Microsoft Dynamics 365

Microsoft workshop:

Extend E-document core solution to suit your requirements

Aleksandar Totovic
Predrag Maricic
Wael AbuSeada

directions lyon
emea 2023

Who are we?

Aleksandar Totovic

Product Manager @Microsoft

Predrag Maricic

Principal engineering manager @Microsoft

Wael AbuSeada

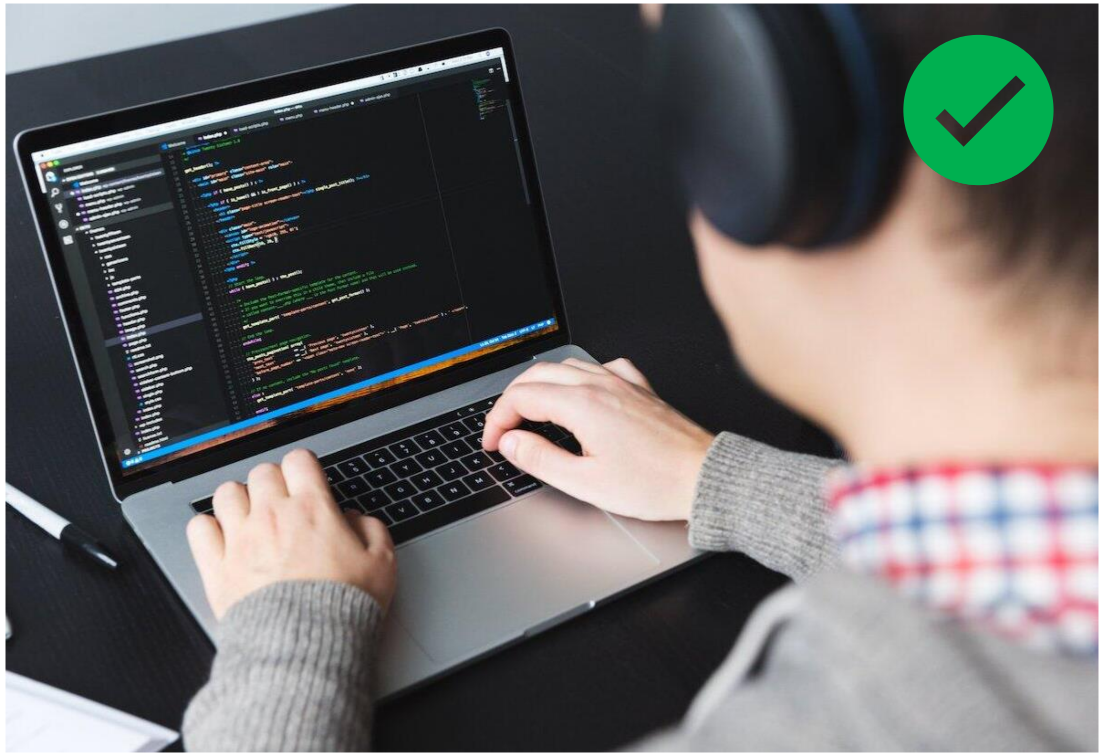
“Legendary” Software Engineer @ Microsoft

Agenda

- Introduction
- Architecture
- Setup dev environment
- Build E-Document format
- Integrate to an endpoint
- Show your work

Introduction

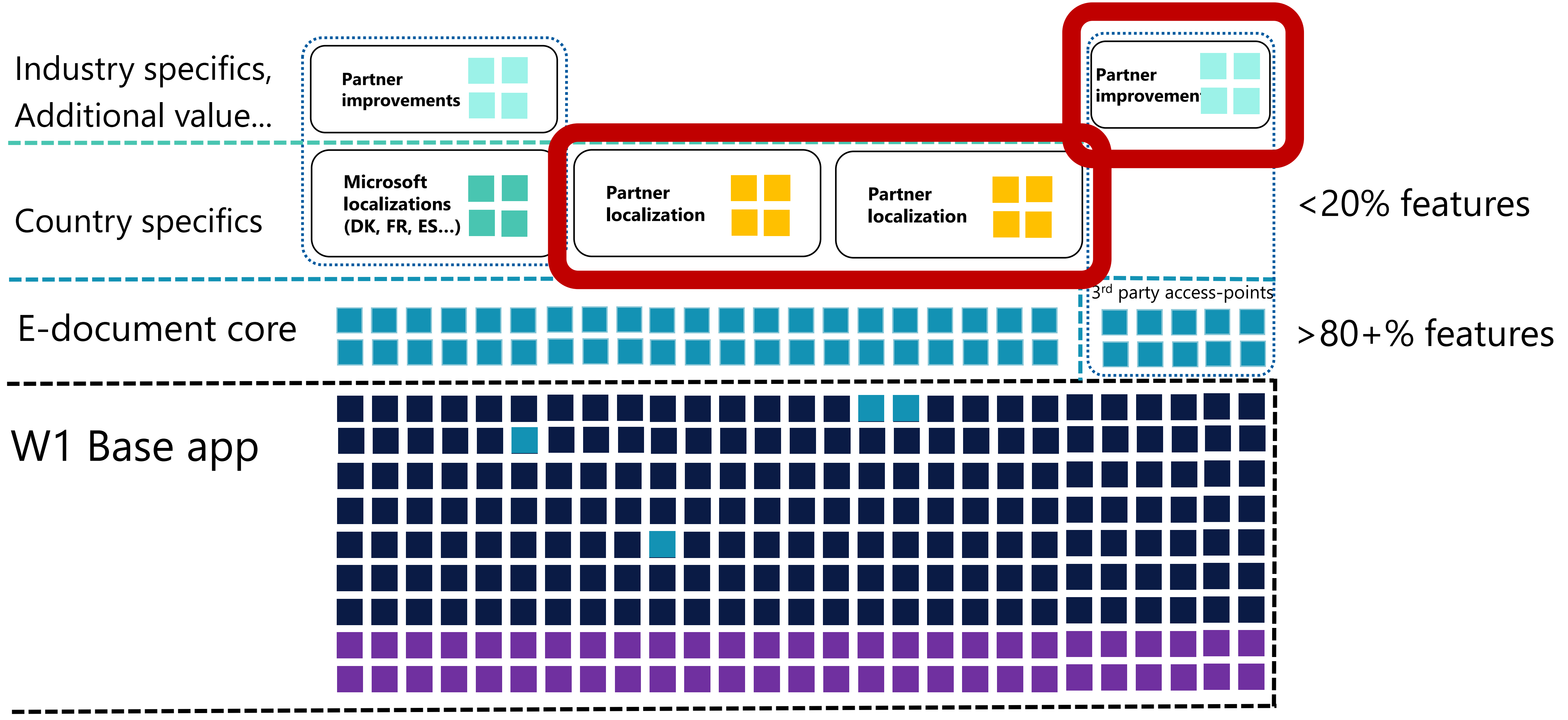
Who is the workshop intended for?



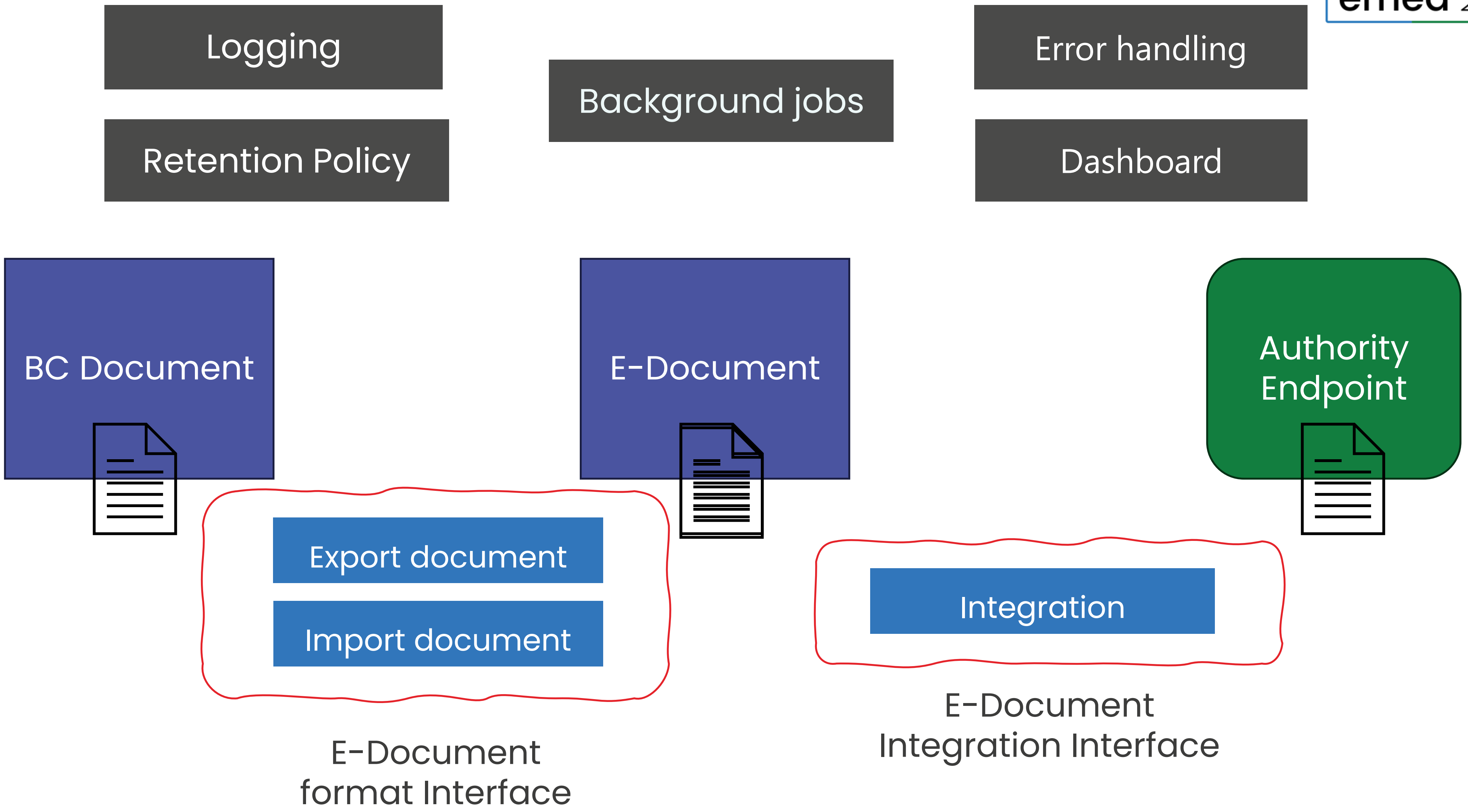
Developers

Consultants, Sales, Managers...

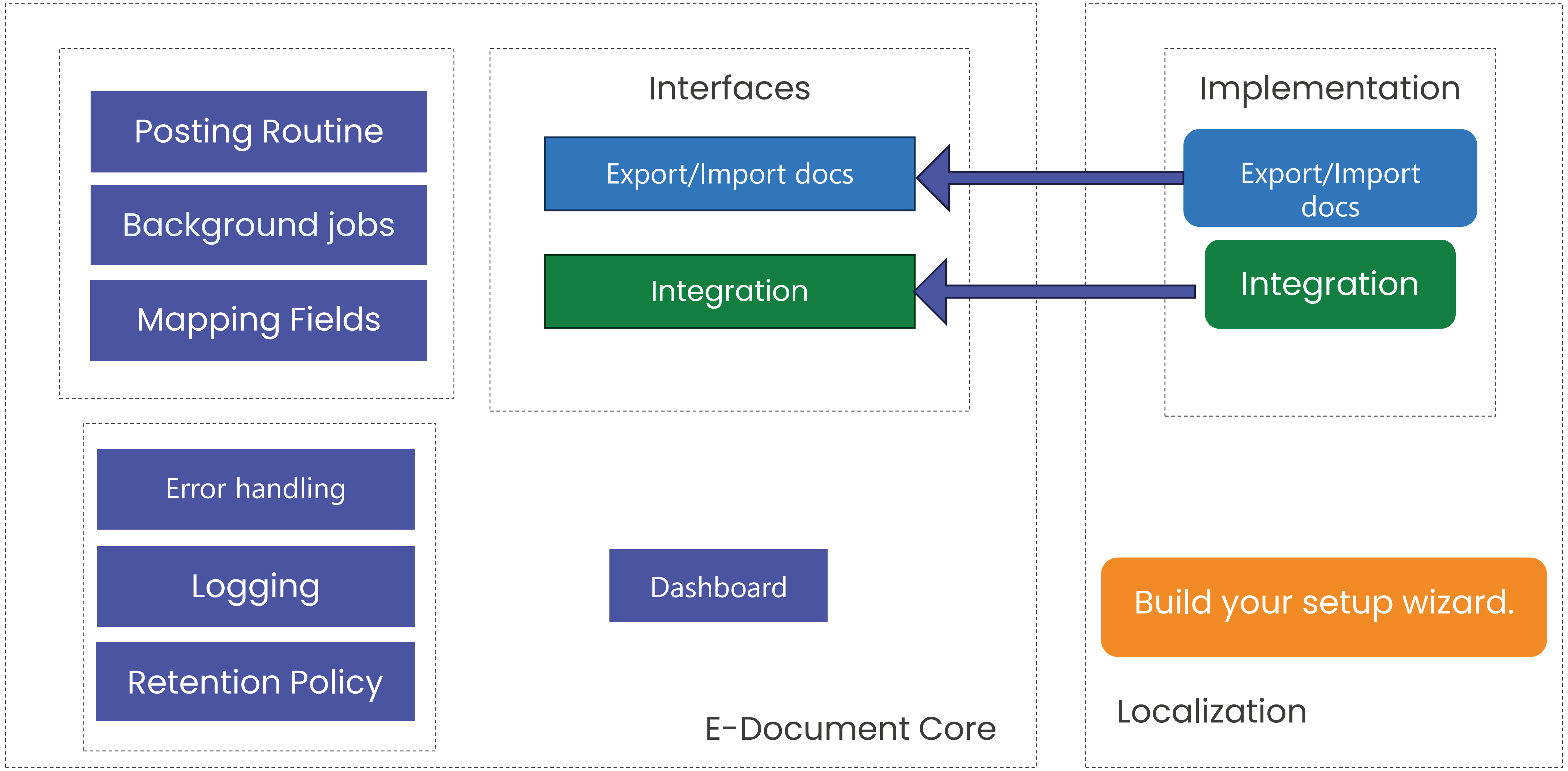
Who is the workshop intended for?



Architecture


Logging
Retention Policy
Background jobs
Error handling
Dashboard
BC Document
E-Document
Authority Endpoint
Export document
Import document
Integration
E-Document format Interface
E-Document Integration Interface

Architecture



Step by step guide to create your own localization app

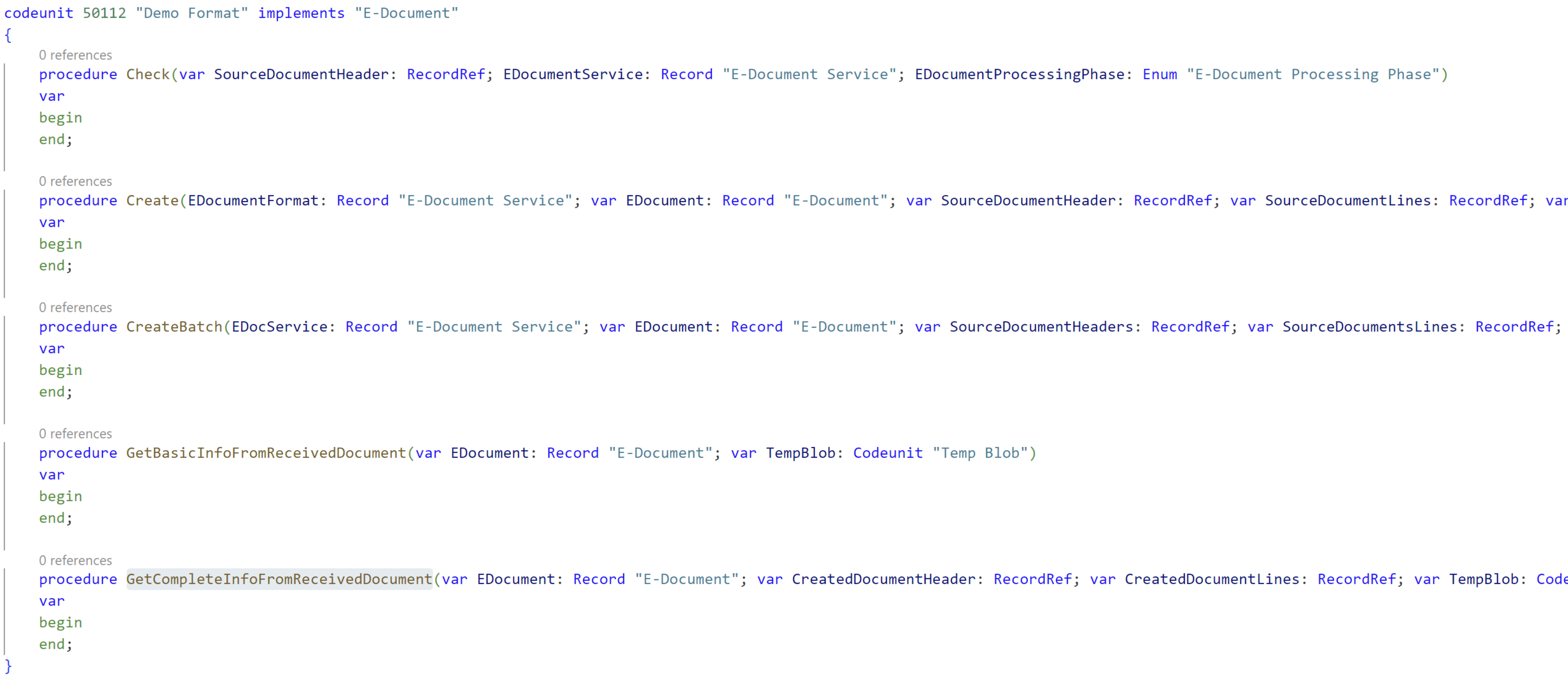
Initial implementation

Initial implementation

- Create new app
- Add dependency on “E-Document core”
- Extend Format enum and create implementation codeunit
- Extend Integration enum and create implementation codeunit

E-document Format interface

E-document Format interface



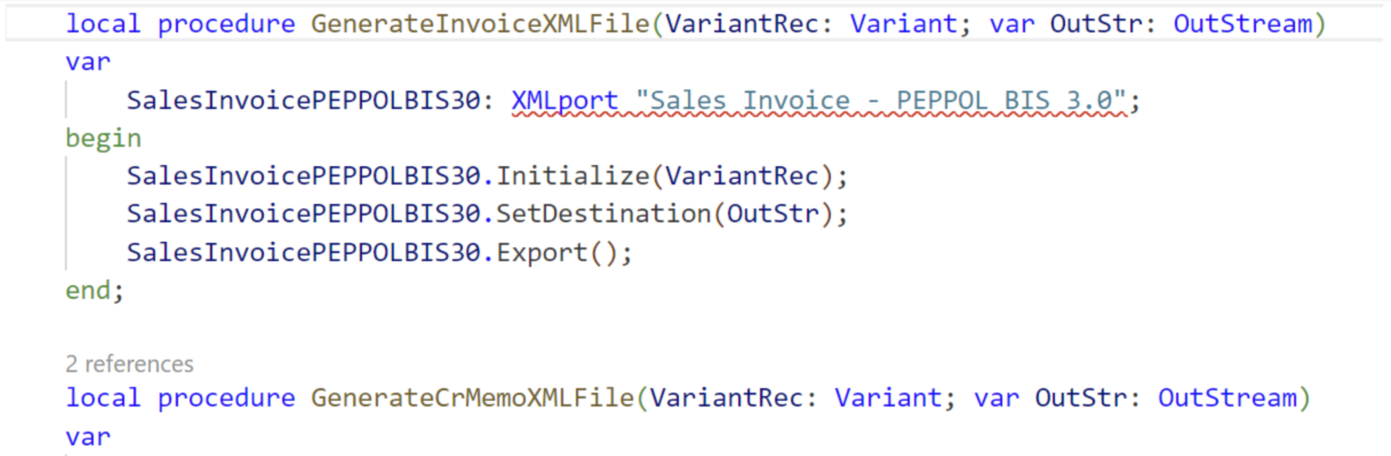
```
codeunit 50112 "Demo Format" implements "E-Document"
{
    0 references
    procedure Check(var SourceDocumentHeader: RecordRef; EDocumentService: Record "E-Document Service"; EDocumentProcessingPhase: Enum "E-Document Processing Phase")
    var
    begin
    end;

    0 references
    procedure Create(EDocumentFormat: Record "E-Document Service"; var EDocument: Record "E-Document"; var SourceDocumentHeader: RecordRef; var SourceDocumentLines: RecordRef; var
    var
    begin
    end;

    0 references
    procedure CreateBatch(EDocService: Record "E-Document Service"; var EDocument: Record "E-Document"; var SourceDocumentHeaders: RecordRef; var SourceDocumentsLines: RecordRef;
    var
    begin
    end;

    0 references
    procedure GetBasicInfoFromReceivedDocument(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob")
    var
    begin
    end;

    0 references
    procedure GetCompleteInfoFromReceivedDocument(var EDocument: Record "E-Document"; var CreatedDocumentHeader: RecordRef; var CreatedDocumentLines: RecordRef; var TempBlob: Code
    var
    begin
    end;
}
```

Sending - Check

```
procedure Check(var SourceDocumentHeader: RecordRef; EDocumentService: Record "E-Document Service"; EDocumentProcessingPhase: Enum "E-Document Processing Phase")
var
    SalesHeader: Record "Sales Header";
begin

    Case SourceDocumentHeader.Number of
        Database::"Sales Header":
            case EDocumentProcessingPhase of
                EDocumentProcessingPhase::Release:
                    begin
                        SourceDocumentHeader.Field(SalesHeader.FieldNo("Customer Posting Group")).TestField();
                        SourceDocumentHeader.Field(SalesHeader.FieldNo("Posting Date")).TestField();
                    end;
                EDocumentProcessingPhase::Post:
                    begin
                        SourceDocumentHeader.Field(SalesHeader.FieldNo("Customer Posting Group")).TestField();
                        SourceDocumentHeader.Field(SalesHeader.FieldNo("Posting Date")).TestField();
                        SourceDocumentHeader.Field(SalesHeader.FieldNo("Bill-to Name")).TestField();
                    end;
            end;
    End;
end;
```

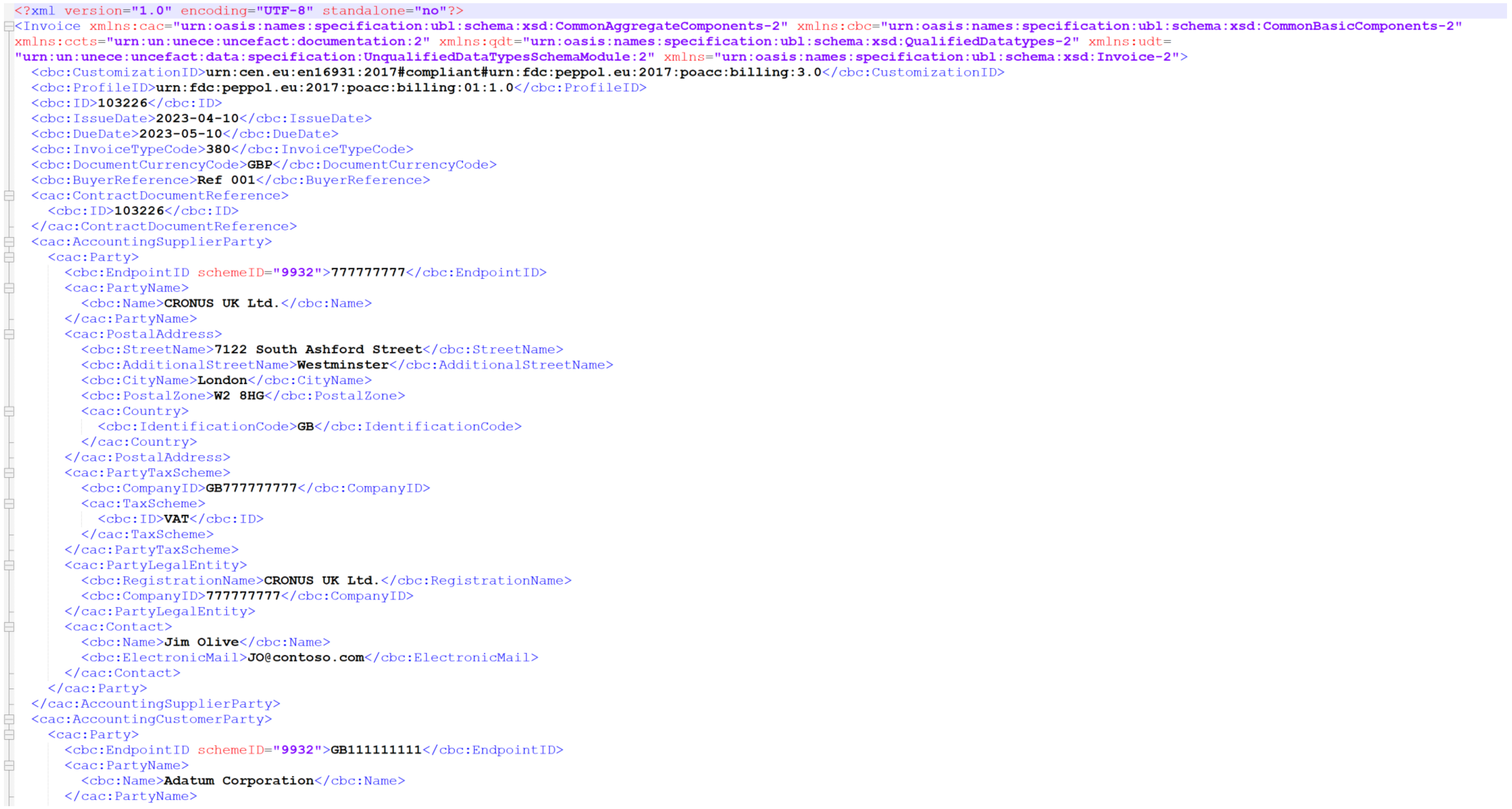
Sending - Create

```
procedure Create(EDocumentService: Record "E-Document Service"; var EDocument: Record "E-Document"; var SourceDocumentHeader: RecordRef; var SourceDocumentLines:
RecordRef; var TempBlob: Codeunit "Temp Blob")
    var
        OutStr: OutStream;
    begin
        TempBlob.CreateOutStream(OutStr);

        case EDocument."Document Type" of
            EDocument."Document Type"::"Sales Invoice":
                GenerateInvoiceXMLFile(SourceDocumentHeader, OutStr);
            EDocument."Document Type"::"Sales Credit Memo":
                GenerateCrMemoXMLFile(SourceDocumentHeader, OutStr);
        end;
    end;
```



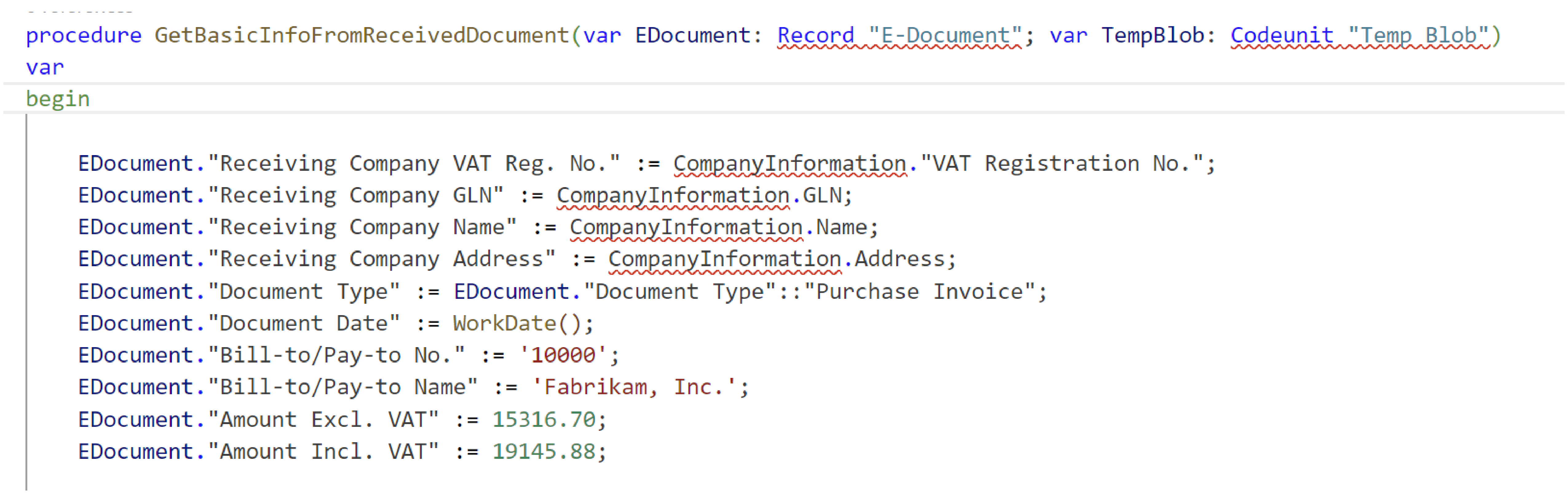
```
local procedure GenerateInvoiceXMLFile(VariantRec: Variant; var OutStr: OutStream)
var
    SalesInvoicePEPPOLBIS30: XMLport "Sales Invoice - PEPPOL BIS 3.0";
begin
    SalesInvoicePEPPOLBIS30.Initialize(VariantRec);
    SalesInvoicePEPPOLBIS30.SetDestination(OutStr);
    SalesInvoicePEPPOLBIS30.Export();
end;

2 references
local procedure GenerateCrMemoXMLFile(VariantRec: Variant; var OutStr: OutStream)
var
```

Sending - Create



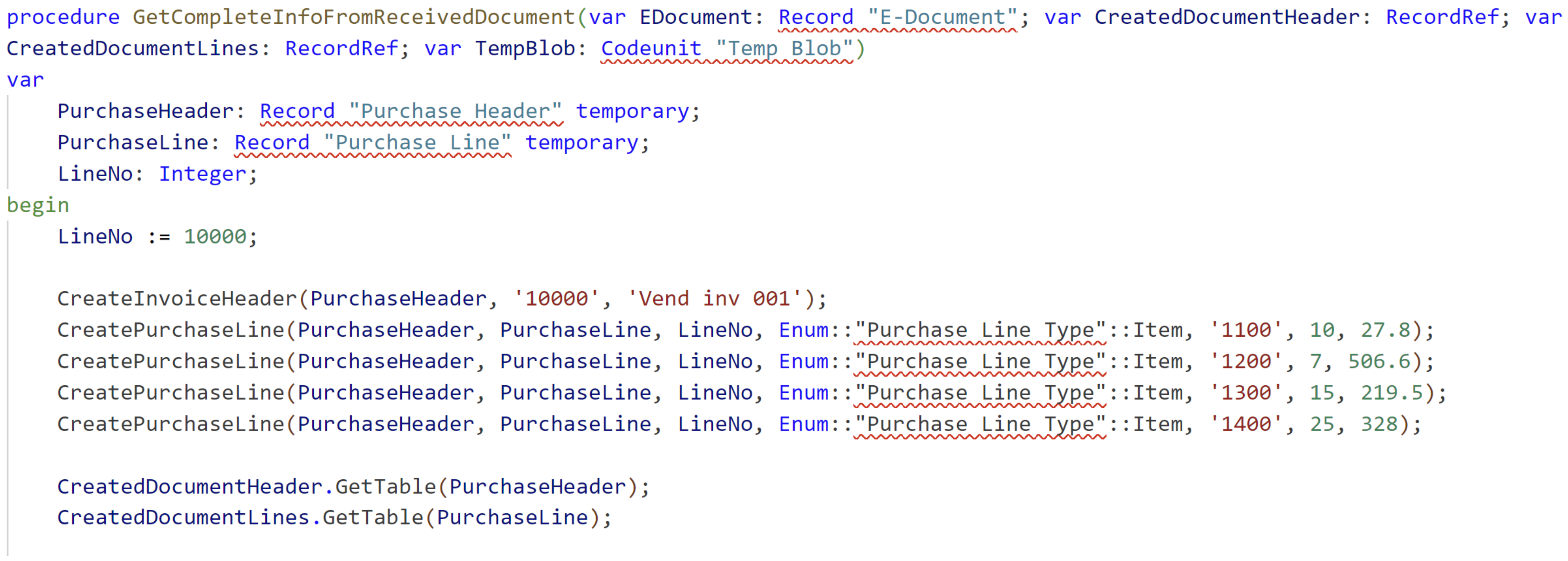
```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Invoice xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
xmlns:ccts="urn:un:unece:uncefact:documentation:2" xmlns:qdt="urn:oasis:names:specification:ubl:schema:xsd:QualifiedDatatypes-2" xmlns:udt=
"urn:un:unece:uncefact:data:specification:UnqualifiedDataTypesSchemaModule:2" xmlns="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2">
  <cbc:CustomizationID>urn:cen.eu:en16931:2017#compliant#urn:fdc:peppol.eu:2017:poacc:billing:3.0</cbc:CustomizationID>
  <cbc:ProfileID>urn:fdc:peppol.eu:2017:poacc:billing:01:1.0</cbc:ProfileID>
  <cbc:ID>103226</cbc:ID>
  <cbc:IssueDate>2023-04-10</cbc:IssueDate>
  <cbc:DueDate>2023-05-10</cbc:DueDate>
  <cbc:InvoiceTypeCode>380</cbc:InvoiceTypeCode>
  <cbc:DocumentCurrencyCode>GBP</cbc:DocumentCurrencyCode>
  <cbc:BuyerReference>Ref 001</cbc:BuyerReference>
  <cac:ContractDocumentReference>
    <cbc:ID>103226</cbc:ID>
  </cac:ContractDocumentReference>
  <cac:AccountingSupplierParty>
    <cac:Party>
      <cbc:EndpointID schemeID="9932">777777777</cbc:EndpointID>
      <cac:PartyName>
        <cbc:Name>CRONUS UK Ltd.</cbc:Name>
      </cac:PartyName>
      <cac:PostalAddress>
        <cbc:StreetName>7122 South Ashford Street</cbc:StreetName>
        <cbc:AdditionalStreetName>Westminster</cbc:AdditionalStreetName>
        <cbc:CityName>London</cbc:CityName>
        <cbc:PostalZone>W2 8HG</cbc:PostalZone>
        <cac:Country>
          <cbc:IdentificationCode>GB</cbc:IdentificationCode>
        </cac:Country>
      </cac:PostalAddress>
      <cac:PartyTaxScheme>
        <cbc:CompanyID>GB777777777</cbc:CompanyID>
        <cac:TaxScheme>
          <cbc:ID>VAT</cbc:ID>
        </cac:TaxScheme>
      </cac:PartyTaxScheme>
      <cac:PartyLegalEntity>
        <cbc:RegistrationName>CRONUS UK Ltd.</cbc:RegistrationName>
        <cbc:CompanyID>777777777</cbc:CompanyID>
      </cac:PartyLegalEntity>
      <cac:Contact>
        <cbc:Name>Jim Olive</cbc:Name>
        <cbc:ElectronicMail>JO@contoso.com</cbc:ElectronicMail>
      </cac:Contact>
    </cac:Party>
  </cac:AccountingSupplierParty>
  <cac:AccountingCustomerParty>
    <cac:Party>
      <cbc:EndpointID schemeID="9932">GB111111111</cbc:EndpointID>
      <cac:PartyName>
        <cbc:Name>Adatum Corporation</cbc:Name>
      </cac:PartyName>
```

Receiving - GetBasicInfoFromReceivedDocument

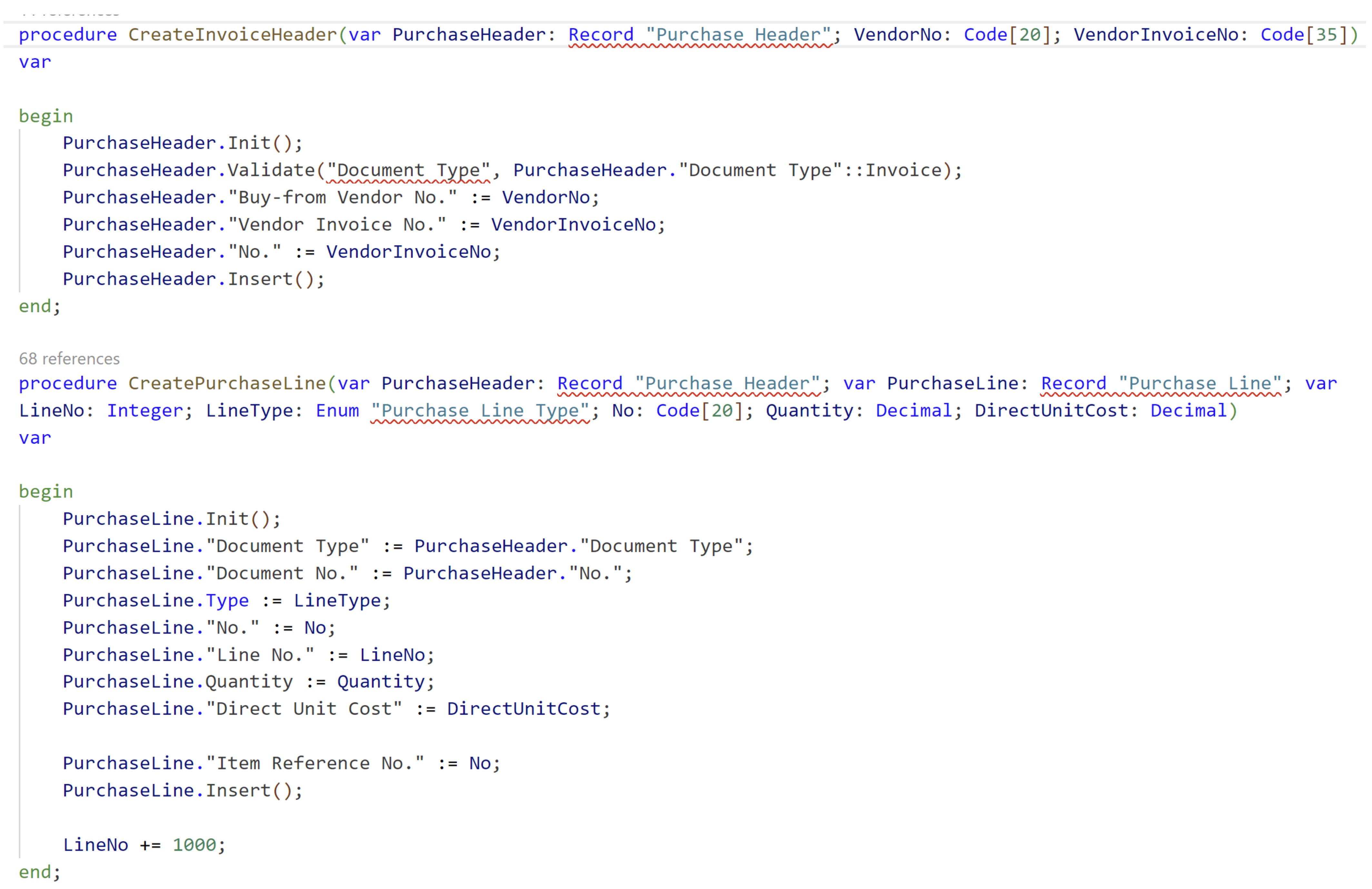


```
procedure GetBasicInfoFromReceivedDocument(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob")
var
begin

    EDocument."Receiving Company VAT Reg. No." := CompanyInformation."VAT Registration No.";
    EDocument."Receiving Company GLN" := CompanyInformation.GLN;
    EDocument."Receiving Company Name" := CompanyInformation.Name;
    EDocument."Receiving Company Address" := CompanyInformation.Address;
    EDocument."Document Type" := EDocument."Document Type"::"Purchase Invoice";
    EDocument."Document Date" := WorkDate();
    EDocument."Bill-to/Pay-to No." := '10000';
    EDocument."Bill-to/Pay-to Name" := 'Fabrikam, Inc.';
    EDocument."Amount Excl. VAT" := 15316.70;
    EDocument."Amount Incl. VAT" := 19145.88;
```

Receiving - GetCompleteInfoFromReceivedDocument



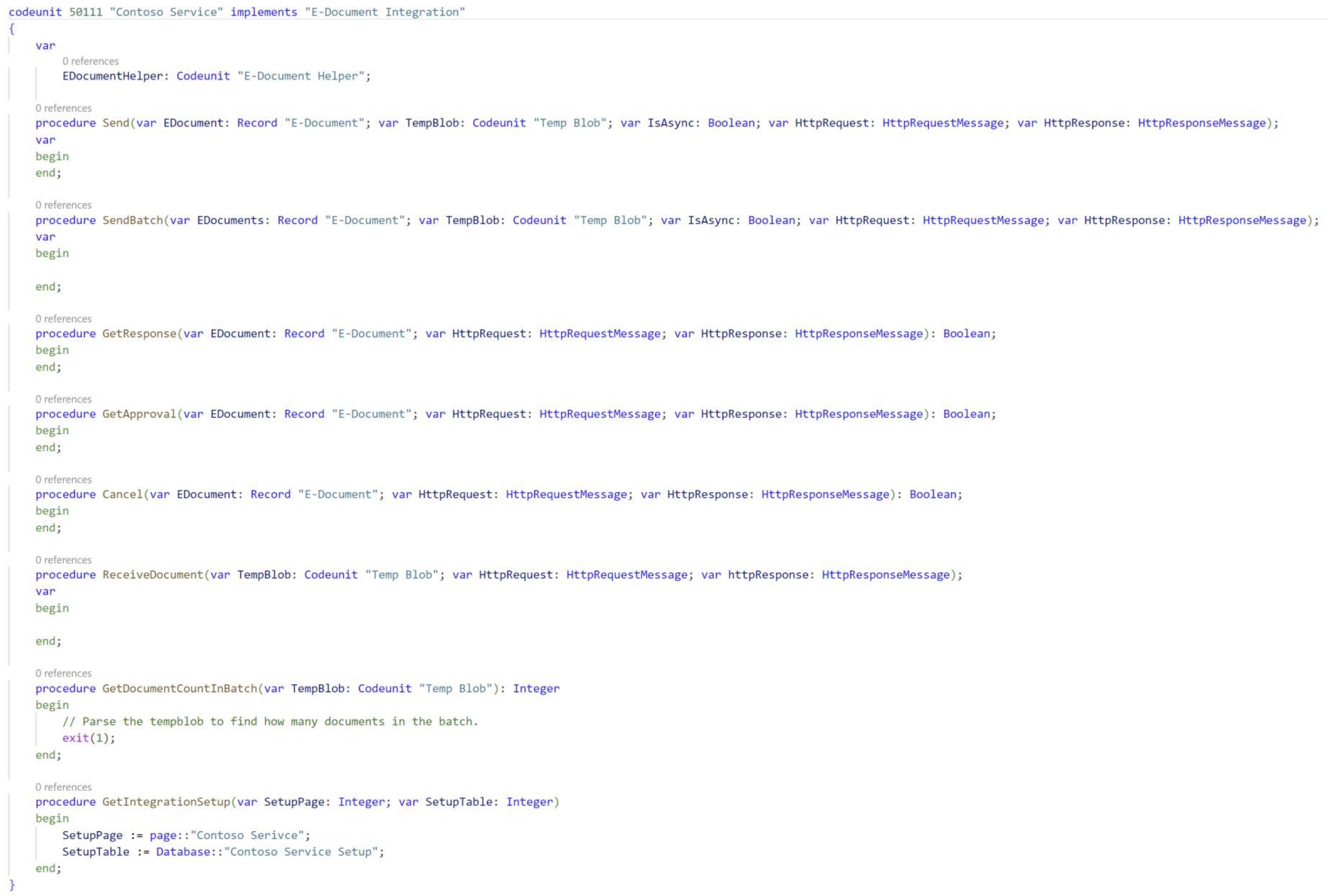
```
procedure GetCompleteInfoFromReceivedDocument(var EDocument: Record "E-Document"; var CreatedDocumentHeader: RecordRef; var
CreatedDocumentLines: RecordRef; var TempBlob: Codeunit "Temp Blob")
var
    PurchaseHeader: Record "Purchase Header" temporary;
    PurchaseLine: Record "Purchase Line" temporary;
    LineNo: Integer;
begin
    LineNo := 10000;

    CreateInvoiceHeader(PurchaseHeader, '10000', 'Vend inv 001');
    CreatePurchaseLine(PurchaseHeader, PurchaseLine, LineNo, Enum::"Purchase Line Type"::Item, '1100', 10, 27.8);
    CreatePurchaseLine(PurchaseHeader, PurchaseLine, LineNo, Enum::"Purchase Line Type"::Item, '1200', 7, 506.6);
    CreatePurchaseLine(PurchaseHeader, PurchaseLine, LineNo, Enum::"Purchase Line Type"::Item, '1300', 15, 219.5);
    CreatePurchaseLine(PurchaseHeader, PurchaseLine, LineNo, Enum::"Purchase Line Type"::Item, '1400', 25, 328);

    CreatedDocumentHeader.GetTable(PurchaseHeader);
    CreatedDocumentLines.GetTable(PurchaseLine);
```



```
procedure CreateInvoiceHeader(var PurchaseHeader: Record "Purchase Header"; VendorNo: Code[20]; VendorInvoiceNo: Code[35])
var

begin
    PurchaseHeader.Init();
    PurchaseHeader.Validate("Document Type", PurchaseHeader."Document Type"::Invoice);
    PurchaseHeader."Buy-from Vendor No." := VendorNo;
    PurchaseHeader."Vendor Invoice No." := VendorInvoiceNo;
    PurchaseHeader."No." := VendorInvoiceNo;
    PurchaseHeader.Insert();
end;

68 references
procedure CreatePurchaseLine(var PurchaseHeader: Record "Purchase Header"; var PurchaseLine: Record "Purchase Line"; var
LineNo: Integer; LineType: Enum "Purchase Line Type"; No: Code[20]; Quantity: Decimal; DirectUnitCost: Decimal)
var

begin
    PurchaseLine.Init();
    PurchaseLine."Document Type" := PurchaseHeader."Document Type";
    PurchaseLine."Document No." := PurchaseHeader."No.";
    PurchaseLine.Type := LineType;
    PurchaseLine."No." := No;
    PurchaseLine."Line No." := LineNo;
    PurchaseLine.Quantity := Quantity;
    PurchaseLine."Direct Unit Cost" := DirectUnitCost;

    PurchaseLine."Item Reference No." := No;
    PurchaseLine.Insert();

    LineNo += 1000;
end;
```

Integration interface

Integration interface



```
codeunit 50111 "Contoso Service" implements "E-Document Integration"
{
    var
        0 references
        EDocumentHelper: Codeunit "E-Document Helper";

    0 references
    procedure Send(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob"; var IsAsync: Boolean; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage);
    var
    begin
    end;

    0 references
    procedure SendBatch(var EDocuments: Record "E-Document"; var TempBlob: Codeunit "Temp Blob"; var IsAsync: Boolean; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage);
    var
    begin

    end;

    0 references
    procedure GetResponse(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
    begin
    end;

    0 references
    procedure GetApproval(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
    begin
    end;

    0 references
    procedure Cancel(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
    begin
    end;

    0 references
    procedure ReceiveDocument(var TempBlob: Codeunit "Temp Blob"; var HttpRequest: HttpRequestMessage; var httpResponse: HttpResponseMessage);
    var
    begin

    end;

    0 references
    procedure GetDocumentCountInBatch(var TempBlob: Codeunit "Temp Blob"): Integer
    begin
        // Parse the tempblob to find how many documents in the batch.
        exit(1);
    end;

    0 references
    procedure GetIntegrationSetup(var SetupPage: Integer; var SetupTable: Integer)
    begin
        SetupPage := page::"Contoso Serivce";
        SetupTable := Database::"Contoso Service Setup";
    end;
}
```

Sending - Send

```
    procedure Send(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob"; var IsAsync: Boolean; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage);
    var
        // Record that hold integration setup
        ExampleIntegration: Record "Example - Test Integration";
        HttpClient: HttpClient;
        Payload: Text;
    begin
        ExampleIntegration.Get();
        Payload := EDocumentHelper.TempBlobToTxt(TempBlob);

        // Manipulate the payload and set the headers if needed
        HttpRequest.Content.WriteFrom(Payload);
        HttpRequest.Method := 'POST';
        HttpRequest.SetRequestUri(ExampleIntegration."Sending Endpoint");

        HttpClient.Send(HttpRequest, HttpResponse);

        // Parse the response if needed.
    end;
```

Sending - SendBatch

```
    procedure SendBatch(var EDocuments: Record "E-Document"; var TempBlob: Codeunit "Temp Blob"; var IsAsync: Boolean; var HttpRequest: HttpRequestMessage; var
HttpResponse: HttpResponseMessage);
    var
        // Record that hold integration setup
        ExampleIntegration: Record "Example - Test Integration";
        HttpClient: HttpClient;
        Payload: Text;
    begin
        ExampleIntegration.Get();
        Payload := EDocumentHelper.TempBlobToTxt(TempBlob);

        // Manipulate the payload and set the headers if needed
        HttpRequest.Content.WriteFrom(Payload);
        HttpRequest.Method := 'POST';
        HttpRequest.SetRequestUri(ExampleIntegration."Sending Endpoint");

        HttpClient.Send(HttpRequest, HttpResponse);

        // Parse the response if needed.
    end;
```

Sending - GetResponse

```
procedure GetResponse(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
var
    // Record that hold integration setup
    ExampleIntegration: Record "Example - Test Integration";
    HttpClient: HttpClient;
begin
    ExampleIntegration.Get();

    // Manipulate the payload and set the headers if needed
    HttpRequest.Method := 'GET';
    HttpRequest.SetRequestUri(ExampleIntegration."Get Response Endpoint");

    HttpClient.Send(HttpRequest, HttpResponse);

    // Parse the response if needed.

    exit(HttpResponse.IsSuccessStatusCode);
end;
```

Sending - GetApproval

```
procedure GetApproval(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
var
    // Record that hold integration setup
    ExampleIntegration: Record "Example - Test Integration";
    HttpClient: HttpClient;
begin
    ExampleIntegration.Get();

    // Manipulate the payload and set the headers if needed
    HttpRequest.Method := 'GET';
    HttpRequest.SetRequestUri(ExampleIntegration."Get Response Endpoint");

    HttpClient.Send(HttpRequest, HttpResponse);

    // Parse the response if needed.

    exit(HttpResponse.IsSuccessStatusCode);
end;
```

Sending - Cancel

```
procedure Cancel(var EDocument: Record "E-Document"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage): Boolean;
var
    // Record that hold integration setup
    ExampleIntegration: Record "Example - Test Integration";
    HttpClient: HttpClient;
begin
    ExampleIntegration.Get();

    // Manipulate the payload and set the headers if needed
    HttpRequest.Method := 'Delete';
    HttpRequest.SetRequestUri(ExampleIntegration."Cancel Endpoint");

    HttpClient.Send(HttpRequest, HttpResponse);

    // Parse the response if needed.

    exit(HttpResponse.IsSuccessStatusCode);
end;
```

Receiving - GetDocumentCountInBatch

```
procedure GetDocumentCountInBatch(var TempBlob: Codeunit "Temp Blob"): Integer
begin
    // Parse the TempBlob to find how many documents in the batch.
    exit(1);
end;
```

Receiving - ReceiveDocument

```
procedure ReceiveDocument(var TempBlob: Codeunit "Temp Blob"; var HttpRequest: HttpRequestMessage; var HttpResponse: HttpResponseMessage);
 var
     // Record that hold integration setup
     ExampleIntegration: Record "Example - Test Integration";
     HttpClient: HttpClient;
     Result: Text;
 begin
     ExampleIntegration.Get();

     HttpRequest.Method := 'GET';
     HttpRequest.SetRequestUri(ExampleIntegration."Receiving Endpoint");

     HttpClient.Send(HttpRequest, HttpResponse);

     HttpResponse.Content.ReadAs(Result);
     WriteToTempBlob(TempBlob, Result);
 end;
```

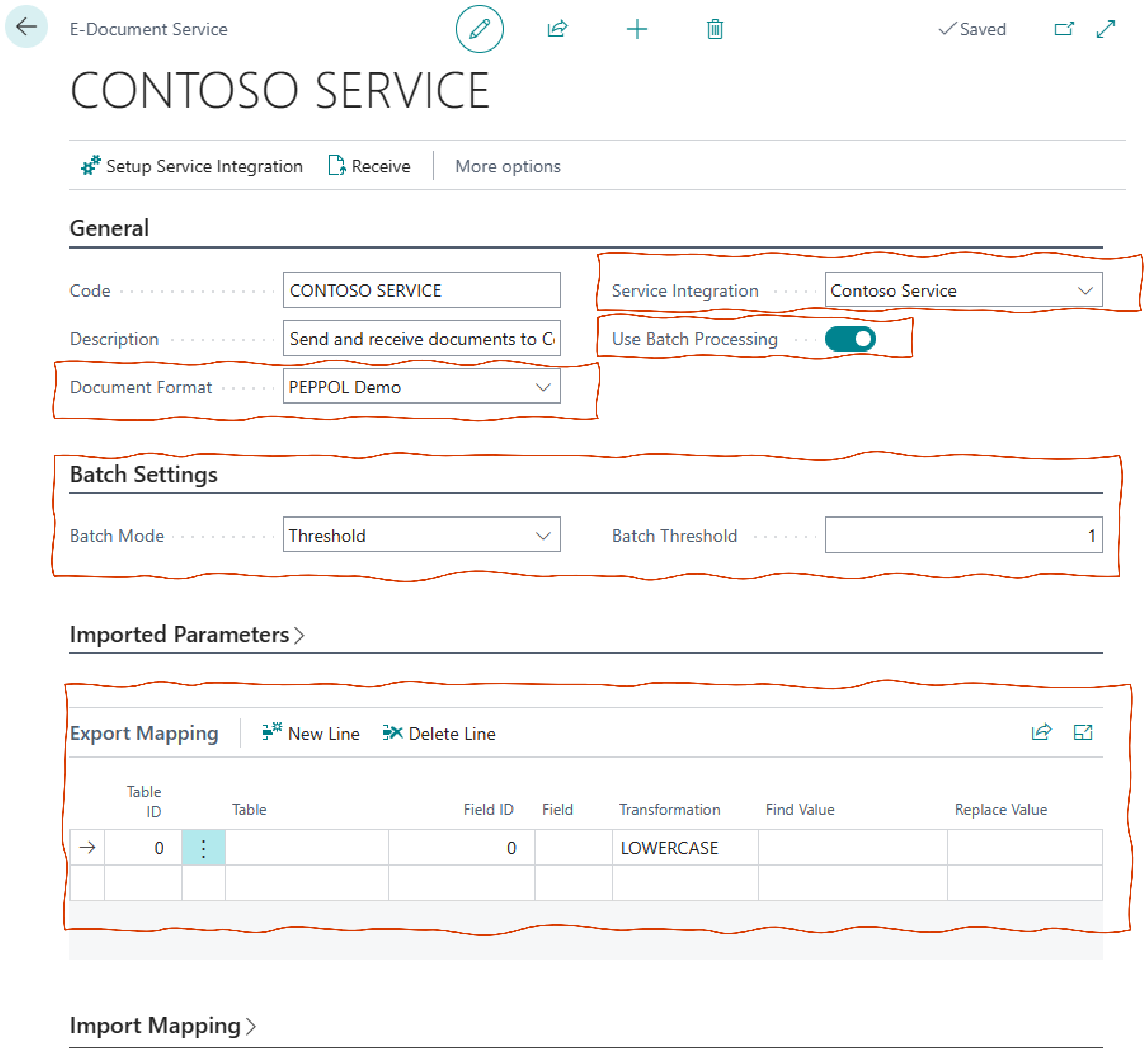
You can use our dummy test endpoint

- Send:
  - https://bcedocument.azurewebsites.net/post
- Receive (will spit out the latest invoice submitted)
  - https://bcedocument.azurewebsites.net/get

If you want something specific, I can quickly set something up for you, just ask.

Setup

Setup E-document Service



E-Document Service

CONTOSO SERVICE

Setup Service Integration · Receive · More options

General

Code	CONTOSO SERVICE
Description	Send and receive documents to C
Document Format	PEPPOL Demo
Service Integration	Contoso Service
Use Batch Processing	●

Batch Settings

Batch Mode	Threshold
Batch Threshold	1

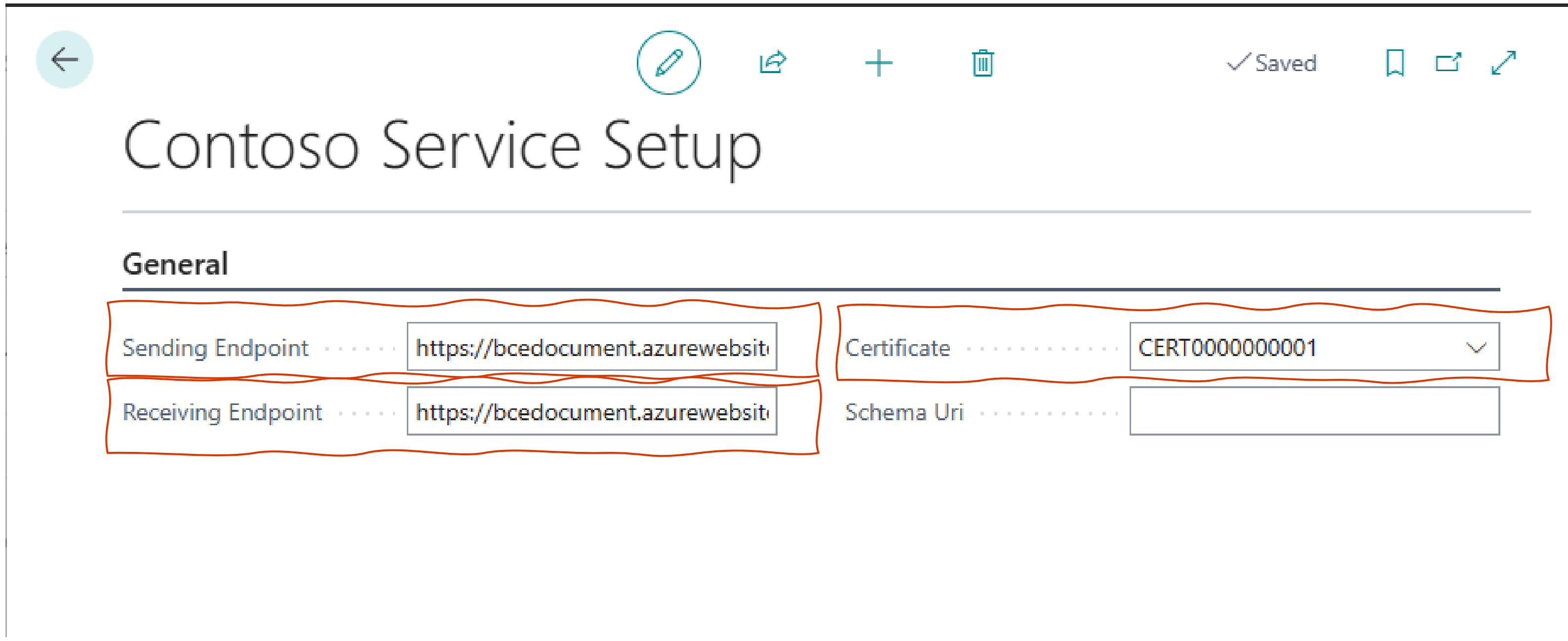
Imported Parameters

Export Mapping · New Line · Delete Line

Table ID	Table	Field ID	Field	Transformation	Find Value	Replace Value
0		0		LOWERCASE		

Import Mapping

Setup integration parameters

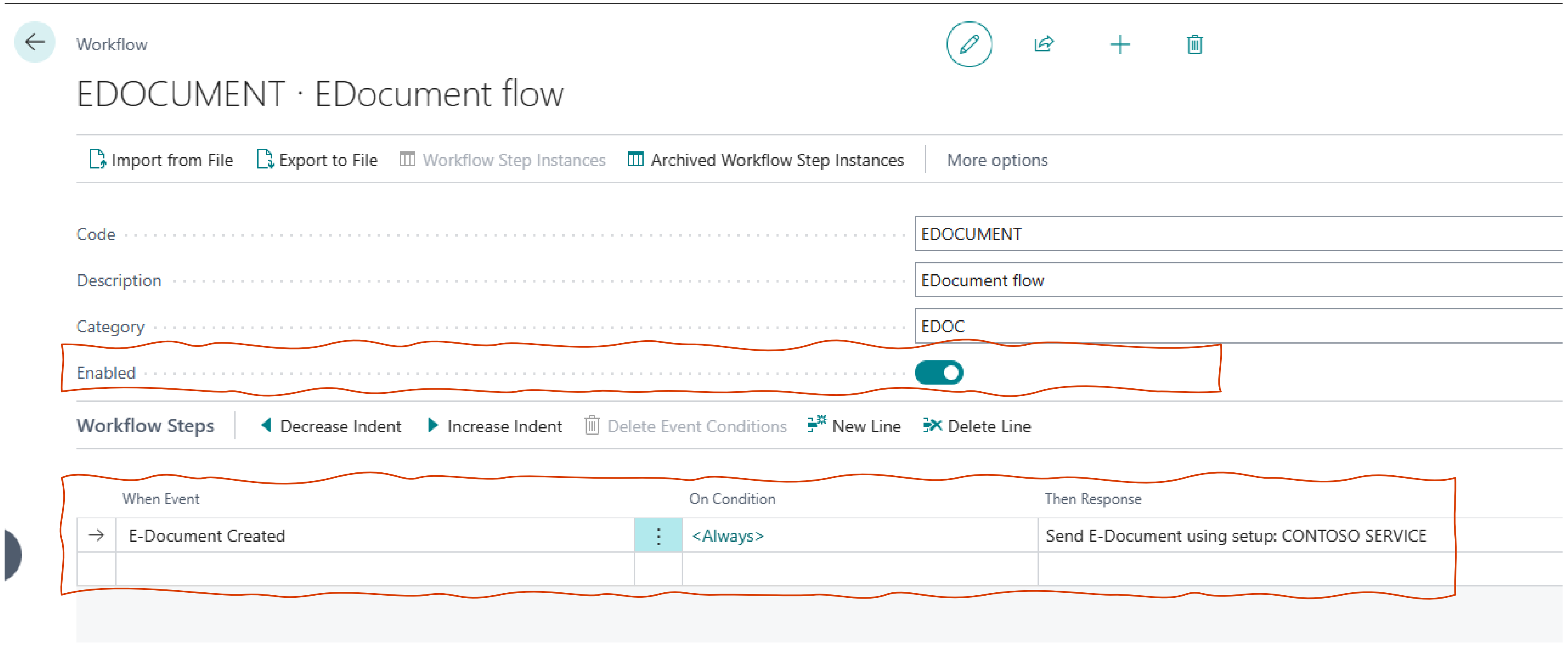


Contoso Service Setup

General

Sending Endpoint	https://bcedocument.azurewebsit	Certificate	CERT0000000001
Receiving Endpoint	https://bcedocument.azurewebsit	Schema Uri	

Setup Workflow


Workflow

EDOCUMENT · EDocument flow

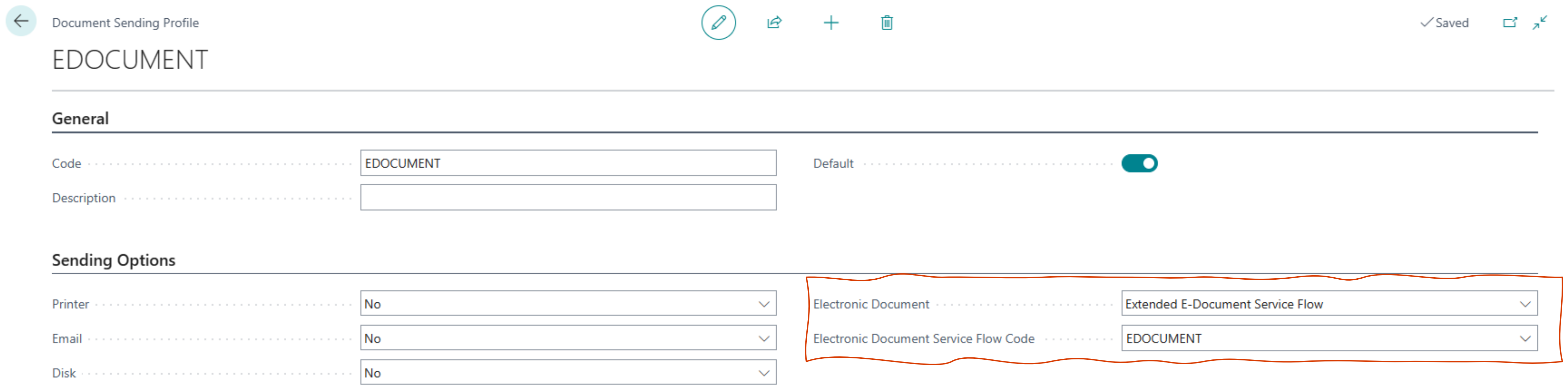
Import from File | Export to File | Workflow Step Instances | Archived Workflow Step Instances | More options

Code	EDOCUMENT
Description	EDocument flow
Category	EDOC
Enabled	●

Workflow Steps | Decrease Indent | Increase Indent | Delete Event Conditions | New Line | Delete Line

When Event	On Condition	Then Response
E-Document Created	<Always>	Send E-Document using setup: CONTOSO SERVICE

Setup document sending profile



Document Sending Profile

EDOCUMENT

Saved

General

Code	EDOCUMENT	Default	●
Description			

Sending Options

Printer	No	Electronic Document	Extended E-Document Service Flow
Email	No	Electronic Document Service Flow Code	EDOCUMENT
Disk	No		

Demo

Get started with setup dev environment

Setup Dev environment

You will need:

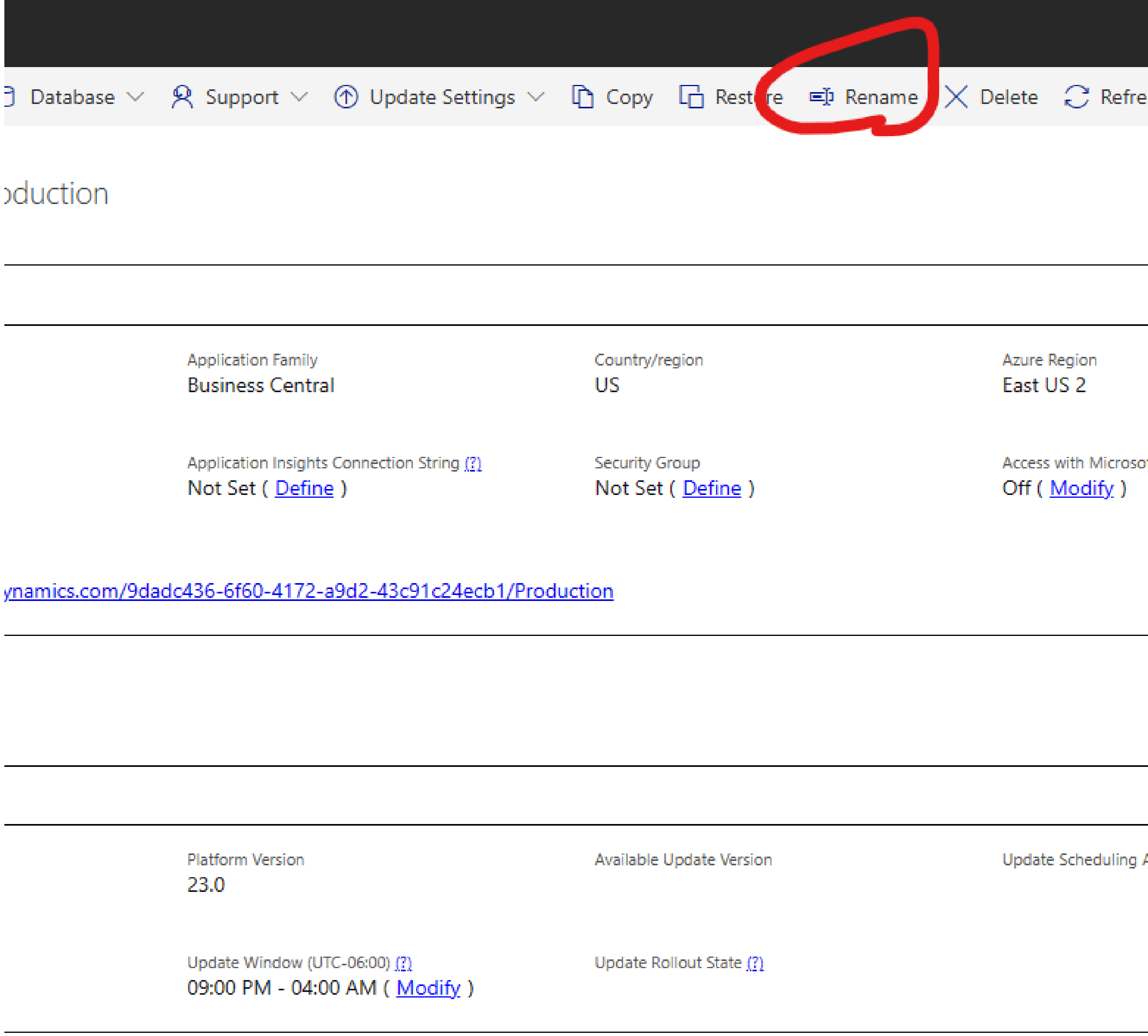
1. Laptop
2. VSCode
3. Updated AL compiler in VSCode

Setup Dev environment

- Login to admin center using:
  - Admin center: https://businesscentral.dynamics.com/9dadc436-6f60-4172-a9d2-43c91c24ecb1/admin

Setup Dev environment

- Assign environment to yourself by renaming it.



ɔduction

Application Family
Business Central

Country/region
US

Azure Region
East US 2

Application Insights Connection String (?)
Not Set (Define)

Security Group
Not Set (Define)

Access with Microso
Off (Modify)

ynamics.com/9dadc436-6f60-4172-a9d2-43c91c24ecb1/Production

Platform Version
23.0

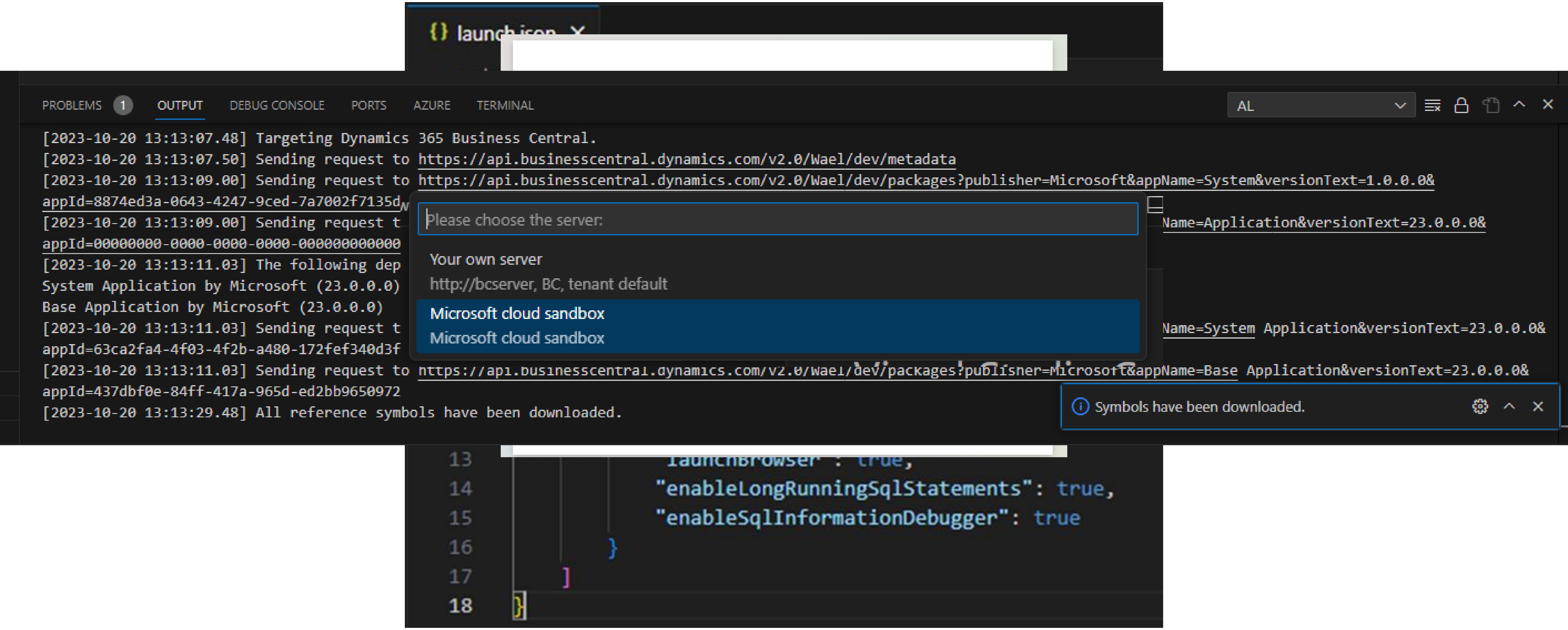
Available Update Version

Update Scheduling A

Update Window (UTC-06:00) (?)
09:00 PM - 04:00 AM (Modify)

Update Rollout State (?)

Debug sandbox



```
[2023-10-20 13:13:07.48] Targeting Dynamics 365 Business Central.
[2023-10-20 13:13:07.50] Sending request to https://api.businesscentral.dynamics.com/v2.0/Wael/dev/metadata
[2023-10-20 13:13:09.00] Sending request to https://api.businesscentral.dynamics.com/v2.0/Wael/dev/packages?publisher=Microsoft&appName=System&versionText=1.0.0.0&
appId=8874ed3a-0643-4247-9ced-7a7002f7135d
[2023-10-20 13:13:09.00] Sending request t                                                                     Name=Application&versionText=23.0.0.0&
appId=00000000-0000-0000-0000-000000000000
[2023-10-20 13:13:11.03] The following dep
System Application by Microsoft (23.0.0.0)
Base Application by Microsoft (23.0.0.0)
[2023-10-20 13:13:11.03] Sending request t                                                                     Name=System Application&versionText=23.0.0.0&
appId=63ca2fa4-4f03-4f2b-a480-172fef340d3f
[2023-10-20 13:13:11.03] Sending request to https://api.businesscentral.dynamics.com/v2.0/Wael/dev/packages?publisher=Microsoft&appName=Base Application&versionText=23.0.0.0&
appId=437dbf0e-84ff-417a-965d-ed2bb9650972
[2023-10-20 13:13:29.48] All reference symbols have been downloaded.
```

Please choose the server:

Your own server
http://bcserver, BC, tenant default

Microsoft cloud sandbox
Microsoft cloud sandbox

Symbols have been downloaded.

```
            "launchBrowser": true,
            "enableLongRunningSqlStatements": true,
            "enableSqlInformationDebugger": true
        }
    ]
}
```

Working time, find all materials here:

Skeleton app
guide
Presentation PDF

https://github.com/WaelAbuSeada/e-documents

You can find the open source of "E-Document Core" here:

ALAppExtensions/Apps/W1/EDocument/app at main · microsoft/ALAppExtensions (github.com)

Advanced implementation

Sending - Createbatch

```
procedure CreateBatch(EDocumentService: Record "E-Document Service"; var EDocuments: Record "E-Document"; var SourceDocumentHeaders: RecordRef; var
SourceDocumentsLines: RecordRef; var TempBlob: Codeunit "Temp Blob")
    var
        OutStr: OutStream;
    begin
        TempBlob.CreateOutStream(OutStr);
        if EDocuments.FindSet() then
            repeat
                AddDocuments(SourceDocumentHeaders,SourceDocumentsLines,OutStr)
            until EDocuments.Next() = 0;
    end;
```

Receiving - GetBasicInfoFromReceivedDocument

```
procedure GetBasicInfoFromReceivedDocument(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob")
var
    CompanyInformation: Record "Company Information";
begin
    if EDocument."Index In Batch" = 0 then
        ParseBasicInfo(EDocument, TempBlob)
    else
        ParseBatchOfDocuments(EDocument, TempBlob);
end;
```

Receiving - GetBasicInfoFromReceivedDocument

```
procedure ParseBasicInfo(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob")
var
    XmlDoc: XmlDocument;
    DocInstr: InStream;
    NamespaceManager: XmlNamespaceManager;
begin
    // Create an XML document from the blob
    TempBlob.CreateInStream(DocInstr);
    XmlDocument.ReadFrom(DocInstr, XmlDoc);

    // Parse the document to fill EDocument information
    EDocument."Bill-to/Pay-to No." := CopyStr(GetPEPPOLNode('//cac:InvoiceLine/cbc:ID', XmlDoc, NamespaceManager), 1, 20);
    EDocument."Bill-to/Pay-to Name" := CopyStr(GetPEPPOLNode('//cac:AccountingCustomerParty/cac:Party/cac:PartyName/cbc:Name', XmlDoc, NamespaceManager), 1, 20);
    Evaluate(EDocument."Document Date", GetPEPPOLNode('//cbc:IssueDate', XmlDoc, NamespaceManager));
    EDocument."Document Type" := EDocument."Document Type"::"Purchase Invoice";
end;

procedure ParseBatchOfDocuments(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob")
var
    XmlDoc: XmlDocument;
    DocInstr: InStream;
    NamespaceManager: XmlNamespaceManager;
begin
    // Create an XML document from the blob
    TempBlob.CreateInStream(DocInstr);
    XmlDocument.ReadFrom(DocInstr, XmlDoc);

   // Based on the batching technique, use the index in EDocument."Index In Batch" to access the document header and lines
   ParseDocumentHeaderAndLines(EDocument,XmlDoc,EDocument."Index In Batch");
end;
```

Receiving - GetCompleteInfoFromReceivedDocument

```
procedure GetCompleteInfoFromReceivedDocument(var EDocument: Record "E-Document"; var CreatedDocumentHeader: RecordRef; var
CreatedDocumentLines: RecordRef; var TempBlob: Codeunit "Temp Blob")
    begin
        if EDocument."Index In Batch" = 0 then
            ParseCompleteInfoAndCreatePurchaseDoc(EDocument, TempBlob)
        else
            ParseBatchDocumentCompleteInfoAndCreatePurchaseDoc(EDocument, TempBlob);
    end;
```

Sending – Send - async

```
    procedure Send(var EDocument: Record "E-Document"; var TempBlob: Codeunit "Temp Blob"; var IsAsync: Boolean; var HttpRequest: HttpRequestMessage; var HttpResponse:
HttpResponseMessage);
    var
        // Record that hold integration setup
        ExampleIntegration: Record "Example - Test Integration";
        HttpClient: HttpClient;
        Payload: Text;
    begin
        ExampleIntegration.Get();
        Payload := EDocumentHelper.TempBlobToTxt(TempBlob);

        // Manipulate the payload and set the headers if needed
        HttpRequest.Content.WriteFrom(Payload);
        HttpRequest.Method := 'POST';
        HttpRequest.SetRequestUri(ExampleIntegration."Sending Endpoint");

        HttpClient.Send(HttpRequest, HttpResponse);

        // Parse the response if needed.
    end;
```

Wizard

Show your work

Other resources, learn more!

Join the conversation
https://twitter.com/MSDYN365BC

Submit your ideas
aka.ms/BCIdeas

Have a question?
aka.ms/BCYammer

Looking for resources?
aka.ms/BCAll

Didn't watch the launch event?
aka.ms/BCLE

Other resources

Learn more!

Have a question?
aka.ms/BCYammer

Join the conversation
https://twitter.com/MSDYN365BC

Looking for resources?
aka.ms/BCAll

Submit your ideas
aka.ms/BCIdeas

Didn't watch the launch event?
aka.ms/BCLE

Don’t miss these sessions during the conference

Session title	Room	Date	Time
Localizations today and in the future	Tete d’Or 1&2	Nov. 2	2:00 pm
New demo tool	Thone 2	Nov. 3	10:30 am
Planned enhancements in E-documents (roundtable)	Gratte Ciel 3	Nov. 3	11:30 am
What’s new in VAT	Forum 2	Nov. 3	12:45 am

Thank you

MERCI TACK ขอบคุณครับ KIITOS MULŢUMESC DANKE

DANK U GRACIAS 감사합니다 شكراً DZIĘKUJĘ

OBRIGADO THANK YOU TEŞEKKÜRLER ευχαριστώ धन्यवाद

AČIŪ நன்றி

Merci beaucoup!

תודה 多謝晒

PALDIES ДЯКУЮ ĎAKUJEM متشكرم TERIMA KASIH TAK

HVALA DANKE DĚKUJI 谢谢 CÁM ƠN DANKON

GRAZIE KÖSZÖNÖM БЛАГОДАРЯ ありがとうございます