Ejercicio 5. Control de Registro

Desarrollaremos los siguientes requisitos:

1. En el pedido de compra, por cada producto, un usuario debe poder especificar:
    - La valoración general del producto
2. El sistema no debe permitir la recepción del pedido de compra si el usuario no ha rellenado toda la información relativa al control de calidad

En este ejercicio aprenderemos:

- A crear e utilizar Enums
- A desarrollar y ejecutar Tests para nuestros desarrollos
- A utilizar el nuevo comando **AL: Find event**

Indicaciones

Para completar el ejercicio realiza las siguientes acciones:

1. Crea una nueva rama de Git
2. Haz el desarrollo necesario para que el usuario pueda especificar la valoración general de un producto en un pedido de compra
3. Haz los tests necesarios para asegurar que se cumplen los requisitos
4. Haz el desarrollo necesario para no permitir registrar un pedido de compra si el usuario no ha rellenado toda la información relativa al control de calidad
5. Haz un merge de tu rama de desarrollo con la rama principal
6. Elimina la rama del desarrollo

Indicaciones paso a paso

Para completar el ejercicio sigue los siguientes pasos en Visual Studio Code:

1. Crea una nueva rama de Git llamada *Feature_PostingControl*
2. Creación de un campo de tipo Enum en las líneas de compra
    1. En la extensión principal, crea el siguiente objeto enum

```
enum 50100 "Clip Quality Control Result"
{
    Extensible = true;

    value(0; " ")
    {
        Caption = ' ', comment = 'ESP=" "';
    }
    value(1; Satisfactory)
    {
        Caption = 'Satisfactory', comment = 'ESP="Satisfactorio"';
    }
```

```
    value(2; "Non-Satisfactory")
    {
        Caption = 'Non-Satisfactory', comment = 'ESP="No
Satisfactorio"';
    }
}
```

2. En la extensión principal, crea un nuevo campo en la tabla de líneas de compra

```
tableextension 50101 "Clip Purchase Line" extends "Purchase Line"
{
    fields
    {
        field(50100; "Clip Quality Control Result"; enum "Clip Quality
Control Result")
        {
            Caption = 'Quality Control Result', comment =
'ESP="Resultado Control de Calidad"';
            DataClassification = CustomerContent;
        }
    }
}
```

3. En la extensión principal, muestra el campo anterior en la subpágina de líneas de pedido de compra

```
pageextension 50102 "Clip Purchase Order Subform" extends "Purchase
Order Subform"
{
    layout
    {
        addbefore("Qty. to Receive")
        {
            field("Clip Quality Control Result"; "Clip Quality Control
Result")
            {
                ApplicationArea = Basic, Suite;
                ToolTip = 'Indicates the global resul of the Quality
Control process', comment = 'ESP="Indica el resultado general del
control de calidad"';
            }
        }
    }
}
```

4. Haz un commit con los cambios realizados hasta el momento

3. Desarrollo de tests
    1. En la extensión de test, añade el siguiente procedimiento a la codeunit de test

```
[Test]
procedure PurchaseOrderPostingP004();
var
    PurchaseHeader: Record "Purchase Header";
    PurchaseLine: Record "Purchase Line";
    Item: Record Item;
    QualityControlLibrary: Codeunit "Clip Quality Control - Library";
    PurchaseLibrary: Codeunit "Library - Purchase";
    PurchaseDocumentType: Enum "Purchase Document Type";
    PostedDocumentNo: Code[20];
begin
    // [Scenario] The system will not allow to post the reception of a
purchase order when it contains at least one item that requires quality
control and the quality control information is not fully set

    // [Given] Setup: An Item that requires Quality Control
    //                A Purchase Order that uses the item
    //                No Quality Control information set on the order
    QualityControlLibrary.CreateItemWithQualityMeasures(Item);

    PurchaseLibrary.CreatePurchHeader(PurchaseHeader,
PurchaseDocumentType::Order.AsInteger(), '');
    PurchaseLibrary.CreatePurchaseLine(PurchaseLine, PurchaseHeader,
PurchaseLine.Type::Item.AsInteger(), Item."No.", 1);

    // [When] Exercise: Post the reception
    asserterror PostedDocumentNo :=
PurchaseLibrary.PostPurchaseDocument(PurchaseHeader, true, false);

    // [Then] Verify: The order has not been posted
    AssertThat.AreEqual('', PostedDocumentNo, 'C01.P004.A001
PostedDocumentNo');
    AssertThat.AreEqual('All the information related to Quality Control
must be filled in in line 10000', GetLastErrorText(), 'C01.P004.A002
Error Message');
end;
```

2. Publica la extensión de test
3. Ejecuta el siguiente comando en el terminal

```
./RunTest.ps1
```

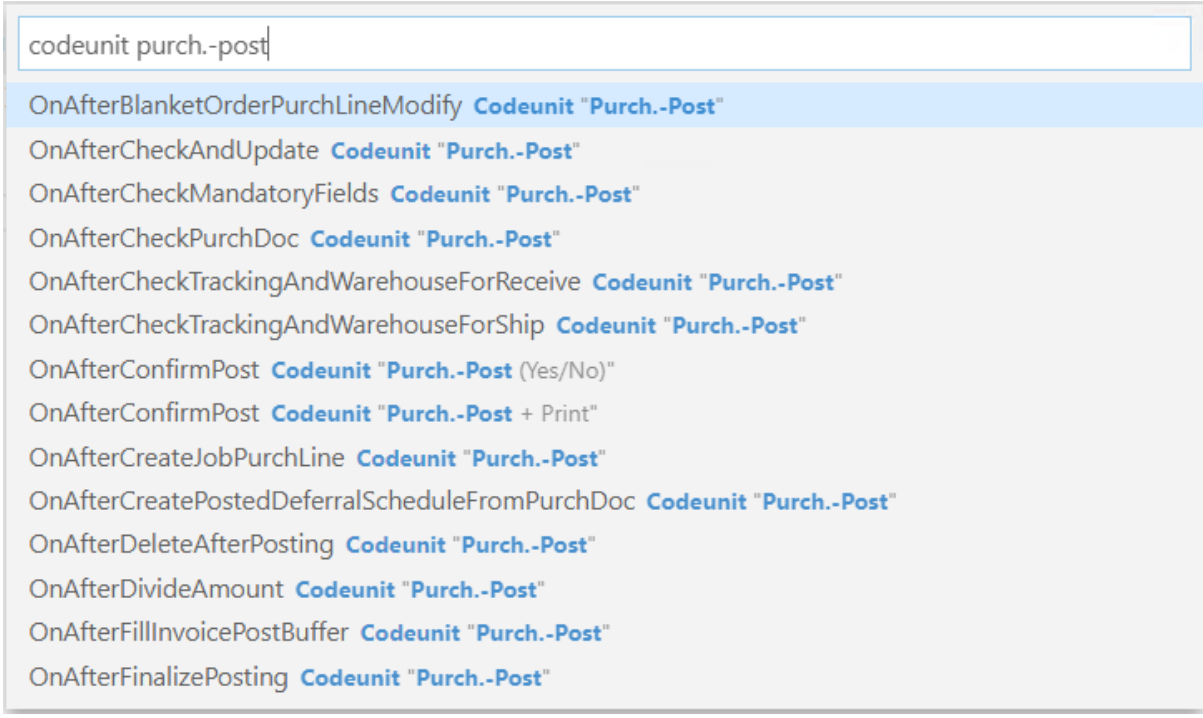
4. Comprueba que el test desarrollado falla

4. Desarrollo de un control en el proceso de registro de un pedido de compra
    1. En la extensión principal, crea la siguiente codeunit

```
codeunit 50102 "Clip Purchase Post"
{
```

```
}
```

2. Ejecuta el comando **AL: Find event** para ver una lista completa de los eventos a los que te puedes suscribir. Escribe *codeunit purch.-post* para ver sólo eventos de la codeunit de registro de las compras



3. Selecciona el evento **OnBeforePostPurchaseDoc** y el sistema creará la función suscrita al evento seleccionado
4. Escribe el siguiente código en la codeunit

```
codeunit 50102 "Clip Purchase Post"
{
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"Purch.-Post",
'OnBeforePostPurchaseDoc', '', false, false)]
    local procedure OnBeforePostPurchaseDoc(var Sender: Codeunit
"Purch.-Post"; var PurchaseHeader: Record "Purchase Header";
PreviewMode: Boolean; CommitIsSupressed: Boolean; var
HideProgressWindow: Boolean);
    var
        PurchaseLine: Record "Purchase Line";
        PurchaseQualityMeasures: Record "Clip Purchase Quality
Measures";
        QualityControlIsNotFullyFilledInErr: Label 'All the information
related to Quality Control must be filled in in line %1', Comment =
'ESP="Debe rellenar toda la información del control de calidad en la
línea %1"';
    begin
        if PurchaseHeader."Document Type" <> PurchaseHeader."Document
Type"::Order then
            exit;

        if not PurchaseHeader.Receive then
```

```
            exit;

        PurchaseLine.SetRange("Document Type", PurchaseHeader."Document
Type");
        PurchaseLine.SetRange("Document No.", PurchaseHeader."No.");
        PurchaseLine.SetRange(Type, PurchaseLine.Type::Item);
        PurchaseLine.SetFilter("Qty. to Receive", '<>0');
        if PurchaseLine.FindSet() then
            repeat
                if ItemRequieresQualityControl(PurchaseLine."No.") then
begin
                    if PurchaseLine."Clip Quality Control Result" =
PurchaseLine."Clip Quality Control Result"::" " then
                        Error(QualityControlIsNotFullyFilledInErr,
PurchaseLine."Line No.");

                    PurchaseQualityMeasures.SetRange("Document Type",
PurchaseLine."Document Type");
                    PurchaseQualityMeasures.SetRange("Document No.",
PurchaseLine."Document No.");
                    PurchaseQualityMeasures.SetRange("Line No.",
PurchaseLine."Line No.");
                    PurchaseQualityMeasures.SetFilter(Value, '%1', '');
                    if not PurchaseQualityMeasures.IsEmpty() then
                        Error(QualityControlIsNotFullyFilledInErr,
PurchaseLine."Line No.");
                end;
            until PurchaseLine.Next() = 0;
    end;

    local procedure ItemRequieresQualityControl(ItemNo: Code[20]):
Boolean
    var
        Item: Record Item;
    begin
        Item.Get(ItemNo);
        exit(Item."Clip Requires Quality Control");
    end;
}
```

5. Amplia el test para controlar que el error no salte cuando se registra un pedido de compra con productos que no requieren control de calidad

```
[Test]
procedure PurchaseOrderPostingP005();
var
    PurchaseHeader: Record "Purchase Header";
    PurchaseLine: Record "Purchase Line";
    Item: Record Item;
    PurchRcptHeader: Record "Purch. Rcpt. Header";
    QualityControlLibrary: Codeunit "Clip Quality Control - Library";
```

```
    PurchaseLibrary: Codeunit "Library - Purchase";
    PurchaseDocumentType: Enum "Purchase Document Type";
    PostedDocumentNo: Code[20];
begin
    // [Scenario] The system will allow to post the reception of a
purchase order when it contains items that do not require quality
control

    // [Given] Setup: An Item that does not require Quality Control
    //                A Purchase Order that uses the item
    //                No Quality Control information set on the order
    QualityControlLibrary.CreateItemWithQualityMeasures(Item);
    Item."Clip Requires Quality Control" := false;
    Item.Modify();

    PurchaseLibrary.CreatePurchHeader(PurchaseHeader,
PurchaseDocumentType::Order.AsInteger(), '');
    PurchaseLibrary.CreatePurchaseLine(PurchaseLine, PurchaseHeader,
PurchaseLine.Type::Item.AsInteger(), Item."No.", 1);

    // [When] Exercise: Post the reception
    PostedDocumentNo :=
PurchaseLibrary.PostPurchaseDocument(PurchaseHeader, true, false);

    // [Then] Verify: The order has not been posted
    AssertThat.AreNotEqual('', PostedDocumentNo, 'C01.P005.A001
PostedDocumentNo');
    AssertThat.AreEqual(true, PurchRcptHeader.Get(PostedDocumentNo),
'C01.P005.A002 PurchRcptHeader.Get');
end;
```

6. Amplia el test para controlar que el error no salte cuando se registra un pedido de compra con productos que requieren control de calidad y para los que el usuario ha introducido toda la información necesaria

```
[Test]
procedure PurchaseOrderPostingP006();
var
    PurchaseHeader: Record "Purchase Header";
    PurchaseLine: Record "Purchase Line";
    Item: Record Item;
    PurchRcptHeader: Record "Purch. Rcpt. Header";
    QualityControlLibrary: Codeunit "Clip Quality Control - Library";
    PurchaseLibrary: Codeunit "Library - Purchase";
    PurchaseDocumentType: Enum "Purchase Document Type";
    PostedDocumentNo: Code[20];
begin
    // [Scenario] The system will allow to post the reception of a
purchase order when the quality control information is fully set

    // [Given] Setup: An Item that requires Quality Control
```

```
    //                A Purchase Order that uses the item
    //                Quality Control information set on the order
    QualityControlLibrary.CreateItemWithQualityMeasures(Item);

    PurchaseLibrary.CreatePurchHeader(PurchaseHeader,
PurchaseDocumentType::Order.AsInteger(), '');
    PurchaseLibrary.CreatePurchaseLine(PurchaseLine, PurchaseHeader,
PurchaseLine.Type::Item.AsInteger(), Item."No.", 1);


QualityControlLibrary.SetQualityControlInformationOnPurchaseLine(Purcha
seLine, PurchaseLine."Clip Quality Control Result"::Satisfactory);

    // [When] Exercise: Post the reception
    PostedDocumentNo :=
PurchaseLibrary.PostPurchaseDocument(PurchaseHeader, true, false);

    // [Then] Verify: The order has been posted
    AssertThat.AreNotEqual('', PostedDocumentNo, 'C01.P006.A001
PostedDocumentNo');
    AssertThat.AreEqual(true, PurchRcptHeader.Get(PostedDocumentNo),
'C01.P006.A002 PurchRcptHeader.Get');
end;
```

7. Haz un merge de tu rama de desarrollo con la rama principal
8. Elimina la rama de desarrollo