Extensibility Guide | PUBLIC

Enhance the Manage Sales Orders - Version 2 App to Include a Gift Card

THE BEST RUN

Content

1 Overview. 3

1.1 Business Scenario. 3

2 Preparation. 4

2.1 Prerequisites. 4

3 Implementation Steps. 6

3.1 Creating a Custom BO and Service for Storing Gift Card Details. 7

3.2 Creating an SAP Fiori App Using SAP Business Application Studio and Deploying It to SAP S/4HANA Cloud. 11

3.3 Creating an IAM App, a Business Catalog, and Assigning IAM Apps to Business Catalogs. 14

3.4 Extending Sales Order RAP BO for the Gift Card Scenario. 15

4 Provide Access to SAP Fiori App. 23

4.1 Creating a Business Role and Assigning It to a Business User. 23

5 Adapting the Manage Sales Orders - Version 2 App to the Display Amount and Currency Fields. 24

6 Scenario Execution. 25

1 Overview

This sample scenario lets you use SAP S/4HANA Cloud's developer extensibility features to enhance the sales order process so it includes a gift card while creating a sales order. A custom SAP Fiori app is created to maintain gift cards. These gift cards can be included in a sales order for a discount.

1.1 Business Scenario

> i Note
>
> This sample scenario is for learning purposes only. It is intended to give you an understanding of the various technical aspects related to extending SAP S/4HANA Cloud. The sample scenario may not always be available in a readily consumable state due to the continuous improvements being made in the underlying products or services. If this is the case, appropriate adaptations based on the latest documentation of the respective products or services are required.

The gift card scenario lets you maintain gift cards and include them in a sales order to obtain a discount. A custom SAP Fiori app and an RAP BO is built to store the gift card details. The sales order RAP BO is extended to store the gift card amount and currency details. The logic of the sales order RAP BO is also extended to enable selection of a gift card if the sales order's net price is higher than EUR 50. The value of the gift card is deducted from the sales order's net price by creating a price element.

2 Preparation

2.1 Prerequisites

Prerequisites	Details
SAP Business Technology Platform	You have an SAP BTP subaccount (multi-environment) with an entitlement for SAP Business Application Studio. For more information about SAP BTP accounts, refer to SAP Business Technology Platform. Note For non-productive/testing purposes, you can use an SAP BTP trial account. For more information, refer to https://developers.sap.com/tutorials/hcp-create-trial-account.html.
SAP Business Technology Platform	Ensure that the required authorizations for managing the subaccount are assigned. The role collection `Business_Application_Studio_Developer` is generated automatically once you subscribe to SAP Business Application Studio. Assign this role collection to your UI developers.
SAP S/4HANA Cloud	• The *Administrator* (SAP_BR_ADMINISTRATOR) business role is assigned to your user. Make sure that the Extensibility (SAP_CORE_BC_EXT) business catalog is also assigned to the role. • The *Extensibility Specialist* (SAP_BR_EXTENSIBILITY_SPEC) business role is assigned to your user. This is needed to access key user adaptation tools. • The *Developer* (SAP_BR_DEVELOPER) business role is assigned to your user. • The *Internal Sales Representative* (SAP_BR_INTERNAL_SALES_REP) business role is assigned to your user. This is needed to access the *Manage Sales Orders - Version 2* app.

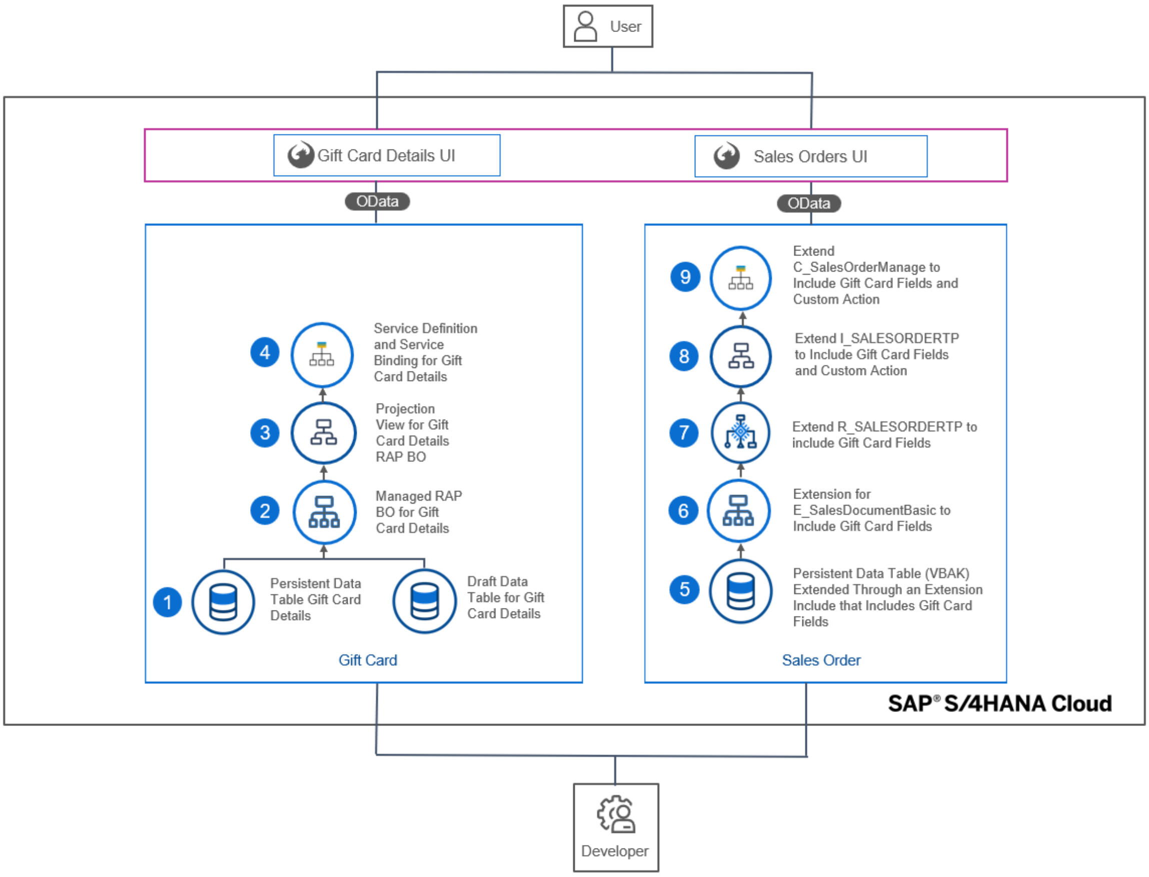
Prerequisites	Details
	ℹ Note Ensure that the business roles have unrestricted access for *Write*, *Read*, and *Value Help*.
SAP Business Technology Platform	**Configure Destinations in SAP BTP** The destination is configured and the API endpoint of the SAP S/4HANA Cloud system (...-api.s4hana...) is used in the URL field. Create a destination to connect to SAP Business Application Studio. For creating a destination, refer to Create a Destination to Connect to SAP Business Application Studio. Use the *Download Trust* button in the *Destinations* menu to download the subaccount certificate.
SAP S/4HANA Cloud	**Create a Communication System** 1. Log on to the SAP Fiori launchpad in the SAP S/4HANA Cloud system as an *Administrator*. 2. Choose the *Communication Systems* tile. 3. Choose *New* to create a new system. 4. Enter a system ID and a system name. 5. Choose *Create*. 6. In the *Technical Data* section, enter information about the system that you want to integrate. Make the required settings for your communication arrangement: • Set the system to *Inbound Only*. • Choose the *SAML Bearer Assertion Provider* option and upload the previously downloaded certificate. • Copy the CN value on the right into the *SAML Bearer Issuer* field manually. 7. Choose *Save*.

3 Implementation Steps

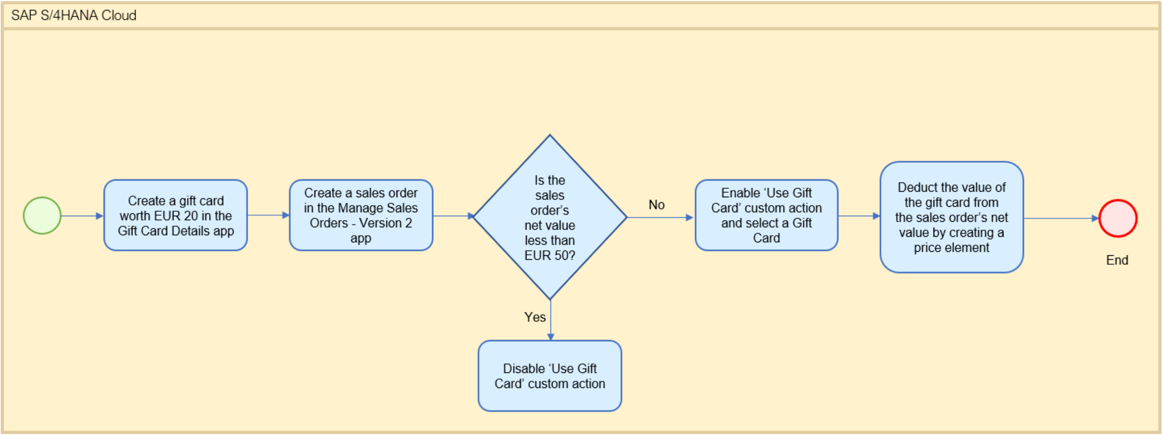
The implementation consists of these steps:

1. Creating a Custom BO and Service for Storing Gift Card Details [page 7]
2. Creating an SAP Fiori App Using SAP Business Application Studio and Deploying It to SAP S/4HANA Cloud [page 11]
3. Creating an IAM App, a Business Catalog, and Assigning IAM Apps to Business Catalogs [page 14]
4. Extending Sales Order RAP BO for the Gift Card Scenario [page 15]

The following graphic shows the basic architecture of the scenario:


User
Gift Card Details UI
Sales Orders UI
OData
OData
4
Service Definition and Service Binding for Gift Card Details
3
Projection View for Gift Card Details RAP BO
2
Managed RAP BO for Gift Card Details
1
Persistent Data Table Gift Card Details
Draft Data Table for Gift Card Details
Gift Card
9
Extend C_SalesOrderManage to Include Gift Card Fields and Custom Action
8
Extend I_SALESORDERTP to Include Gift Card Fields and Custom Action
7
Extend R_SALESORDERTP to include Gift Card Fields
6
Extension for E_SalesDocumentBasic to Include Gift Card Fields
5
Persistent Data Table (VBAK) Extended Through an Extension Include that Includes Gift Card Fields
Sales Order
SAP® S/4HANA Cloud
Developer

The following graphic shows the process flow of the scenario:


SAP S/4HANA Cloud
Create a gift card worth EUR 20 in the Gift Card Details app
Create a sales order in the Manage Sales Orders - Version 2 app
Is the sales order's net value less than EUR 50?
No
Enable 'Use Gift Card' custom action and select a Gift Card
Deduct the value of the gift card from the sales order's net value by creating a price element
End
Yes
Disable 'Use Gift Card' custom action

3.1 Creating a Custom BO and Service for Storing Gift Card Details

The gift card BO lets users store gift card details such as the gift card number, description, amount, and the gift card currency. This RAP business object supports transactional capabilities such as create, update, and delete.

Procedure

1. Log on to client 080 in your development system.
   Log on to the system to start development with valid user credentials.
2. Create an ABAP package.
   1. Choose *ZCUSTOM_DEVELOPMENT* > *New* > *ABAP Package* .
   2. Enter the following data:
      - *Name*: ZXE_S4
      - *Description*: Extensibility Explorer Scenario 4
   3. Mark it as a favorite.
   4. In the *Package Type* field, select *Development*.
   5. Choose *Next*.
   6. On the *Select Transport Request* screen, choose *Create a New Request* and enter a description.
   7. Choose *Finish*.
3. Create a data element for gift card number.
   1. Right-click the package ZXE_S4 and choose *New* > *Other ABAP Repository Object* .
   2. Search for *Data Element*, select it, and choose *Next*.
   3. Enter the following data:

        - *Name*: ZXE4_GIFTCARDNUMBER
        - *Description*: Gift Card Number
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. In the *Data Type Information* section, enter the following data:
        - *Category*: Predefined Type
        - *Data Type*: NUMC
        - *Length*: 10
    7. In the *Field Labels* section, enter the following data:
        - *Short*: Gift Card
        - *Medium*: Gift Card Number
        - *Long*: Gift Card Number
        - *Heading*: Gift Card Number
    8. Save and activate.
4. Create a data element for the gift card description.
    1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ▶.
    2. Search for *Data Element*, select it, and choose *Next*.
    3. Enter the following data:
        - *Name*: ZXE4_GIFTCARDDESC
        - *Description*: Gift Card Description
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. In the *Data Type Information* section, enter the following data:
        - *Category*: Predefined Type
        - *Data Type*: CHAR
        - *Length*: 40
    7. In the *Field Labels* section, enter the following data:
        - *Short*: Descr
        - *Medium*: Description
        - *Long*: Gift Card Description
        - *Heading*: Gift Card Description
    8. Save and activate.
5. Create a data element for the gift card amount.
    1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ▶.
    2. Search for *Data Element*, select it, and choose *Next*.
    3. Enter the following data:
        - *Name*: ZXE4_GIFTCARDAMT
        - *Description*: Gift Card Amount
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. In the *Data Type Information* section, enter the following data:
        - *Category*: Predefined Type
        - *Data Type*: CURR

        - *Length*: 15
        - *Decimals*: 2
    7. In the *Field Labels* section, enter the following data:
        - *Short*: Amount
        - *Medium*: Gift Card Amount
        - *Long*: Gift Card Amount
        - *Heading*: Gift Card Amount
    8. Save and activate.
6. Create a database table to store the gift card details.
    1. Right-click the package ZXE_S4 and choose *New* > *Other ABAP Repository Object*.
    2. Search for *Database Table*, select it, and choose *Next*.
    3. Enter the following data:
        - *Name*: ZXE4_GIFTCARD
        - *Description*: Gift Card Details
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. Replace the generated code with this ABAP code:

Sample Code

```
@EndUserText.label : 'Gift Card Details'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zxe4_giftcard {
  key client         : abap.clnt not null;
  key sap_uuid       : sysuuid_x16 not null;
  giftcardnumber     : zxe4_giftcardnumber not null;
  sap_description    : zxe4_giftcarddesc not null;
  @Semantics.amount.currencyCode : 'zxe4_giftcard.amount_c'
  amount_v           : zxe4_giftcardamt;
  amount_c           : abap.cuky;
  local_last_changed : abp_locinst_lastchange_tstmpl;
  last_changed       : abp_lastchange_tstmpl;
}
```

    7. Save and activate.
7. Generate an RAP BO using RAP Generator.
    1. Right-click the database table ZXE4_GIFTCARD and choose *Generate ABAP Repository Objects...*.
    2. In the *ABAP RESTful Application Programming Model* dropdown, choose *OData UI Service*.
    3. Choose *Next*.
    4. On the *Generate ABAP Repository Objects* screen, enter the following data:
        - *Package*: ZXE_S4
    5. Choose *Next*.
    6. In the *General* section of RAP Layers, enter the following data:
        - *Description*: Gift Card Details
    7. In the *Data Model* section under Business Object menu of RAP Layers, enter the following data:
        - *Data Definition Name*: ZXE4_I_GIFTCARD
        - *Alias Name*: GiftCard

   8. In the *Behavior* section under Business Object menu of RAP Layers, enter the following data:
      - *Implementation Class*: ZBP_XE4_I_GIFTCARD
      - *Draft Table Name*: ZXE4_GIFTCARD_D
   9. In the *Service Projection* section of RAP Layers, enter the following data:
      - *Name*: ZXE4_C_GIFTCARD
   10. In the *Service Definition* section under Business Service menu of RAP Layers, enter the following data:
      - *Name*: ZXE4_SD_GIFTCARD
   11. In the *Service Binding* section under Business Service menu of RAP Layers, enter the following data::
      - *Name*: ZXE4_UI_GIFTCARD_V4
      - *Binding Type*: OData V4 - UI
   12. Choose *Next*.
   13. On the *Preview Generator Output* screen, choose *Next*.
   14. Select a transport request and choose *Finish*.

8. Add additional UI annotations to the metadata extension of the generated projection view.
   1. Replace the generated code of metadata extension ZXE4_C_GIFTCARD with this ABAP code:

Sample Code

```
@Metadata.layer: #CORE
UI: { headerInfo: { typeName: 'Gift Card Details',
                     typeNamePlural: 'Gift Card Details',
                     title: { type: #STANDARD, value:
'Giftcardnumber' },
                     description: { type: #STANDARD, value:
'SapDescription' }}}
@Search.searchable: true
annotate view ZXE4_C_GIFTCARD with
{
  @UI.facet: [ {
    id: 'idIdentification',
    type: #IDENTIFICATION_REFERENCE,
    label: 'Gift Card',
    position: 10
  } ]
  @UI.hidden: true
  SapUUID;

  @UI.lineItem: [ {
    position: 10 ,
    importance: #MEDIUM,
    label: ''
  } ]
  @UI.identification: [ {
    position: 10 ,
    label: ''
  } ]
  @UI.selectionField: [{
    position: 10
    }]
  @Search.defaultSearchElement: true
  @Consumption.valueHelpDefinition: [ {entity: {name:
'ZXE4_GIFTCARDVH', element: 'Giftcardnumber' }} ]
  Giftcardnumber;

  @UI.lineItem: [ {
    position: 20 ,
    importance: #MEDIUM,
    label: ''
  } ]
  @UI.identification: [ {
    position: 20 ,
    label: ''
  } ]
  @Search.defaultSearchElement: true
  SapDescription;

  @UI.lineItem: [ {
    position: 30 ,
    importance: #MEDIUM,
    label: ''
  } ]
  @UI.identification: [ {
    position: 30 ,
    label: ''
  } ]
  @Search.defaultSearchElement: true
  AmountV;

  @Consumption.valueHelpDefinition: [ {entity: {name:
'I_CurrencyStdVH', element: 'Currency' }} ]
  AmountC;

  @UI.hidden: true
  LocalLastChanged;
}
```

9. Publish the generated service binding.
    1. In the *Services* section of the service binding, click service name ZXE4_SD_GIFTCARD and choose *Publish*.
    2. Go to the *Service Version Details* section.
    3. Under *Entity Set and Association*, choose *GiftCard*.
    4. Choose *Preview*.

3.2 Creating an SAP Fiori App Using SAP Business Application Studio and Deploying It to SAP S/4HANA Cloud

This topic describes how to create an application based on custom SAP Fiori Elements list report in SAP Business Application Studio and deploying it to an SAP S/4HANA Cloud system.

> i Note
>
> Prerequisite:
>
> - Ensure that a **Dev Space** of the *SAP Fiori* type is created in the **Dev Space Manager** of SAP Business Application Studio.

Deployment configuration

To deploy an app in SAP S/4HANA Cloud, you need to use the deployment configuration. The target and destination are prefilled with the destination chosen earlier in the wizard. The package and the transport are self-explanatory.

The following objects are created after deployment:

- BSP application (WAPA type)
- SAP Fiori launchpad descriptor item (UIAD type)

SAP Fiori launchpad configuration

You need to configure the SAP Fiori launchpad so that you can display the app as a tile on the SAP S/4HANA Cloud SAP Fiori launchpad. The semantic object and action must be a unique combination. These are used for intent-based navigation on the launchpad.

The title and subtitle determine the texts displayed on the launchpad tile. Note that this information is encoded in the generated SAP Fiori launchpad descriptor item object (UIAD type).

Once you finish using the wizard and the project is generated, the *Application Info* view opens automatically. This view offers options to preview and deploy your app. You can also open the *Application Info* view by choosing ▶ *View* > *Find Command* > *Fiori: Open Application Info* ◀.

To configure the SAP Fiori launchpad:

1. Log on to SAP Business Application Studio using your SAP BTP subaccount.
2. Create a new folder called `zxe4_giftcard` in the projects folder.
3. On the *Get Started* page, choose *Start from Template*.
   If the *Get Started* page is not visible, enter the following command in the *Search* bar:

   > Sample Code
   >
   > ```
   > >Help: Get Started
   > ```

4. On the *Select Template and Target Location* screen, choose *SAP Fiori Application* and then choose *Start*.
5. On the *Template Selection* screen, choose *List Report Page*.
6. Choose *Next*.
7. On the *Data Source and Service Selection* screen, from the *Data Source* menu, choose *Connect to a System*.
8. From the *System* menu, choose the destination created for the SAP S/4HANA Cloud system.
9. Enter the service's user name and password.
10. From the *Service* menu, choose *ZXE4_UI_GIFTCARD_V4*.

    > **i Note**
    >
    > If you get an error message, enter the following command:.
    >
    > > Sample Code
    > >
    > > ```
    > > CF: Login to Cloud Foundry
    > > ```

11. Choose *Next*.
12. On the *Entity Selection* screen, from the *Main Entity* menu, choose *GiftCard*.
13. On the *Entity Selection* screen, choose *Yes* to automatically add table columns to the list page and a section to the object page, if not available.
14. Choose *Next*.
15. On the *Project Attributes* screen, enter the following data:
    - *Module Name*: zxe4_giftcard
    - *Application Title*: Gift Card Details
    - *Application Namespace*: zxe4_giftcard
    - *Description*: Application to maintain Gift Card details
    - *Project Folder Path*:/home/user/projects/zxe4_giftcard
    - *Add Deployment Configuration*: Yes
    - *Add FLP Configuration*: Yes
    - *Configure Odvanced Options*: Yes
    - *UI5 Theme*: Morning Horizon
    - *Add Eslint Configuration to the Project*: No
    - *Add Code Assist Libraries to Your Project*: No
    - *Enable TypeScript*: No
16. Choose *Next*.
17. On the *Deployment Configuration* screen, enter the following data:
    - *SAPUI5 ABAP Repository*: ZXE4_GIFTCARD
    - *Deployment Description*: Gift Card Details
    - *Transport Request*: <Enter your transport request>
    - *Package*: ZXE_S4
18. Choose *Next*.
19. On the *Fiori Launchpad Configuration* screen, enter the following data:
    - *Semantic Object*: GiftCard
    - *Action*: maintain
    - *Title*: Gift Card Details
    - *Subtitle*: Maintain
20. Choose *Finish*.

Preview the Application

You can preview the application from the terminal using the following command:

Sample Code

```
npm run start
```

Deployment

You can start the deployment from *Application Info* or from the terminal with the following command:

Sample Code

```
npm run deploy
```

Once started, you can review the deployment configuration and confirm your settings in the terminal by entering **Y**. Once the terminal output stops and you see a success message, the process is complete. You can use Eclipse or ADT to check whether the following objects were generated in the package ZXE_S4:

- `ZXE4_GIFTCARD_UI5R` (the launchpad app descriptor item)
- `ZXE4_GIFTCARD` (BSP application)

3.3 Creating an IAM App, a Business Catalog, and Assigning IAM Apps to Business Catalogs

Creating an IAM app

1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ◀.
2. Search for *IAM App*, select it, and choose *Next*.
3. Enter the following data:
   - *Name*: ZXE4_GIFTCARD_UI
   - *Description*: IAM App for Gift Card Scenario
   - *Application Type*: EXT=External App
4. Choose *Next*.
5. Select a transport request and choose *Finish*.
6. In the *General* section of the *Overview* tab of the IAM app, enter the following data:
   - *Fiori Launchpad App Descr ID*: ZXE4_GIFTCARD_UI5R
7. Choose *Save* and then choose *Publish Locally*.

Creating a business catalog and assigning IAM apps

1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ◀.
2. Search for *Business Catalog*, select it, and choose *Next*.

3. Enter the following data:
    - *Name*: ZBC_XE4_GIFTCARD
    - *Description*: Business Catalog for Gift Card Scenario
4. Choose *Next*.
5. Select a transport request and choose *Finish*.
6. Choose the *Apps* tab.
7. In the *Apps* section of the business catalog, choose *Add...*.
8. Enter the following data:
    - *IAM App*: ZXE3_FITFOR_APJC_SAJC
9. Choose *Next*.
10. Select a transport request and choose *Finish*.
11. In the *Apps* section of the business catalog, choose *Add...*.
12. Enter the following data:
    - *IAM App*: ZXE4_GIFTCARD_UI_EXT
13. Choose *Next*.
14. Select a transport request and choose *Finish*.
15. In the ZBC_XE4_GIFTCARD business catalog, choose *Publish Locally*.

This business catalog can now be included in a business role that can then be assigned to a business user. Users with this business role can find your custom app when they use the SAP Fiori launchpad.

3.4 Extending Sales Order RAP BO for the Gift Card Scenario

1. Create a Value Help CDS view for gift cards.
    1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ◀.
    2. Search for *Data Definition*, select it, and choose *Next*.
    3. Enter the following data:
        - *Name*: ZXE4_GIFTCARDVH
        - *Description*: Gift Card Value Help
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. On the *Templates* screen choose *Define View Entity*.
    7. Choose *Finish*.
    8. Replace the generated code with the following ABAP code:

Sample Code

```
@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Gift Card Value Help'
@ObjectModel.usageType:{
  serviceQuality: #A,
  sizeCategory: #L,
  dataClass: #MASTER
}
@ObjectModel.dataCategory: #VALUE_HELP
@Search: {
  searchable: true
}
@ObjectModel.representativeKey: 'SapUUID'
@Consumption.valueHelpDefault.fetchValues:#AUTOMATICALLY_WHEN_DISPLAYED
@Consumption.ranked: true
define view entity ZXE4_GIFTCARDVH
  as select from ZXE4_I_GIFTCARD as GiftCard
{
      @UI.hidden:true
  key GiftCard.SapUUID,
      @ObjectModel.text.element: ['SapDescription']
      GiftCard.Giftcardnumber,
      @Search: {
            defaultSearchElement: true,
            ranking: #HIGH,
            fuzzinessThreshold: 0.8 }
      @Semantics.text: true
      GiftCard.SapDescription
}
```

    9. Save and activate.

2. Create a CDS Abstract entity for choosing a gift card.
    1. Right-click the package ZXE_S4 and choose ▶ *New* ▶ *Other ABAP Repository Object* ▶.
    2. Search for *Data Definition*, select it, and choose *Next*.
    3. Enter the following data:
        - *Name*: ZXE4_ASSIGNGIFTCARDTOSOP
        - *Description*: Sales Order Gift Card
    4. Choose *Next*.
    5. Select a transport request and choose *Finish*.
    6. On the *Templates* screen choose *Define Abstract Entity with Parameters*.
    7. Choose *Finish*.
    8. Replace the generated code with the following ABAP code:

Sample Code

```
@EndUserText.label: 'Sales Order Gift Card'
define abstract entity ZXE4_ASSIGNGIFTCARDTOSOP{
@Consumption.valueHelpDefinition: [{
  entity: {
    name: 'ZXE4_GIFTCARDVH',
    element: 'Giftcardnumber'
  }
}]
    @EndUserText.label: 'Gift Card'
    Giftcardnumber     : zxe4_giftcardnumber;
}
```

    9. Save and activate.

3. Extend the SDSALESDOC _INCL_EEW_PS extension structure to include the fields for the gift card amount and currency.
    1. In the ADT menu bar, choose *Open ABAP Development Object*. In the Windows Operating System, you can access it with the keyboard shortcut `Control` + `Shift` + `A`.
    2. Enter SDSALESDOC_INCL_EEW_PS in the search bar.

3. Select SDSALESDOC_INCL_EEW_PS (Structure) and choose *OK*.
4. In the *Project Explorer* view, choose *Link with Editor*, so that the object is visible in the Project Explorer.
5. In the *Project Explorer* view, right-click the SDSALESDOC_INCL_EEW_PS structure and choose *Append Structure*.
   This 'Extension Include' allows us to extend the persistent table (VBAK) with additional fields.
6. Enter the following data:
   - *Package*: ZXE_S4
   - *Name*: ZXE4_SALESORDER_APPEND
   - *Description*: Sales Order Extension for Gift Card Fields
7. Choose *Next*.
8. Select a transport request and choose *Finish*.
9. Replace the generated code with the following ABAP code:

   Sample Code

   ```
   @EndUserText.label : 'Sales Order Extension for Gift Card Fields'
   @AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
   extend type sdsalesdoc_incl_eew_ps with zxe4_salesorder_append {
     @Semantics.amount.currencyCode :
   'zxe4_salesorder_append.zz_giftcardcurrency_sdh'
     zz_giftcardamount_sdh   : zxe4_giftcardamt;
     zz_giftcardcurrency_sdh : abap.cuky;
   }
   ```
10. Save and activate.

4. Extend E_SalesDocumentBasic to include the fields for the gift card amount and the currency.
   `E_SalesDocumentBasic` is the basic interface CDS view for the VBAK (Sales Order header) table.
   1. Right-click the package ZXE_S4 and choose *New* > *Other ABAP Repository Object*.
   2. Search for *Data Definition*, select it, and choose *Next*.
   3. Enter the following data:
      - *Name*: ZXE4_E_SALESDOCUMENT_EXT
      - *Description*: E_SALESDOCUMENTBASIC Extension
      - *Referenced Object*: E_SALESDOCUMENTBASIC
   4. Choose *Next*.
   5. Select a transport request and choose *Finish*.
   6. On the *Templates* screen choose *Extend View Entity*.
   7. Choose *Finish*.
   8. Replace the generated code with the following ABAP code:

      Sample Code

      ```
      extend view entity E_SALESDOCUMENTBASIC with association
      [0..1] to I_Currency as _zz_giftcardcurrency_sdh on
      $projection.zz_giftcardcurrency_sdh = _zz_giftcardcurrency_sdh.Currency
       {
          @Semantics.amount.currencyCode: 'zz_giftcardcurrency_sdh'
          Persistence.zz_giftcardamount_sdh as zz_giftcardamount_sdh,
          @ObjectModel.foreignKey.association: '_zz_giftcardcurrency_sdh'
          Persistence.zz_giftcardcurrency_sdh as zz_giftcardcurrency_sdh,

          _zz_giftcardcurrency_sdh
      }
      ```

    9. Save and activate.

5. Extend R_SalesOrderTP to include the fields for the gift card amount and currency. R_SalesOrderTP is the RAP BO for the sales order.
    1. In the ADT menu bar, choose *Open ABAP Development Object*. In the Windows Operating System, you can access it with the keyboard shortcut `Control` + `Shift` + `A`.
    2. Enter R_SalesOrderTP in the search bar.
    3. Select R_SalesOrderTP (Data Definition) and choose *OK*.
    4. In the *Project Explorer* view, choose *Link with Editor*, so that the object is visible in the Project Explorer.
    5. In the *Project Explorer* view, right-click the R_SALESORDERTP data definition and choose *New Data Definition*.
    6. Enter the following data:
        - *Package*: ZXE_S4
        - *Name*: ZXE4_R_SALESORDERTP_EXT
        - *Description*: R_SalesOrderTP Extension
    7. Choose *Next*.
    8. Select a transport request and choose *Next*.
    9. Choose *Extend View Entity* as the template and choose *Finish*.
    10. Replace the generated code with the following code:

        Sample Code

        ```
        extend view entity R_SALESORDERTP with {
          @Semantics.amount.currencyCode: 'zz_giftcardcurrency_sdh'
            _Extension.zz_giftcardamount_sdh as zz_giftcardamount_sdh,
            _Extension.zz_giftcardcurrency_sdh as zz_giftcardcurrency_sdh
        }
        ```

    11. Save and activate the extension.

6. Extend CDS I_SalesOrderTP to include the fields for the gift card amount and the currency. I_SalesOrderTP is the interface for the Sales Order RAP BO.
    1. In the ADT menu bar, choose *Open ABAP Development Object*. In the Windows Operating System, you can access it using the keyboard shortcut `Control` + `Shift` + `A`.
    2. Enter I_SalesOrderTP in the search bar.
    3. Select I_SalesOrderTP (Data Definition) and choose *OK*.
    4. In the *Project Explorer* view, choose *Link with Editor*, so that the object is visible in the Project Explorer.
    5. In the *Project Explorer* view, right-click the I_SALESORDERTP data definition and choose *New Data Definition*.
    6. Enter the following data:
        - *Package*: ZXE_S4
        - *Name*: ZXE4_I_SALESORDERTP_EXT
        - *Description*: I_SalesOrderTP Extension
    7. Choose *Next*.
    8. Select a transport request and choose *Next*.
    9. Choose *Extend View Entity* as the template and choose *Finish*.
    10. Replace the generated code with the following code:

        Sample Code

        ```
        extend view entity I_SALESORDERTP with {
        ```

```
  @Semantics.amount.currencyCode: 'zz_giftcardcurrency_sdh'
    SalesOrder.zz_giftcardamount_sdh as zz_giftcardamount_sdh,
    SalesOrder.zz_giftcardcurrency_sdh as zz_giftcardcurrency_sdh
}
```

   11. Save and activate the extension.

7. Extend the behavior definition of R_SalesOrderTP to include the gift card logic.
   1. In the ADT menu bar, choose *Open ABAP Development Object*. In the Windows Operating System, you can access it using the keyboard shortcut `Control` + `Shift` + `A`.
   2. Enter R_SalesOrderTP in the search bar.
   3. Select R_SalesOrderTP (Behavior Definition) and choose *OK*.
   4. In the *Project Explorer* view, choose *Link with Editor*, so that the object is visible in the Project Explorer.
   5. In the *Project Explorer* view, right-click the R_SALESORDERTP behavior definition and choose *New Behavior Extension*.
   6. Enter the following data:
      - *Package*: ZXE_S4
      - *Name*: ZXE4_GIFTCARD_EXT
      - *Description*: I_SalesOrderTP Extension
      - *BO Interface*: I_SALESORDERTP
   7. Choose *Next*.
   8. Select a transport request and choose *Next*.
   9. Save and activate.
   10. In line 2 of the behavior extension, click zbp_xe4_giftcard_ext and choose `Ctr` + `1`.
   11. In the *Quick Assist Proposal* dialog box, choose *Create Behavior Implementation Class* (zbp_xe4_giftcard_ext).
   12. Enter *Behavior Implementation for ZXE4_GIFTCARD_EXT* as the description.
   13. Choose *Next*.
   14. Select a transport request and choose *Finish*.
   15. Save and activate the generated class.
   16. Inside your behavior definition ZXE4_GIFTCARD_EXT, replace the generated code with the following code:

Sample Code

```
extension using interface i_salesordertp
implementation in class zbp_xe4_giftcard_ext unique;
extend behavior for SalesOrder
{
  action ( authorization : update , features : instance )
zz_use_gift_card
    parameter ZXE4_ASSIGNGIFTCARDTOSOP result [0..1] $self;
  field(readonly) zz_giftcardamount_sdh, zz_giftcardcurrency_sdh;
  side effects {
    action zz_use_gift_card affects entity _Item, entity
_PricingElement;
  }
}
```

8. Provide the implementation logic for the Sales Order behavior definition extension.
   1. Open the ZBP_XE4_GIFTCARD_EXT class.
   2. On the *Local Types* tab, replace the existing code with the following code:

Sample Code

```
CLASS lhc_salesorder DEFINITION INHERITING FROM
cl_abap_behavior_handler.
  PRIVATE SECTION.
    METHODS get_instance_features FOR INSTANCE FEATURES
      IMPORTING keys REQUEST requested_features FOR SalesOrder RESULT
result.
    METHODS zz_use_gift_card FOR MODIFY
      IMPORTING it_keys FOR ACTION SalesOrder~zz_use_gift_card RESULT
result.
ENDCLASS.
CLASS lhc_salesorder IMPLEMENTATION.
  METHOD get_instance_features.
    READ ENTITIES OF I_SalesOrderTP IN LOCAL MODE
      ENTITY SalesOrder
      FIELDS ( TotalNetAmount )
        WITH CORRESPONDING #( keys )
          RESULT DATA(lt_result_salesorder).
    result = VALUE #( FOR ls_salesorder IN lt_result_salesorder ( %tky
= ls_salesorder-%tky

%features-%action-zz_use_gift_card = COND #( WHEN ls_salesorder-
TotalNetAmount< '50'
  THEN if_abap_behv=>fc-o-disabled

                                       ELSE if_abap_behv=>fc-o-
enabled ) ) ).
  ENDMETHOD.
  METHOD zz_use_gift_card.
    SELECT giftcardnumber, amount_v, amount_c FROM zxe4_giftcard FOR
ALL ENTRIES IN @it_keys
        WHERE giftcardnumber = @it_keys-%param-Giftcardnumber
        INTO TABLE @DATA(lt_discount).
    "action implementation
    LOOP AT it_keys REFERENCE INTO DATA(lr_key).
      READ TABLE lt_discount INTO DATA(ls_discount) WITH KEY
giftcardnumber = lr_key->%param-Giftcardnumber.
      CHECK ls_discount IS NOT INITIAL.
      MODIFY ENTITIES OF i_salesordertp IN LOCAL MODE
        ENTITY salesorder
        UPDATE SET FIELDS WITH VALUE #(
        ( %tky                    = lr_key->%tky
          %data-zz_giftcardamount_sdh  = ls_discount-amount_v
          %data-zz_giftcardcurrency_sdh = ls_discount-amount_c )
         )
       CREATE BY \_pricingelement SET FIELDS WITH VALUE #(
        ( %tky    = lr_key->%tky
          %target =   VALUE #( (
            %cid                = 'CIDGIFTCARD'
            conditiontype       = 'DRV1'
            "conditiontype       = 'XXXX'
            conditionrateamount = ls_discount-amount_v * ( -1 )
            conditioncurrency   = ls_discount-amount_c ) ) ) )
        FAILED   DATA(ls_modify_failed)
        REPORTED DATA(ls_modify_reported).
      failed   = CORRESPONDING #( APPENDING BASE ( failed   )
ls_modify_failed   ).
      reported = CORRESPONDING #( APPENDING BASE ( reported )
ls_modify_reported ).
    ENDLOOP.
    READ ENTITIES OF i_salesordertp IN LOCAL MODE
      ENTITY salesorder
         ALL FIELDS WITH
         CORRESPONDING #( it_keys )
       RESULT DATA(lt_salesorder).
    result = VALUE #( FOR salesorder IN lt_salesorder ( %tky   =
salesorder-%tky
                                                        %param =
CORRESPONDING #( salesorder ) ) ).
  ENDMETHOD.
ENDCLASS.
CLASS lsc_R_SALESORDERTP DEFINITION INHERITING FROM
cl_abap_behavior_saver.
  PROTECTED SECTION.
    METHODS cleanup_finalize REDEFINITION.
ENDCLASS.
CLASS lsc_R_SALESORDERTP IMPLEMENTATION.
  METHOD cleanup_finalize.
  ENDMETHOD.
ENDCLASS.
```

   3. Save and activate.

9. Extend the Sales Order projection view to include the gift card fields.
   1. Right-click the package ZXE_S4 and choose ▸ *New* ▸ *Other ABAP Repository Object* ▸.
   2. Search for *Data Definition*, select it, and choose *Next*.
   3. Enter the following data:
      - *Name*: ZXE4_C_SALESORDERMANAGE_EXT
      - *Description*: Sales Order Projection View Extension
      - *Referenced Object*: C_SALESORDERMANAGE
   4. Choose *Next*.
   5. Select a transport request and choose *Finish*.
   6. On the *Templates* screen, choose *Extend View Entity*.
   7. Choose *Finish*.
   8. Replace the generated code with the following ABAP code:

Sample Code

```
@EndUserText.label: 'Sales Order Projection View Extension'
extend view C_SalesOrderManage with ZXE4_C_SALESORDERMANAGE_EXT
    association [0..1] to I_Currency as _zz_giftcardcurrency_sdh on
$projection.zz_giftcardcurrency_sdh = _zz_giftcardcurrency_sdh.Currency
 {
    @Semantics.amount.currencyCode: 'zz_giftcardcurrency_sdh'
    @UI: {
    fieldGroup:[{qualifier: 'OrderData', importance: #HIGH, type:
#FOR_ACTION, dataAction: 'zz_use_gift_card', label: 'Use Gift Card' }]}
@UI.lineItem: [{ position: 65, importance:#HIGH }]
    SalesOrder.zz_giftcardamount_sdh as zz_giftcardamount_sdh,
    @ObjectModel.foreignKey.association: '_zz_giftcardcurrency_sdh'
    SalesOrder.zz_giftcardcurrency_sdh as zz_giftcardcurrency_sdh,

    _zz_giftcardcurrency_sdh
}
```

   9. Save and activate.

10. Extend the behavior definition of C_SALESORDERMANAGE to include the gift card action.
    1. In the ADT menu bar, choose *Open ABAP Development Object*.
    2. Enter C_SALESORDERMANAGE in the search bar.
    3. Select C_SALESORDERMANAGE (Behavior Definition) and choose *OK*.
    4. In the *Project Explorer* view, choose *Link with Editor*, so that the object is visible in the Project Explorer.

5. In the *Project Explorer* view, right-click the C_SALESORDERMANAGE behavior definition and choose *New Behavior Extension*.
6. Enter the following data:
   - *Package*: ZXE_S4
   - *Name*: ZXE4_C_SALESORDERMANAGE_B_EXT
   - *Description*: Sales Order Extension for Gift Card
7. Choose *Next*.
8. Select a transport request and choose *Finish*.
9. Replace the generated code with the following ABAP code:

   Sample Code

   ```
   extension for projection;
   extend behavior for SalesOrder
   {
   use action zz_use_gift_card;
   }
   extend behavior for SalesOrderItem
   {
   }
   ```
10. Save and activate.

4 Provide Access to SAP Fiori App

4.1 Creating a Business Role and Assigning It to a Business User

1. Log on to SAP Fiori launchpad with the user that has the *Administrator* (SAP_BR_ADMINISTRATOR) role assigned in the customizing client (100) of the development system.
2. Search for the *Maintain Business Roles* app and select it.
3. In the *Maintain Business Roles* app, choose *New* and enter the following data::
   - *Business Role ID*: ZBR_XE4_GIFTCARD
   - *Business Role Description*: Gift Card Scenario
4. Choose *Create*.
5. In the *Assigned Business Catalogs* section, choose *Add*.
6. Search for the `ZBC_XE4_GIFTCARD` business catalog, select it, and choose *OK*.
7. Close the *Add Business Catalog* dialog box.
8. In the *Access Categories* section of the *General Role Details* tab:
   - Change the value of the *Write, Read, Value Help* field to *Unrestricted*.
   - Change the value of the *Read, Value Help* field to *Unrestricted*.
9. Go to the *Assigned Business Users* tab and choose *Add*.
10. Search for the business user that needs to be enriched with the *Gift Card Scenario* business role.
11. Select it, and choose *OK*.
12. Choose *Save*.
    The newly created *Gift Card Scenario* (ZBR_XE4_GIFTCARD) business role provides the user with access to the gift card maintenance app.

5 Adapting the Manage Sales Orders - Version 2 App to the Display Amount and Currency Fields

1. Log on to the customizing client (100) of the development system with the user that contains the business role based on these business role templates:
   - *Extensibility Specialist* (SAP_BR_EXTENSIBILITY_SPEC)
   - *Internal Sales Representative* (SAP_BR_INTERNAL_SALES_REP)

   > i Note
   >
   > If the SAP S/4HANA Cloud system is on the SAP S/4HANA Cloud 2302 release, then the Key-User Extensibility business catalogs are available under the *Administrator* (SAP_BR_ADMINISTRATOR) business role template.

2. Adapt the *Manage Sales Orders - Version 2* SAP Fiori UI at runtime using Key User adaptation.
   1. Go to SAP Fiori launchpad, search for and select *Manage Sales Orders - Version 2*.
   2. Choose ▶ *Create* ▶ *Create Sales Order* ◀.
   3. Enter the following data:
      - *Sales Order Type*: Standard Order (OR)
      - *Sales Organization*: Dom. Sales Org DE (1010)
      - *Distribution Channel*: DE distribution chan (10)
      - *Division*: Product Division 00 (00)
   4. Go to the *Actions* menu and choose *Adapt UI*.
      When you're in the adaptation mode of an object page, you can edit all UI elements (such as fields, groups of fields, or sections) that are highlighted when you hover over them or select them.
   5. On the *General Information* tab, in the *Basic Data* section, under *Order Data*, right-click the UI element container and choose *Add: Field* from the context menu.
   6. In the *Available Contents: Fields* menu, search for and select *Gift Card Amount*, and choose *OK*.
   7. On the page header, choose *Activate New Version* to activate the current draft or a selected version that is not currently active, so that it becomes a new version.
   8. In the *Activate New Version* menu, enter *Gift Card* as the version name and choose *Confirm*.
   9. On the page header, choose *Publish Version* to publish a version of a changed app or an app variant in a target system.

   The gift card amount is now visible in the *Order Data* section.

For more information about adapting SAP Fiori UIs at runtime, refer to Adapting SAP Fiori UIs at Runtime - Key User Adaptation.

6 Scenario Execution

Scenario Testing

1. Enter the gift card details.
    1. Log on to the the customizing client (100) of the development system with the user that has the *Gift Card Scenario* (ZBR_XE4_GIFTCARD) business role assigned to it.
    2. On SAP Fiori launchpad, search for *Gift Card Details* and select it.
    3. In the *Gift Card Details* app, choose *Create*.
    4. Enter the following data:
        - *Gift Card Number*: 1001
        - *Description*: EUR 20 Gift Card
        - *Gift Card Amount*: 20.00
        - *Currency*: EUR
    5. Choose *Create*.

    A gift card worth EUR 20.00 has been created.
2. Create a sales order and include a gift card.
    1. Log on to the the customizing client (100) of the development system with the user that has the role based on the (SAP_BR_INTERNAL_SALES_REP) business role template assigned to it.
    2. On SAP Fiori launchpad, search for *Manage Sales Orders* and select it.
    3. Choose ▶ *Create* ▶ *Create Sales Order* ◀.
    4. Enter the following data:
        - *Sales Order Type*: Standard Order (OR)
        - *Sales Organization*: Dom. Sales Org DE (1010)
        - *Distribution Channel*: DE distribution chan (10)
        - *Division*: Product Division 00 (00)
    5. Choose *Continue*.
    6. On the *General Information* tab, in the *Basic Data* section, enter the following data:
        - *Sold-to Party*: <Enter Customer ID>
    7. On the *Items* tab, enter the following data for the newly created line item:
        - *Product*: Trad.Good 11, PD, Reg.Trading (TG11)
        - *iRequested Quantity*: 10 Pc

        Since the net value of the sales order is EUR 175.50 and it exceeds EUR 50, the *Use Gift Card* action on the *General Information* tab is active.
    8. Choose *Use Gift Card* and enter the following data:
        - *Gift Card*: EUR 20 Gift Card (1001)
    9. Choose *Use Gift Card* .
    10. On the *Prices* tab of the sales order, go to *Price Elements*.

A price element of the *DRV1* condition type is created, with the gift card amount deducted from the net value of sales order.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon ↗ : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon ↗: You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

THE BEST RUN SAP