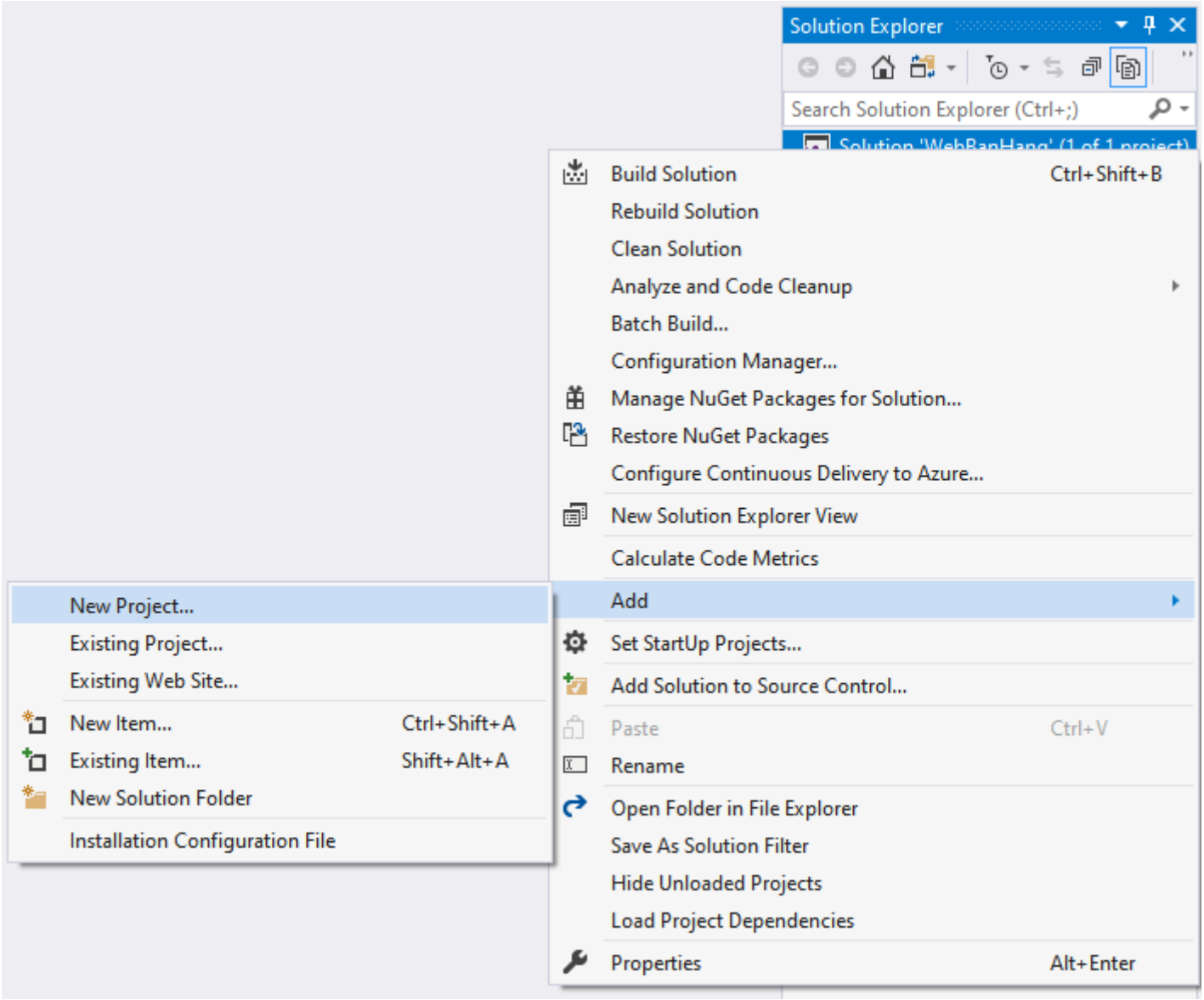
[1] Còn gọi là BLL
[2] Còn gọi là DAL

- Bước 5: Lớp BUS nhận kết quả, kiểm tra, xử lí rồi gửi trả lên cho lớp GUI.
- Bước 6: Lớp GUI hiển thị kết quả cho người dùng.

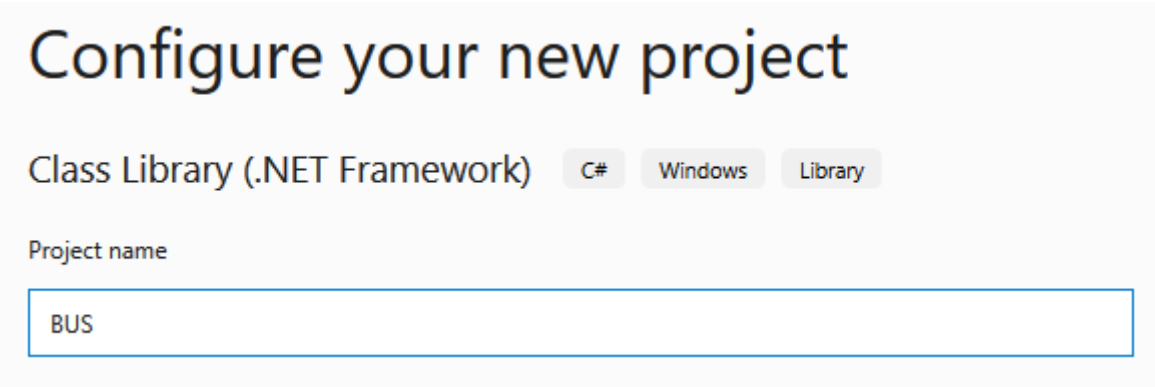
2. Cách tạo mô hình 3 lớp bằng Visual Studio

Để tạo mô hình 3 lớp bằng VS, đặt tên project đầu tiên khi tạo solution là GUI.

Sau khi đã tạo xong solution, clich chuột phải vào solution, chọn *Add* → *New Project*…:



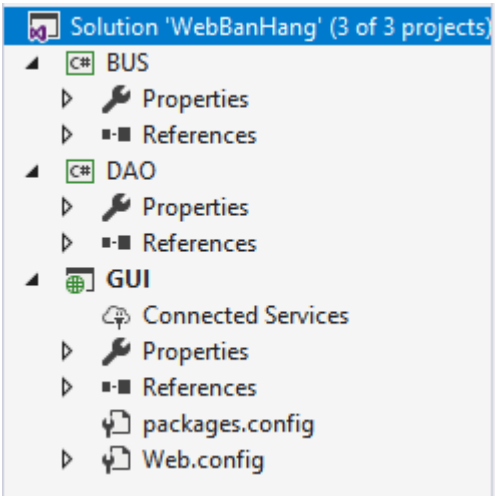
Đặt tên cho project mới là BUS, kiểu project là *Class Library*:



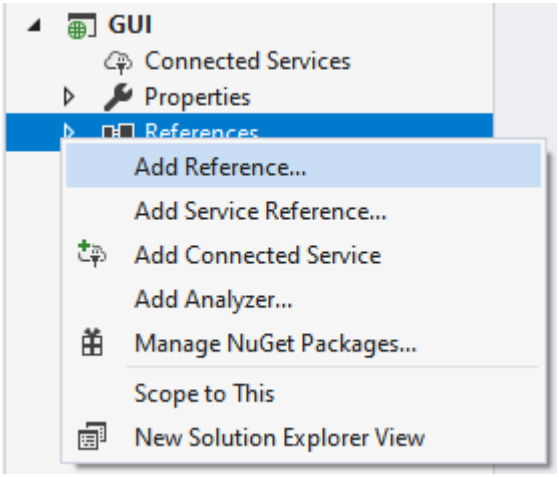
Thực hiện thao tác tương tựng để tạo thêm 1 project có tên là DAO, kiểu Class Library.

Sau khi tạo xong 2 project BUS và DAO, có thể xóa file *Class1.cs* được tạo sẵn trong đó.

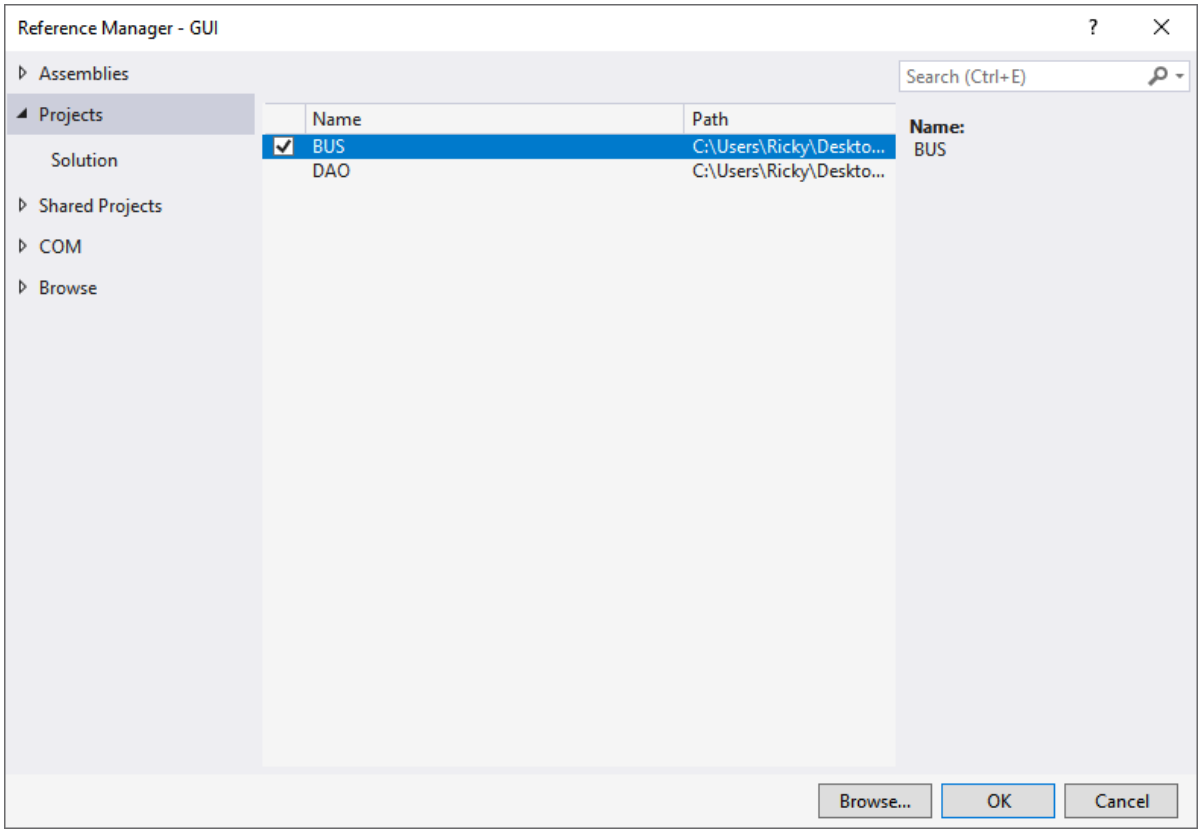
Cấu trúc solution sau khi tạo xong sẽ như sau:



Để các lớp có thể tương tác được với nhau, cần phải thêm *References* vào lớp GUI và BUS bằng cách click chuột phải vào mục *References* và chọn *Add Reference*...:



Trong cửa sổ hiện ra, chọn *Projects* ở cột bên trái, sau đó check vào những mục muốn thêm vào *References*, sau đó nhấn OK. Với lớp GUI, cần thêm *References* là BUS. Với lớp BUS, cần thêm *References* là DAO:



Cuối cùng, cần thêm class `DataProvider` vào project DAO bằng cách copy file *DataProvider.cs* được cung cấp sẵn vào thư mục chứa project.

II. Class DataProvider

Class `DataProvider` thuộc lớp DAO cung cấp các phương thức giúp mở kết nối, thực thi truy vấn và trả về kết quả, sau đó đóng kết nối.

Các thuộc tính:

```
private static SqlDataAdapter adapter = new SqlDataAdapter();
private static SqlConnection conn = new SqlConnection(<chuỗi kết nối>);
```

Phương thức `OpenConnection()` giúp tạo và mở kết nối tới CSDL:

```
private static SqlConnection OpenConnection()
{
    if (conn.State == ConnectionState.Closed || conn.State ==
   ConnectionState.Broken)
    {
        conn.Open();
    }
    return conn;
}
```

Phương thức `ExecuteSelectQuery()` giúp thực thi câu lệnh `SELECT`:

```
public static DataTable ExecuteSelectQuery(string query, SqlParameter[]
param)
{
    SqlCommand cmd = new SqlCommand();
    DataTable dtbKetQua= new DataTable();
    try
    {
        cmd.Connection = OpenConnection();
        cmd.CommandText = query;
        cmd.Parameters.AddRange(param);
        adapter.SelectCommand = cmd;
        adapter.Fill(dtbKetQua);
        conn.Close();
    }
    catch (SqlException e)
    {
        return null;
    }
    return dtbKetQua;
}
```

Phương thức `ExecuteInsertQuery()` giúp thực thi câu lệnh `INSERT`:

```
public static int ExecuteInsertQuery(string query, SqlParameter[] param)
{
    SqlCommand cmd = new SqlCommand();
    int rowsAffected;
    try
    {
        cmd.Connection = OpenConnection();
        cmd.CommandText = query;
```

```
                cmd.Parameters.AddRange(param);
                adapter.InsertCommand = cmd;
                rowsAffected = cmd.ExecuteNonQuery();
                conn.Close();
            }
            catch (SqlException e)
            {
                return 0;
            }
            return rowsAffected;
        }
```

Phương thức `ExecuteUpdateQuery()` và `ExecuteDeleteQuery()` để thực thi câu lệnh `UPDATE` và `DELETE` được viết tương tự phương thức `ExecuteInsertQuery()`.

Lưu ý: Các thuộc tính và phương thức của lớp đối tượng `DataProvider` đều được khai báo `static` để có thể được gọi thông qua tên lớp mà không cần tạo đối tượng.

III. Cài đặt tính năng đăng nhập bằng mô hình 3 lớp

Để minh họa cho mô hình 3 lớp, chúng ta sẽ cài đặt tính năng đăng nhập cho trang web.

Giao diện trang *DangNhap.aspx* như sau:

Thông thường, mỗi bảng trong CSDL sẽ có 1 class thuộc BUS và 1 class thuộc DAO để đảm nhiệm các thao tác liên quan đến bảng đó. Các class này phải có tầm vực `public`.

Ví dụ:

- CSDL có bảng TaiKhoan thì project sẽ có 2 class `TaiKhoanBUS` và `TaiKhoanDAO`.
- CSDL có bảng SanPham thì project sẽ có 2 class `SanPhamBUS` và `SanPhamDAO`.
- …

1. DAO

Cách viết 1 phương thức tương tác với CSDL ở lớp DAO (cụ thể là ở class `TaiKhoanDAO`):

- Viết câu truy vấn, lưu vào 1 biến chuỗi.
- Liệt kê danh sách tham số của câu truy vấn, lưu vào 1 mảng kiểu SqlParameter.
- Gọi phương thức thực thi truy vấn tương ứng của `DataProvider`. Xử lí kết quả truy vấn trước khi trả kết quả (nếu cần).

Cài đặt phương thức `KTTKTonTai()` để kiểm tra 1 tài khoản nào đó có tồn tại hay không:

```
    public static bool KTTKTonTai(string tenTK)
    {
        string query = "SELECT COUNT(*) FROM TaiKhoan WHERE TenTaiKhoan =
        @TenTaiKhoan";
        SqlParameter[] param = new SqlParameter[1];
        param[0] = new SqlParameter("@TenTaiKhoan", tenTK);
        return Convert.ToInt32(DataProvider.ExecuteSelectQuery(query,
        param).Rows[0][0]) == 1;
    }
```

Cài đặt phương thức `LayMatKhau()` để lấy mật khẩu của 1 tài khoản nào đó từ CSDL:

```
public static string LayMatKhau(string tenTK)
{
    string query = "SELECT MatKhau FROM TaiKhoan WHERE TenTaiKhoan =
    @TenTaiKhoan";
    SqlParameter[] param = new SqlParameter[1];
    param[0] = new SqlParameter("@TenTaiKhoan", tenTK);
    return DataProvider.ExecuteSelectQuery(query,
    param).Rows[0][0].ToString();
}
```

2. BUS

Trong mỗi class thuộc lớp BUS, cần phải thêm namespace DAO bằng câu lệnh:

```
using DAO;
```

Cài đặt phương thức `KTDangNhap()` để kiểm tra tài khoản và mật khẩu có đúng hay không:

```
public static bool KTDangNhap(string tenTK, string mk)
{
    if (!TaiKhoanDAO.KTTKTonTai(tenTK))
    {
        return false;
    }
    else
    {
        return mk == TaiKhoanDAO.LayMatKhau(tenTK);
    }
}
```

3. GUI

Trong mã nguồn mỗi trang web thuộc lớp GUI, cần phải thêm namespace BUS bằng câu lệnh:

```
using BUS;
```

Cài đặt sự kiện `Click` cho nút Đăng nhập:

```
protected void btnDangNhap_Click(object sender, EventArgs e)
{
    string tenTK = txtTenTK.Text;
    string mk = txtMatKhau.Text;

    if (TaiKhoanBUS.KTDangNhap(tenTK, mk))
    {
        // Đăng nhập thành công
    }
    else
    {
        // Đăng nhập thất bại
    }
}
```

IV. Bài tập

Cài đặt trang Đăng kí tài khoản bằng mô hình 3 lớp.