

## Buổi 5

# GridView

Trong khuôn khổ tài liệu buổi 5, chúng ta sử dụng CSDL WebBanHang:

- File [WebBanHang] Mo ta CSDL.png: Lược đồ mô tả cấu trúc bảng, khóa ngoại.
- File [WebBanHang] CSDL.sql: Phát sinh CSDL.

### I. Khái niệm

Trong ASP.NET, GridView là 1 control cho phép hiển thị dữ liệu dưới dạng bảng (*table*). Mỗi cột hiển thị 1 thuộc tính (*property*), mỗi dòng hiển thị 1 bản ghi (*record*). Do đó, GridView là 1 control rất thích hợp để hiển thị dữ liệu từ CSDL, hoặc từ các đối tượng thuộc kiểu DataTable.

Trong khuôn khổ tài liệu buổi 5, chúng ta sẽ minh họa chức năng quản lý loại sản phẩm, gồm có: xem, thêm, sửa, xóa loại sản phẩm.

Cú pháp để tạo GridView trong ASP.NET:

```
<asp:GridView ID="grvLoaiSanPham" runat="server"></asp:GridView>
```

Ở trang *QLLoaiSanPham.aspx*, phía trên GridView chúng ta sẽ tạo 1 form dùng cho việc thêm và sửa loại sản phẩm như sau:

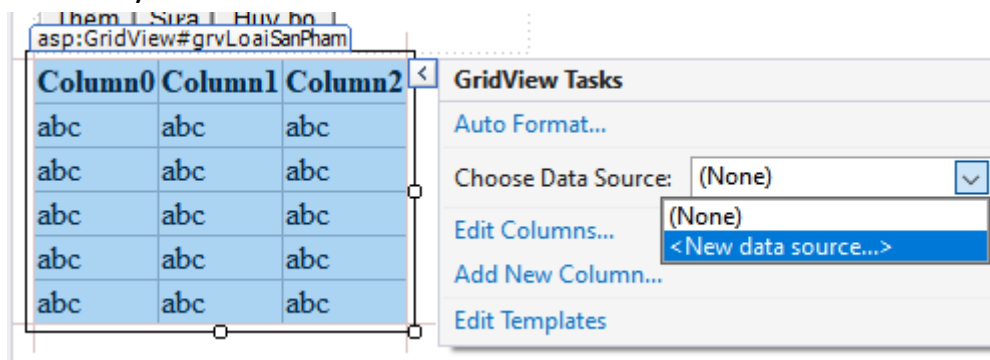
Mã loại sản phẩm:	<input type="text"/>	
Tên loại sản phẩm:	<input type="text"/>	
Trạng thái:	<input checked="" type="checkbox"/>	
<input type="button" value="Thêm"/>	<input type="button" value="Sửa"/>	<input type="button" value="Hủy bỏ"/>

### II. Hiển thị dữ liệu lên GridView

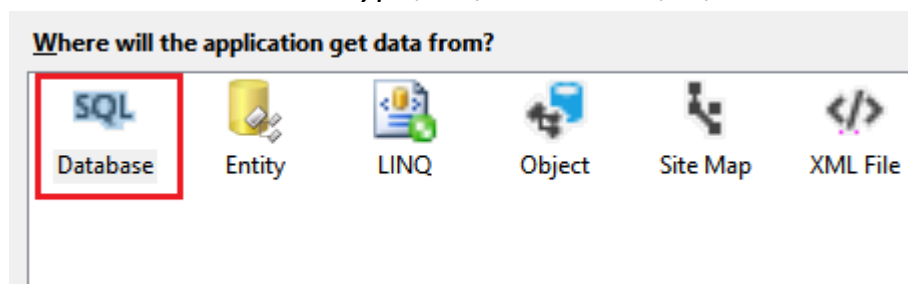
Có 2 cách để đổ (*fill*) dữ liệu vào GridView: sử dụng SqlDataSource hoặc sử dụng code behind.

#### 1. Đổ dữ liệu vào GridView bằng SqlDataSource

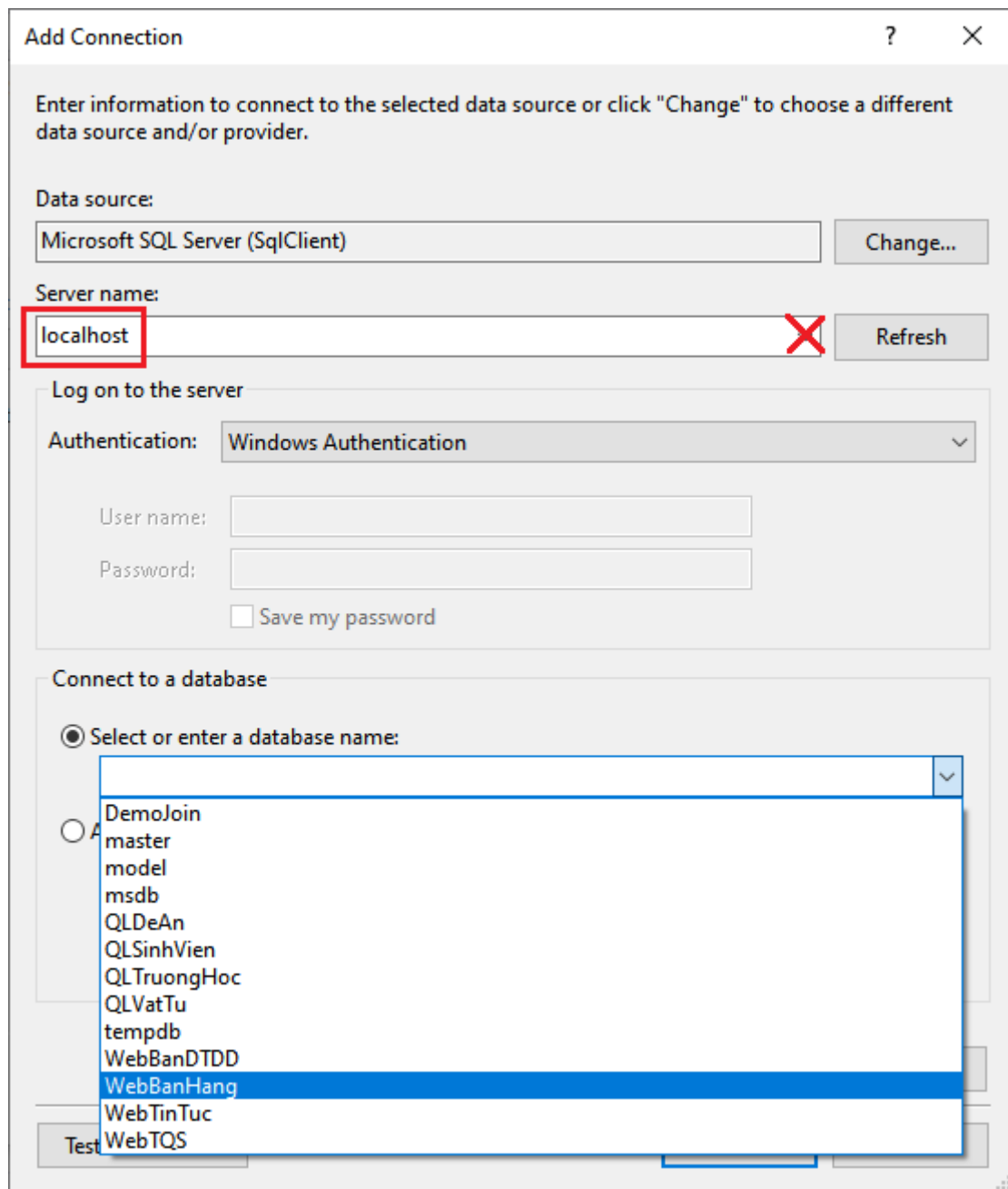
Chuyển sang chế độ hiển thị giao diện (*Design* hoặc *Split*) trong VS. Để tạo SqlDataSource, thực hiện theo hình dưới đây:



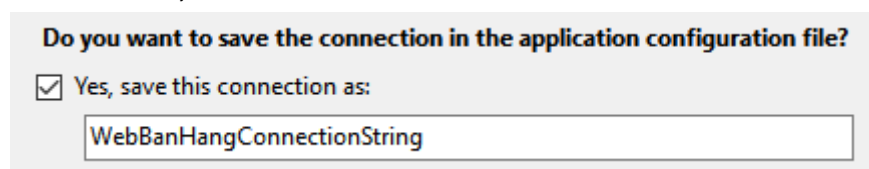
Trong cửa sổ *Choose a Data Source Type*, chọn *Database*, đặt tên cho DataSource và nhấn OK:



Trong cửa sổ *Choose Your Data Connection*, chọn kết nối tới CSDL trong danh sách, hoặc nhấn nút *New Connection...* để tạo kết nối mới. Với trường hợp tạo kết nối mới, trong cửa sổ *Add Connection*, điền *localhost* vào mục *Server name* (Khuyến cáo: Không nên click vào nút mũi tên xổ xuống ở mục này), chọn tên CSDL ở mục *Select or enter a database name*, sau đó nhấn *OK*:




Quay trở lại cửa sổ *Choose Your Data Connection*, nhấn *Next*. Trong cửa sổ tiếp theo, nếu muốn lưu chuỗi kết nối vào file *Web.config* để sử dụng sau này thì check vào ô *Yes, save this connection as* và đặt tên cho chuỗi kết nối, sau đó nhấn *Next*.



Trong cửa sổ *Configure the Select Statement*, có 2 lựa chọn:

- *Specify a custom SQL Statement or stored procedure*: Tự viết truy vấn để lấy dữ liệu.
- *Specify columns from a table or view*: Chọn bảng, chọn cột cần lấy dữ liệu.

Configure Data Source - SqlDataSource1

 **Configure the Select Statement**

How would you like to retrieve data from your database?

☐ Specify a custom SQL statement or stored procedure

☒ Specify columns from a table or view

Name:

LoaiSanPham

Columns:

☒ \*

☐ MaLoaiSP

☐ TenLoaiSP

☐ TrangThai

☐ Return only unique rows

WHERE...

ORDER BY...

Advanced...

SELECT statement:

SELECT \* FROM [LoaiSanPham]

< Previous

Next >

Finish

Cancel

Sau khi đã viết truy vấn hoặc chọn bảng, nhấn *Next*. Trong cửa sổ *Test Query*, có thể nhấn nút *Test Query* để kiểm tra kết quả câu truy vấn, sau đó nhấn *Finish*.

Kết quả GridView hiển thị như sau:

MaLoaiSP	TenLoaiSP	TrangThai
LSP001	Sách giáo khoa	<input checked="" type="checkbox"/>
LSP002	Sách tham khảo	<input checked="" type="checkbox"/>
LSP003	Sách nước ngoài	<input checked="" type="checkbox"/>
LSP004	Báo & Tạp chí	<input checked="" type="checkbox"/>
LSP005	Tiểu thuyết & Tự truyện	<input checked="" type="checkbox"/>
LSP006	Khác	<input checked="" type="checkbox"/>

## 2. Đổ dữ liệu vào GridView bằng code behind

Sau khi đã biết về mô hình 3 lớp, chúng ta sẽ sử dụng mô hình này để lấy dữ liệu từ CSDL, lưu vào 1 DataTable. GridView hỗ trợ thuộc tính DataSource để đổ dữ liệu từ DataTable.

Đầu tiên, cần tạo 2 lớp đối tượng LoaiSanPhamBUS và LoaiSanPhamDAO để phục vụ các thao tác trên bảng LoaiSanPham.

Tại lớp DAO, chúng ta cài đặt phương thức lấy toàn bộ dữ liệu của bảng LoaiSanPham:

```
public static DataTable LayDSLoaiSanPham()
{
    string query = "SELECT * FROM LoaiSanPham";
```

```

SqlParameter[] param = new SqlParameter[0];
return DataProvider.ExecuteSelectQuery(query, param);
}

```

Tại lớp BUS, chúng ta cài đặt phương thức lấy dữ liệu từ DAO và gửi trả cho GUI. Với yêu cầu này, dữ liệu không có sự thay đổi nên không cần thêm bước xử lý:

```

public static DataTable LayDSLoaiSanPham()
{
    return LoaiSanPhamDAO.LayDSLoaiSanPham();
}

```

Tại lớp GUI, chúng ta cần đổ dữ liệu lên GridView ngay khi trang web vừa hiển thị lên. Do đó, chúng ta sẽ viết đoạn code để đổ dữ liệu vào GridView trong sự kiện Page\_Load:

```

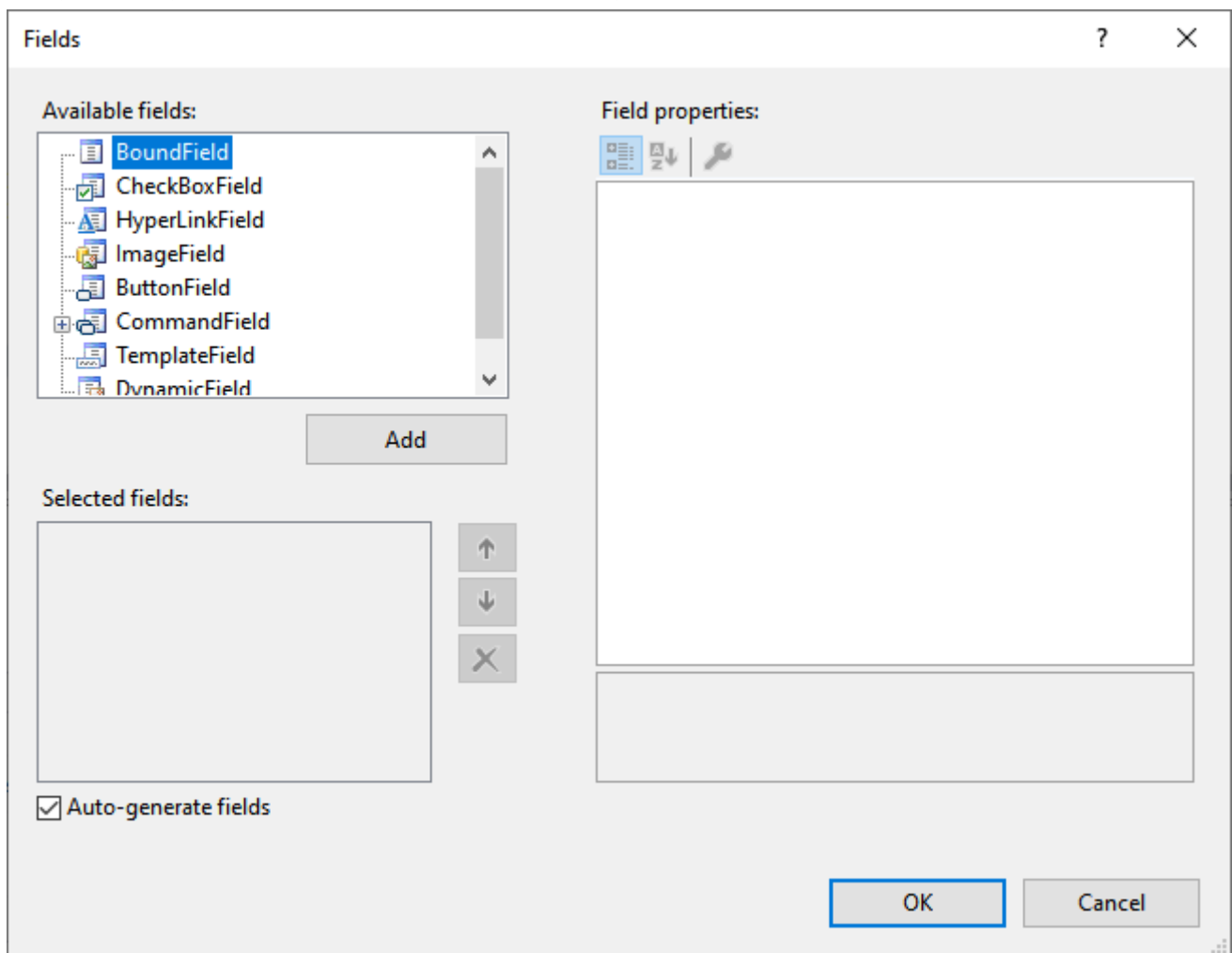
protected void Page_Load(object sender, EventArgs e)
{
    grvLoaiSanPham.DataSource = LoaiSanPhamBUS.LayDSLoaiSanPham();
    grvLoaiSanPham.DataBind();
}

```

### 3. Điều chỉnh cột cho GridView

Chúng ta có thể điều chỉnh cột cho GridView, như đặt lại tiêu đề cột, chỉ hiển thị 1 số cột nhất định, thêm các cột chức năng cho GridView...

Chuyển sang chế độ hiển thị giao diện trong VS, chọn *Edit columns...*



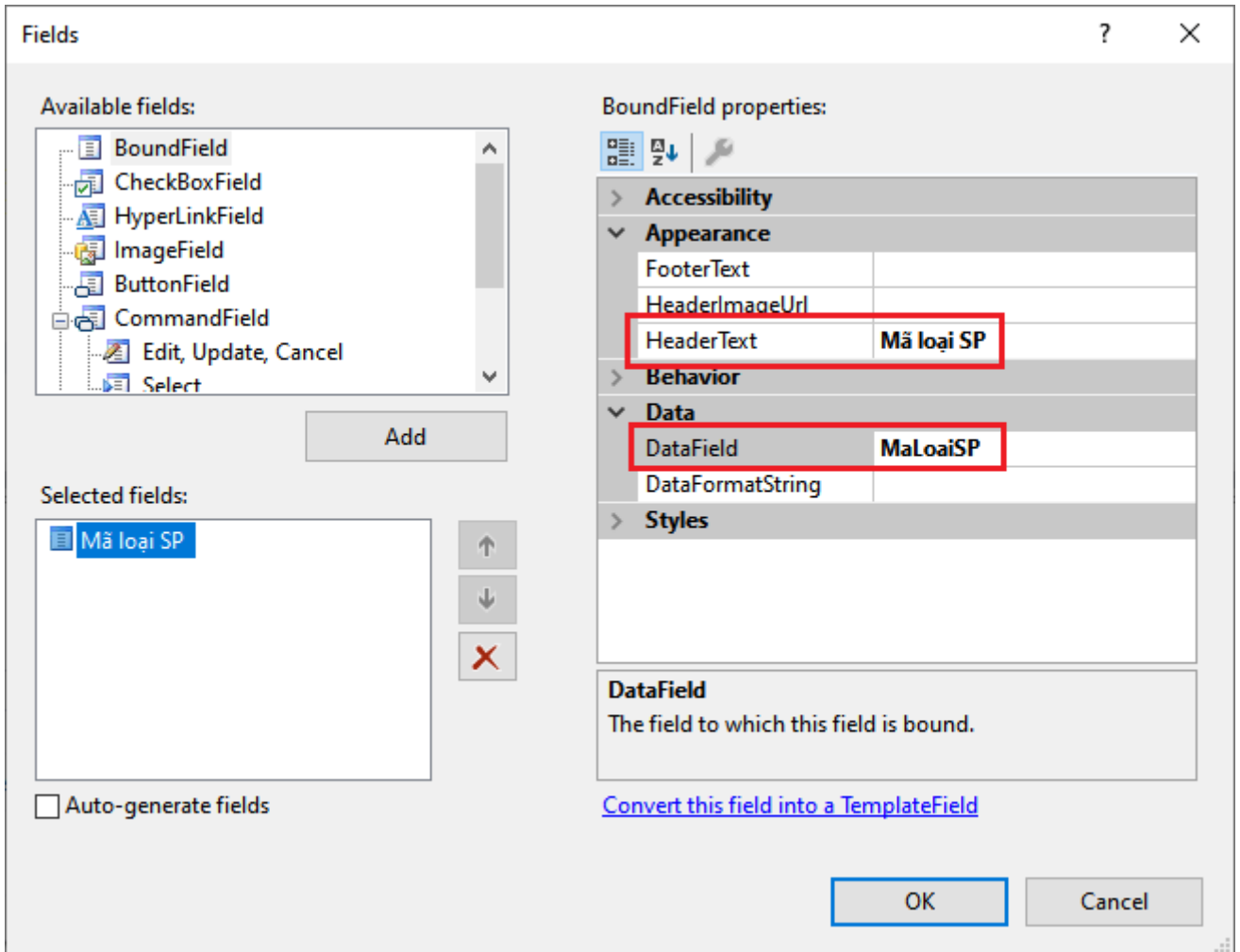
Trong cửa sổ hiện ra, bỏ chọn *Auto-generate fields*<sup>1</sup> để ngăn GridView tự phát sinh cột dữ liệu. GridView hỗ trợ các loại cột sau (xem mục *Available fields*):

<sup>1</sup> Cách khác: Thay đổi giá trị thuộc tính `AutoGenerateColumns` của GridView thành `false`.

- BoundField: Hiển thị dữ liệu dạng text.
- CheckBoxField, HyperLinkField, ImageField, ButtonField: Hiển thị dữ liệu dạng checkbox, hyperlink, hình ảnh, button.
- CommandField: Hiển thị các nút lệnh như Chọn, Sửa, Xóa.

Mỗi loại cột đều có thuộc tính HeaderText để quy định tên hiển thị cho cột, và thuộc tính DataField để quy định cột này sẽ lấy dữ liệu từ trường (*field*) nào trong DataSource.

Tạo 1 cột dạng BoundField để hiển thị Mã loại sản phẩm như sau:



Tương tự, tạo 1 cột dạng BoundField để hiển thị Tên loại sản phẩm và 1 cột dạng CheckBoxField để hiển thị trạng thái. Sau đó, nhấn OK.

Kết quả GridView hiển thị như sau:

Mã loại SP	Tên loại SP	Trạng thái
LSP001	Sách giáo khoa	<input checked="" type="checkbox"/>
LSP002	Sách tham khảo	<input checked="" type="checkbox"/>
LSP003	Sách nước ngoài	<input checked="" type="checkbox"/>
LSP004	Báo & Tạp chí	<input checked="" type="checkbox"/>
LSP005	Tiểu thuyết & Tự truyện	<input checked="" type="checkbox"/>
LSP006	Khác	<input checked="" type="checkbox"/>

### III. Tương tác với dữ liệu thông qua GridView

Trong phần III, chúng ta sẽ bổ sung các chức năng Thêm, Sửa, Xóa cho GridView.

#### 1. Chức năng Thêm

Để thực hiện chức năng Thêm dữ liệu vào bảng, chúng ta sẽ sử dụng form đã tạo sẵn, sau đó cập nhật lại GridView để hiển thị dữ liệu mới.

Tại lớp DAO, chúng ta cài đặt phương thức thêm dữ liệu vào bảng LoaiSanPham:

```
public static bool ThemLoaiSanPham(string maLoaiSP, string tenLoaiSP, bool
trangThai)
{
    string query = "INSERT INTO LoaiSanPham (MaLoaiSP, TenLoaiSP,
TrangThai) VALUES (@MaLoaiSP, @TenLoaiSP, @TrangThai)";
    SqlParameter[] param = new SqlParameter[3];
    param[0] = new SqlParameter("@MaLoaiSP", maLoaiSP);
    param[1] = new SqlParameter("@TenLoaiSP", tenLoaiSP);
    param[2] = new SqlParameter("@TrangThai", trangThai);
    return DataProvider.ExecuteInsertQuery(query, param) == 1;
}
```

Ngoài ra, tại lớp DAO cần phải có phương thức kiểm tra mã loại sản phẩm đã tồn tại chưa trước khi thêm dữ liệu:

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

Tại lớp BUS, chúng ta cài đặt phương thức gọi lớp DAO thực hiện thao tác thêm dữ liệu vào bảng LoaiSanPham, sau đó trả kết quả về cho lớp GUI:

```
public static bool ThemLoaiSanPham(string maLoaiSP, string tenLoaiSP, bool
trangThai)
{
    if (LoaiSanPhamDAO.KTMaLoaiSanPhamTonTai(maLoaiSP))
    {
        return false;
    }
    else
    {
        return LoaiSanPhamDAO.ThemLoaiSanPham(maLoaiSP, tenLoaiSP,
trangThai);
    }
}
```

Tại lớp GUI, chúng ta xử lý sự kiện Click cho nút *Thêm* như sau:

```
protected void btnThem_Click(object sender, EventArgs e)
{
    string maLoaiSP = txtMaLoaiSP.Text;
    string tenLoaiSP = txtTenLoaiSP.Text;
    bool trangThai = chkTrangThai.Checked;
    if (LoaiSanPhamBUS.ThemLoaiSanPham(maLoaiSP, tenLoaiSP, trangThai))
    {
        grvLoaiSanPham.DataSource = LoaiSanPhamBUS.LayDSLoaiSanPham();
        grvLoaiSanPham.DataBind();
    }
}
```

```

        else
        {
            // Thêm thất bại
        }
    }
}

```

## 2. Chức năng Sửa

Chức năng Sửa được thực hiện như sau: Chọn 1 dòng muốn sửa trong GridView. Thông tin của dòng đó sẽ được hiển thị lên form. Sau đó, người dùng nhập thông tin mới (không cho phép sửa thuộc tính đang là khóa chính). Cuối cùng, nhấn nút *Sửa*, dữ liệu sẽ được cập nhật vào CSDL và GridView sẽ được cập nhật lại để hiển thị dữ liệu mới.

Tại lớp DAO, chúng ta cài đặt phương thức sửa dữ liệu vào bảng LoaiSanPham:

```

public static bool SuaLoaiSanPham(string maLoaiSP, string tenLoaiSP, bool
trangThai)
{
    string query = "UPDATE LoaiSanPham SET TenLoaiSP = @TenLoaiSP,
TrangThai = @TrangThai WHERE MaLoaiSP = @MaLoaiSP";
    SqlParameter[] param = new SqlParameter[3];
    param[0] = new SqlParameter("@MaLoaiSP", maLoaiSP);
    param[1] = new SqlParameter("@TenLoaiSP", tenLoaiSP);
    param[2] = new SqlParameter("@TrangThai", trangThai);
    return DataProvider.ExecuteNonQuery(query, param) == 1;
}

```

Ngoài ra, tại lớp DAO, chúng ta cần phải có phương thức lấy thông tin của 1 mã loại sản phẩm cụ thể:

```

public static DataRow LayLoaiSanPham(string maLoaiSP)
{
    string query = "SELECT * FROM LoaiSanPham WHERE MaLoaiSP = @MaLoaiSP";
    SqlParameter[] param = new SqlParameter[1];
    param[0] = new SqlParameter("@MaLoaiSP", maLoaiSP);
    return DataProvider.ExecuteSelectQuery(query, param).Rows[0];
}

```

Tại lớp BUS, chúng ta cài đặt phương thức gọi lớp DAO thực hiện thao tác sửa dữ liệu vào bảng LoaiSanPham, sau đó trả kết quả về cho lớp GUI:

```

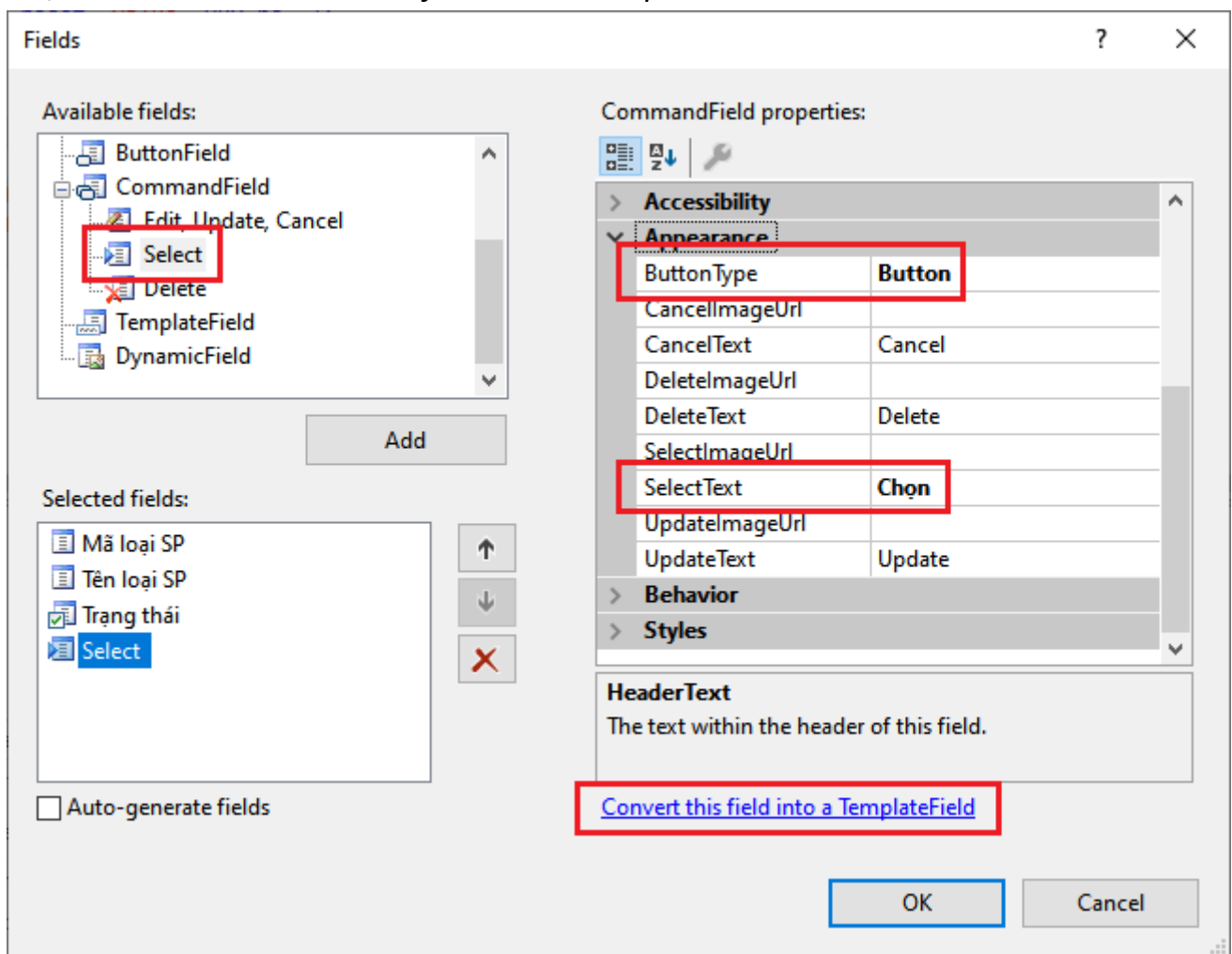
public static bool SuaLoaiSanPham(string maLoaiSP, string tenLoaiSP, bool
trangThai)
{
    if (LoaiSanPhamDAO.KTMaLoaiSanPhamTonTai(maLoaiSP))
    {
        return LoaiSanPhamDAO.SuaLoaiSanPham(maLoaiSP, tenLoaiSP,
trangThai);
    }
    else
    {
        return false;
    }
}

```

Ngoài ra, tại lớp BUS, chúng ta cần phải có phương thức gọi lớp DAO thực hiện thao tác lấy thông tin của 1 mã loại sản phẩm cụ thể, sau đó trả kết quả về cho lớp GUI:

```
public static DataRow LayLoaiSanPham(string maLoaiSP)
{
    if (LoaiSanPhamDAO.KiemTraMaLoaiSanPhamTonTai(maLoaiSP))
    {
        return LoaiSanPhamDAO.LayLoaiSanPham(maLoaiSP);
    }
    else
    {
        return null;
    }
}
```

Tại lớp GUI, để thêm nút Chọn vào mỗi dòng trong GridView, chúng ta thêm cột CommandField như sau, sau đó nhấn *Convert this field into a TemplateField*:



Mã nguồn ASP.NET dành cho nút *Chọn* (sau khi đã chuyển thành TemplateField) như sau:

```
<asp:TemplateField ShowHeader="False">
    <ItemTemplate>
        <asp:Button ID="btnChon" runat="server" CausesValidation="False"
            CommandName="ChonSP" CommandArgument='<%# Eval("MaLoaiSP") %>'
            Text="Chọn" />
    </ItemTemplate>
</asp:TemplateField>
```



Trong đó:

- CommandName do lập trình viên tự đặt để phân biệt với các CommandField và TemplateField khác trong GridView.
- CommandArgument là tham số cho nút Chọn. Giá trị `<%# Eval("MaLoaiSP") %>` quy định nút *Chọn* sẽ lấy tham số là thuộc tính MaLoaiSP của dòng được chọn.

Mặc dù nút *Chọn* là 1 button, nhưng chúng ta không thể cài đặt sự kiện Click như với 1 button thông thường, mà phải xử lý thông qua sự kiện RowCommand của GridView như sau:

```
protected void grvLoaiSanPham_RowCommand(object sender,
GridViewCommandEventArgs e)
{
    if (e.CommandName == "ChonSP")
    {
        string maLoaiSP = e.CommandArgument.ToString();
        DataRow dr = LoaiSanPhamBUS.LayLoaiSanPham(maLoaiSP);
        if (dr != null)
        {
            txtMaLoaiSP.Text = dr["MaLoaiSP"].ToString();
            txtTenLoaiSP.Text = dr["TenLoaiSP"].ToString();
            chkTrangThai.Checked = Convert.ToBoolean(dr["TrangThai"]);

            txtMaLoaiSP.Enabled = false;
        }
    }
}
```

Tuy nhiên, sau khi biên dịch trang web và nhấn nút *Chọn*, chúng ta gặp phải 1 lỗi (exception):

#### Server Error in '/' Application.

**Invalid postback or callback argument.** Event validation is enabled using `<pages enableEventValidation="true"/>` in configuration or `<%@ Page EnableEventValidation="true" %>` in a page. For security purposes, this feature verifies that arguments to postback or callback events originate from the server control that originally rendered them. If the data is valid and expected, use the `ClientScriptManager.RegisterForEventValidation` method in order to register the postback or callback data for validation.

**Description:** An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

**Exception Details:** System.ArgumentException: Invalid postback or callback argument. Event validation is enabled using `<pages enableEventValidation="true"/>` in configuration or `<%@ Page EnableEventValidation="true" %>` in a page. For security purposes, this feature verifies that arguments to postback or callback events originate from the server control that originally rendered them. If the data is valid and expected, use the `ClientScriptManager.RegisterForEventValidation` method in order to register the postback or callback data for validation.

#### Source Error:

An unhandled exception was generated during the execution of the current web request. Information regarding the origin and location of the exception can be identified using the exception stack trace below.

#### Stack Trace:

```
[ArgumentException: Invalid postback or callback argument. Event validation is enabled using <pages enableEventValidation="true"/> in configuration or <%@ Page EnableEventValidation="true" %> in a page. For security purposes, this feature verifies that arguments to postback or callback events originate from the server control that originally rendered them. If the data is valid and expected, use the ClientScriptManager.RegisterForEventValidation method in order to register the postback or callback data for validation.]
   System.Web.UI.ClientScriptManager.ValidateEvent(String uniqueId, String argument) +9832642
   System.Web.UI.WebControls.Button.RaisePostBackEvent(String eventArgument) +131
   System.Web.UI.WebControls.Button.System.Web.UI.IPostBackEventHandler.RaisePostBackEvent(String eventArgument) +12
   System.Web.UI.Page.RaisePostBackEvent(IPostBackEventHandler sourceControl, String eventArgument) +15
   System.Web.UI.Page.RaisePostBackEvent(NameValueCollection postData) +35
   System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +1696
```

**Version Information:** Microsoft .NET Framework Version: 4.0.30319; ASP.NET Version: 4.8.4001.0

Lỗi *Invalid postback or callback argument* xảy ra khi xử lý 1 sự kiện trên GridView (trong ví dụ hiện tại là sự kiện RowCommand) mà trong sự kiện Page\_Load lại có thực hiện *binding*<sup>2</sup> dữ liệu cho GridView đó. Mỗi khi sự kiện RowCommand được gọi, trang web sẽ thực hiện *postback*<sup>3</sup> dữ liệu lên máy chủ, GridView sẽ thực hiện binding dữ liệu lại, do đó các dòng, các control trong GridView sẽ được gán id mới, khiến GridView không thể biết được control nào vừa gọi sự kiện RowCommand.

<sup>2</sup> Đổ dữ liệu từ CSDL vào GridView

<sup>3</sup> Postback là hành động gửi dữ liệu từ trang web lên máy chủ để xử lý

Cách khắc phục: Chỉ thực hiện binding dữ liệu cho GridView tại lần chạy đầu tiên, các lần chạy sau đó (các lần postback) sẽ không binding dữ liệu nữa.

Sự kiện Page\_Load được bổ sung như sau:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!Page.IsPostBack)
    {
        grvLoaiSanPham.DataSource = LoaiSanPhamBUS.LayDSLoaiSanPham();
        grvLoaiSanPham.DataBind();
    }
}
```

Sau khi đã chọn được 1 dòng và hiển thị thông tin dòng đó lên form, chúng ta cài đặt sự kiện Click cho nút *Sửa* như sau:

```
protected void btnSua_Click(object sender, EventArgs e)
{
    string maLoaiSP = txtMaLoaiSP.Text;
    string tenLoaiSP = txtTenLoaiSP.Text;
    bool trangThai = chkTrangThai.Checked;

    if (LoaiSanPhamBUS.SuaLoaiSanPham(maLoaiSP, tenLoaiSP, trangThai))
    {
        grvLoaiSanPham.DataSource = LoaiSanPhamBUS.LayDSLoaiSanPham();
        grvLoaiSanPham.DataBind();

        txtMaLoaiSP.Text = string.Empty;
        txtTenLoaiSP.Text = string.Empty;
        chkTrangThai.Checked = true;
        txtMaLoaiSP.Enabled = true;
    }
    else
    {
        // Sửa thất bại
    }
}
```

### 3. Chức năng Xóa

Cách thêm nút *Xóa* tương tự như nút *Chọn*, hoặc có thể bổ sung đoạn code sau vào mã nguồn ASP.NET ngay phía dưới nút *Chọn*:

```
<asp:TemplateField ShowHeader="False">
    <ItemTemplate>
        <asp:Button ID="btnChon" ... />
        <asp:Button ID="btnXoa" runat="server" CausesValidation="False"
            CommandName="XoaSP" CommandArgument='<%# Eval("MaLoaiSP") %>'
            Text="Xóa" OnClientClick="return confirm('Bạn có chắc chắn muốn
            xóa?');" />
    </ItemTemplate>
</asp:TemplateField>
```

Sự kiện ClientClick của nút Xóa sử dụng JavaScript để hiển thị cửa sổ xác nhận trước khi postback dữ liệu lên máy chủ và xóa thực sự.

Tại lớp DAO, chúng ta cài đặt phương thức xóa dữ liệu từ bảng LoaiSanPham:

```
public static bool XoaLoaiSanPham(string maLoaiSP)
{
    string query = "UPDATE LoaiSanPham SET TrangThai = 0 WHERE MaLoaiSP = @MaLoaiSP";
    SqlParameter[] param = new SqlParameter[1];
    param[0] = new SqlParameter("@MaLoaiSP", maLoaiSP);
    return DataProvider.ExecuteDeleteQuery(query, param) == 1;
}
```

Tại lớp BUS, chúng ta cài đặt phương thức gọi lớp DAO thực hiện thao tác xóa dữ liệu từ bảng LoaiSanPham, sau đó trả kết quả về cho lớp GUI:

```
public static bool SuaLoaiSanPham(string maLoaiSP, string tenLoaiSP, bool trangThai)
{
    if (LoaiSanPhamDAO.KTMaLoaiSanPhamTonTai(maLoaiSP))
    {
        return LoaiSanPhamDAO.XoaLoaiSanPham(maLoaiSP);
    }
    else
    {
        return false;
    }
}
```

Tại lớp GUI, chúng ta cài đặt thêm vào sự kiện RowCommand của GridView như sau:

```
protected void grvLoaiSanPham_RowCommand(object sender,
GridViewCommandEventArgs e)
{
    if (e.CommandName == "ChonSP") { ... }

    if (e.CommandName == "XoaSP")
    {
        string maLoaiSP = e.CommandArgument.ToString();

        if (LoaiSanPhamBUS.XoaLoaiSanPham(maLoaiSP))
        {
            grvLoaiSanPham.DataSource =
                LoaiSanPhamBUS.LayDSLoaiSanPham();
            grvLoaiSanPham.DataBind();
        }
        else
        {
            // Xóa thất bại
        }
    }
}
```