Desarrollo desde 0

Dynamics 365 Business Central

- La sesión empezará en breve -

ClipPlatform

Desarrollo – Registro

ClipPlatform

DES03 Registro de Cursos

El proceso de registro de un documento de venta, en caso de estar vendiendo un curso, deberá crear un Movimiento de Curso de una forma similar a como el estándar de Business Central crea un Movimiento de Producto cuando se venden Productos o un Movimiento de Recurso cuando se venden Recursos.

Para ello, se creará:

- Una tabla de Movimientos de Curso con la siguiente información:

Campo	Tipo de Datos	Comentario
Entry No.	Integer	Clave Primaria. Se autorellenará empezando en el número 1 e incrementando en 1 en cada nuevo movimiento
Document No.	Code 20	Se autorellenará con el Nº Documento de la factura de venta
Posting Date	Date	Se autorellenará con la Fecha de Registro de la factura de venta
Course No.	Code 20	Debe tener relación con la tabla de cursos. Se autorellenará con el curso seleccionado en la línea de la factura de venta
Course Edition	Code 20	Debe tener relación con la tabla de ediciones de curso. Se autorellenará con la edición seleccionada en la línea de la factura de venta
Description	Text 100	Se autorellenará con la descripción en la línea de la factura de venta
Quantity	Decimal	Se autorellenará con la cantidad en la línea de la factura de venta
Unit Price (LCY)	Decimal	Se autorellenará con el precio unitario de la factura de venta
Amount (LCY)	Decimal	Debe ser el resultado de la multiplicación de Cantidad * Precio Unitario, redondeando el resultado según las normas estipuladas por la Divisa Local

Los movimientos de curso deben contar con una página de tipo lista que debe ser accesible:
- Directamente en el buscador
- Desde una acción en la lista y en la ficha de los cursos

```
table 50103 "CLIP Course Ledger Entry"
{
    Caption = 'Course Ledger Entry';
    // DrillDownPageID = "Course Ledger Entries";
    // LookupPageID = "Course Ledger Entries";
    DataClassification = CustomerContent;

    fields
    {
        field(1; "Entry No."; Integer) { Caption = 'Entry No.'; }
        field(3; "Document No."; Code[20]) { Caption = 'Document No.'; }
        field(4; "Posting Date"; Date) { Caption = 'Posting Date'; }
        field(5; "Course No."; Code[20]) { Caption = 'Course No.'; TableRelation = "CLIP Course"; }
        field(7; Description; Text[100]) { Caption = 'Description'; }
        field(11; Quantity; Decimal) { Caption = 'Quantity'; DecimalPlaces = 0 : 5; }
        field(15; "Unit Price"; Decimal) { AutoFormatType = 2; Caption = 'Unit Price'; }
        field(16; "Total Price"; Decimal) { AutoFormatType = 1; Caption = 'Total Price'; }
    }

    keys
    {
        key(Key1; "Entry No.") { Clustered = true; }
        key(Key2; "Course No.", "Posting Date") { }
    }

    fieldgroups
    {
        fieldgroup(DropDown; "Entry No.", Description, "Document No.", "Posting Date") { }
    }

    procedure GetLastEntryNo(): Integer;
    var
        FindRecordManagement: Codeunit "Find Record Management";
    begin
        exit(FindRecordManagement.GetLastEntryIntFieldValue(Rec, FieldNo("Entry No.")))
    end;

    // procedure CopyFromResJnlLine(CourseJnlLine: Record "CLIP Course Journal Line")
    // begin
    //     "Document No." := CourseJnlLine."Document No.";
    //     "Posting Date" := CourseJnlLine."Posting Date";
    //     "Course No." := CourseJnlLine."Course No.";
    //     Description := CourseJnlLine.Description;
    //     Quantity := CourseJnlLine.Quantity;
    //     "Unit Price" := CourseJnlLine."Unit Price";
    //     "Total Price" := CourseJnlLine."Total Price";

    //     OnAfterCopyFromCourseJnlLine(Rec, CourseJnlLine);
    // end;

    // [IntegrationEvent(false, false)]
    // procedure OnAfterCopyFromCourseJnlLine(var CourseLedgerEntry: Record "CLIP Course Ledger Entry"; CourseJournalLine: Record "CLIP Course Journal Line")
    // begin
    // end;
}
```

Desarrollo – Registro

ClipPlatform

```
page 50104 "CLIP Course Ledger Entries"
{
    ApplicationArea = All;
    Caption = 'Course Ledger Entries';
    PageType = List;
    SourceTable = "CLIP Course Ledger Entry";
    UsageCategory = History;
    Editable = false;

    layout
    {
        area(content)
        {
            repeater(General)
            {
                field("Entry No."; Rec."Entry No.") { ToolTip = 'Specifies the value of the Entry No. field.', Comment = 'ESP="Nº Mov."'; ApplicationArea = All; }
                field("Posting Date"; Rec."Posting Date") { ToolTip = 'Specifies the value of the Posting Date field.', Comment = 'ESP="Fecha Registro"'; ApplicationArea = All; }
                field("Document No."; Rec."Document No.") { ToolTip = 'Specifies the value of the Document No. field.', Comment = 'ESP="Nº Documento"'; ApplicationArea = All; }
                field("Course No."; Rec."Course No.") { ToolTip = 'Specifies the value of the Course No. field.', Comment = 'ESP="Nº Curso"'; ApplicationArea = All; }
                field(Description; Rec.Description) { ToolTip = 'Specifies the value of the Description field.', Comment = 'ESP="Descripción"'; ApplicationArea = All; }
                field(Quantity; Rec.Quantity) { ToolTip = 'Specifies the value of the Quantity field.', Comment = 'ESP="Cantidad"'; ApplicationArea = All; }
                field("Unit Price"; Rec."Unit Price") { ToolTip = 'Specifies the value of the Unit Price field.', Comment = 'ESP="Precio Unitario"'; ApplicationArea = All; }
                field("Total Price"; Rec."Total Price") { ToolTip = 'Specifies the value of the Total Price field.', Comment = 'ESP="Precio Total"'; ApplicationArea = All; }
            }
        }
    }
}
```

Desarrollo – Registro

ClipPlatform

```
actions
{
    area(Navigation)
    {
        action(Entries)
        {
            Caption = 'Entries', comment = 'ESP="Movimientos"';
            ApplicationArea = All;
            RunObject = page "CLIP Course Ledger Entries";
            RunPageLink = "Course No." = field("No.");
            Image = Entries;
            ToolTip = 'Show the course entries', comment = 'ESP="Muestra los movimientos del curso"';
        }
    }
}
```

Desarrollo – Registro

ClipPlatform

- Haz un commit con los cambios

Desarrollo – Registro

ClipPlatform

Los movimientos de curso sólo se crearán cuando un curso se esté facturando (o abonando) un documento de venta. No deberán crearse movimientos de curso en la creación de albaranes de venta (o albaranes de devolución).

- Un proceso de registro
  El proceso de registro de un curso se realizará utilizando la misma estructura que Business Central utiliza para el proceso de registro de productos o recursos.
  Es decir, el proceso de registro de venta creará líneas de diario de cursos y ejecutará el proceso de registro de la línea de diario.
  El proceso de registro de la línea de diario de curso es el que tomará la información de una línea de diario y acabará creando el movimiento de curso.

  Para ello, será necesario crear:
  - Una tabla de Líneas de Diario de Curso
  - Una codeunit suscrita al registro de ventas que cree Líneas de Diario de Curso dada una línea de venta
  - Una codeunit de registro de Líneas de Diario de Curso que cree Movimientos de Curso dada una Línea de Diario de Curso

Testing – Registro

ClipPlatform

- Realiza un test que compruebe que se crean correctamente movimientos de curso

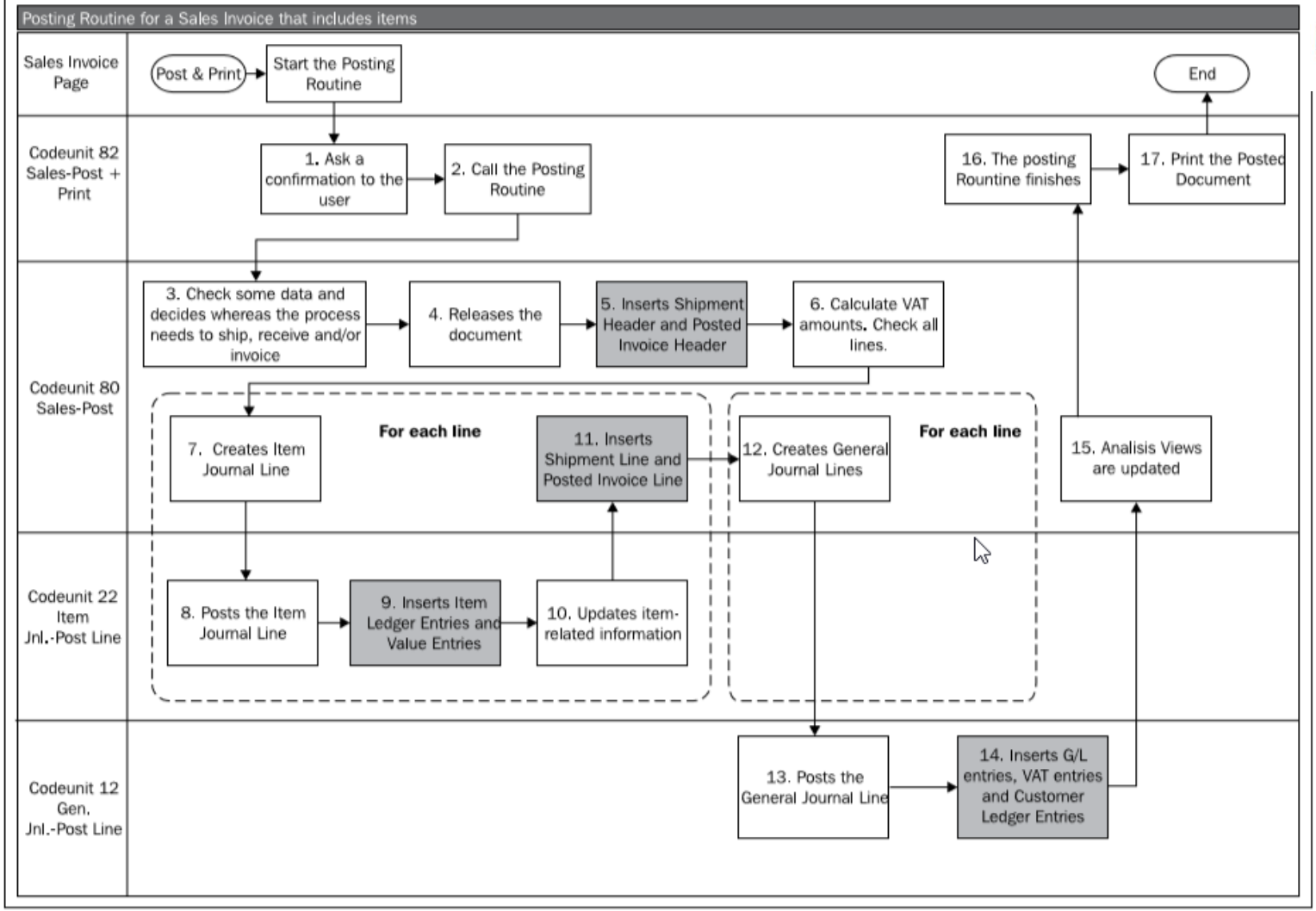
```
[Test]
procedure CourseSalesPostingP003()
var
    Course: Record "CLIP Course";
    CourseEdition: Record "CLIP Course Edition";
    SalesHeader: Record "Sales Header";
    SalesLine: Record "Sales Line";
    CourseLedgerEntry: Record "CLIP Course Ledger Entry";
    LibraryCourse: Codeunit "CLIP Library - Course";
    LibrarySales: Codeunit "Library - Sales";
    PostedDocumentNo: Code[20];
begin
    // [Scenario] when posting a sale of a course and edition, a course ledger entry is created

    // [Given] Setup: A course with Edition
    Course := LibraryCourse.CreateCourse();
    CourseEdition := LibraryCourse.CreateEdition(Course);

    // [Given] Setup: A Sales Document for the Course and Edition
    LibrarySales.CreateSalesHeader(SalesHeader, "Sales Document Type"::Order, '');
    LibrarySales.CreateSalesLine(SalesLine, SalesHeader, "Sales Line Type"::"CLIP Course", Course."No.", 1);
    SalesLine.Validate("CLIP Course Edition", CourseEdition.Edition);
    SalesLine.Modify(true);

    // [When] Exercise: Post the Sales Document
    PostedDocumentNo := LibrarySales.PostSalesDocument(SalesHeader, true, true);

    // [Then] Verify: A course ledger entry is created
    CourseLedgerEntry.SetRange("Document No.", PostedDocumentNo);
    LibraryAssert.AreEqual(1, CourseLedgerEntry.Count(), 'Nº of CourseLedgerEntry is incorrect');
    CourseLedgerEntry.FindFirst();
    LibraryAssert.AreEqual(SalesHeader."Posting Date", CourseLedgerEntry."Posting Date", 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine."No.", CourseLedgerEntry."Course No.", 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine."CLIP Course Edition", CourseLedgerEntry."Course Edition", 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine.Description, CourseLedgerEntry.Description, 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine.Quantity, CourseLedgerEntry.Quantity, 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine."Unit Price", CourseLedgerEntry."Unit Price", 'Incorrect data');
    LibraryAssert.AreEqual(SalesLine.Amount, CourseLedgerEntry."Total Price", 'Incorrect data');
end;
```



Desarrollo – Registro

ClipPlatform

- Debuga el proceso de registro de venta de un Recurso
  - Suscríbete al evento `OnPostSalesLineOnAfterCaseType` de la codeunit 80 “Sales-Post” y crea una función similar a `PostResJnlLine`
    - » Crea la codeunit “Course Jnl.-Post Line”
    - » Crea la tabla “Course Journal Line”

Desarrollo – Registro

ClipPlatform

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"Sales-Post", 'OnPostSalesLineOnAfterCaseType', '', false, false)]
local procedure OnPostSalesLineOnAfterCaseType(SalesHeader: Record "Sales Header"; var SalesLine: Record "Sales
Line"; GenJnlLineDocNo: Code[20]);
begin
    PostCourseJnlLine(SalesHeader, SalesLine, GenJnlLineDocNo);
end;

local procedure PostCourseJnlLine(var SalesHeader: Record "Sales Header"; var SalesLine: Record "Sales Line";
GenJnlLineDocNo: Code[20])
var
    CourseJournalLine: Record "CLIP Course Journal Line";
    CourseJnlPostLine: Codeunit "CLIP Course Jnl.-Post Line";
begin
    if SalesLine."Qty. to Invoice" = 0 then
        exit;

    CourseJournalLine.Init();
    CourseJournalLine.CopyFromSalesHeader(SalesHeader);
    CourseJournalLine.CopyDocumentFields(GenJnlLineDocNo);
    CourseJournalLine.CopyFromSalesLine(SalesLine);

    CourseJnlPostLine.RunWithCheck(CourseJournalLine);
end;
```

Desarrollo – Registro

ClipPlatform

```
codeunit 50101 "CLIP Course Jnl.-Post Line"
{
    Permissions = TableData "CLIP Course Ledger Entry" = imd,
                  TableData "Resource Register" = imd;
    TableNo = "CLIP Course Journal Line";

    trigger OnRun()
    begin
        RunWithCheck(Rec);
    end;

    var
        CourseJournalLine: Record "CLIP Course Journal Line";
        CourseLedgerEntry: Record "CLIP Course Ledger Entry";
        Course: Record "CLIP Course";
        NextEntryNo: Integer;

    procedure RunWithCheck(var CourseJournalLine2: Record "CLIP Course Journal Line")
    begin
        CourseJournalLine.Copy(CourseJournalLine2);
        Code();
        CourseJournalLine2 := CourseJournalLine;
    end;

    local procedure "Code"()
    begin
        if CourseJournalLine.EmptyLine() then
            exit;

        if NextEntryNo = 0 then begin
            CourseLedgerEntry.LockTable();
            NextEntryNo := CourseLedgerEntry.GetLastEntryNo() + 1;
        end;

        Course.Get(CourseJournalLine."Course No.");

        CourseLedgerEntry.Init();
        CourseLedgerEntry.CopyFromResJnlLine(CourseJournalLine);
        CourseLedgerEntry."Total Price" := Round(CourseLedgerEntry."Total Price");
        CourseLedgerEntry."Entry No." := NextEntryNo;
        CourseLedgerEntry.Insert(true);

        NextEntryNo := NextEntryNo + 1;
    end;
}
```

Desarrollo – Registro

ClipPlatform

```
table 50104 "CLIP Course Journal Line"
{
    Caption = 'Course Journal Line';
    DataClassification = CustomerContent;

    fields
    {
        field(1; "Journal Template Name"; Code[10])
        {
            Caption = 'Journal Template Name';
            TableRelation = "Res. Journal Template";
        }
        field(2; "Line No."; Integer)
        {
            Caption = 'Line No.';
        }
        field(4; "Document No."; Code[20])
        {
            Caption = 'Document No.';
        }
        field(5; "Posting Date"; Date)
        {
            Caption = 'Posting Date';

            trigger OnValidate()
            begin
                TestField("Posting Date");
            end;
        }
        field(6; "Course No."; Code[20])
        {
            Caption = 'Course No.';
            TableRelation = "CLIP Course";

            trigger OnValidate()
            begin
                Course.Get("Course No.");
                Description := Course.Name;
                "Unit Price" := Course.Price;
                "Gen. Prod. Posting Group" := Course."Gen. Prod. Posting Group";
            end;
        }
        field(7; "Course Edition"; Code[20])
        {
            Caption = 'Course Edition', comment = 'ESP="Edición Curso"';
            TableRelation = "CLIP Course Edition".Edition where("Course No." =
field("Course No."));
        }
        field(8; Description; Text[100])
        {
            Caption = 'Description';
        }
        field(12; Quantity; Decimal)
        {
            Caption = 'Quantity';
            DecimalPlaces = 0 : 5;

            trigger OnValidate()
            begin
                Validate("Unit Price");
            end;
        }
        field(16; "Unit Price"; Decimal)
        {
            AutoFormatType = 2;
            Caption = 'Unit Price';
            MinValue = 0;

            trigger OnValidate()
            begin
                "Total Price" := Quantity * "Unit Price";
            end;
        }
        field(17; "Total Price"; Decimal)
        {
            AutoFormatType = 1;
            Caption = 'Total Price';

            trigger OnValidate()
            begin
                TestField(Quantity);
                GetGLSetup();
                "Unit Price" := Round("Total Price" / Quantity,
GeneralLedgerSetup."Unit-Amount Rounding Precision");
            end;
        }
        field(23; "Journal Batch Name"; Code[10])
        {
            Caption = 'Journal Batch Name';
            TableRelation = "Res. Journal Batch".Name WHERE("Journal Template Name" =
FIELD("Journal Template Name"));
        }
        field(28; "Gen. Bus. Posting Group"; Code[20])
        {
            Caption = 'Gen. Bus. Posting Group';
            TableRelation = "Gen. Business Posting Group";
        }
        field(29; "Gen. Prod. Posting Group"; Code[20])
        {
            Caption = 'Gen. Prod. Posting Group';
            TableRelation = "Gen. Product Posting Group";
        }
    }

    keys
    {
        key(Key1; "Journal Template Name", "Journal Batch Name", "Line No.")
        {
            Clustered = true;
        }
    }

    fieldgroups
    {
    }

    trigger OnInsert()
    begin
        LockTable();
        ResJournalTemplate.Get("Journal Template Name");
        ResJournalBatch.Get("Journal Template Name", "Journal Batch Name");
    end;

    var
        ResJournalTemplate: Record "Res. Journal Template";
        ResJournalBatch: Record "Res. Journal Batch";
        Course: Record "CLIP Course";
        GeneralLedgerSetup: Record "General Ledger Setup";
        GLSetupRead: Boolean;

    procedure EmptyLine(): Boolean
    begin
        exit(("Course No." = '') and (Quantity = 0));
    end;

    procedure CopyDocumentFields(DocNo: Code[20])
    begin
        "Document No." := DocNo;
    end;

    procedure CopyFromSalesHeader(SalesHeader: Record "Sales Header")
    begin
        "Posting Date" := SalesHeader."Posting Date";
    end;

    procedure CopyFromSalesLine(SalesLine: Record "Sales Line")
    begin
        "Course No." := SalesLine."No.";
        "Course Edition" := SalesLine."CLIP Course Edition";
        Description := SalesLine.Description;
        "Gen. Bus. Posting Group" := SalesLine."Gen. Bus. Posting Group";
        "Gen. Prod. Posting Group" := SalesLine."Gen. Prod. Posting Group";
        Quantity := -SalesLine."Qty. to Invoice";
        "Unit Price" := SalesLine."Unit Price";
        "Total Price" := -SalesLine.Amount;
    end;

    local procedure GetGLSetup()
    begin
        if not GLSetupRead then
            GeneralLedgerSetup.Get();
        GLSetupRead := true;
    end;
}
```