**Individual Project Final Submission Report (200 Points Possible)**

|  |  |
| --- | --- |
| Program Name: Shopping Cart and Product Management | |
| Prepared By: Rukon Khan | Date Prepared: 05/02/2020 |
|  |  |

All instructions (indicated in Italics) **should be deleted** prior to submitting the finalized version.

# Problem Statement

|  |  |
| --- | --- |
| Business Problem to be addressed | *Convert shopping cart lists into orders, create sales report and upload product list from excel for the shopping cart.* |
| Scope | *In addition to the existing function of the shopping cart system, our in-scope functionality is to convert shopping cart items into orders, create an overall sales report and upload functionality to upload product details from an external source. Individual customer sales report is out of scope and exporting product list from sap to external sources also out of scope.* |

# Development Documentation

|  |  |
| --- | --- |
| Pseudocode | 1. Shopping Cart Maintenance Program (Z178\_CART\_MAINT\_FINALPROJECT) 2. View Cart Items by joining z178\_cart, 178\_product, z178\_product\_txt tables 3. Add product into z178\_cart table. 4. Remove a product from z178\_cart table. 5. Clear cart by removing all items z z178\_cart table. 6. Convert shopping cart’s customer specific product and quantity list into orders. This functionality creates a unique GUID for each orders and adds customer, product, quantity and date details into z178\_lineitems table.   After converting shopping cart items into orders, program clears the shopping cart items for that specific customer.   1. Product Management (Z178\_PRODUCT\_MAINT\_FINALPROJEC) 2. This program takes product, description, price, currency and units as input. 3. Adds update product action inserts product information into z178\_product and z178\_product\_txt table. 4. Remove product action deletes a product from the z178\_product and z178\_product\_txt table. 5. Display Product action displays all the product and product description by combining z178\_product and z178\_product\_txt tables. 6. Upload product from excel action will require to select and excel file in .xls format. Within the excel file there should be product details in order of column Product Name, Price, Unit of Measurement and Product Description. Product name should be capitalized. This upload product functionality uses SAP BAPI function 'TEXT\_CONVERT\_XLS\_TO\_SAP' to upload excel file details into an internal table. Then we insert appropriate data from internal table into the z178\_product and z178\_product\_txt tables. There is also an option to choose if the excel file has header or not. 7. Sales Report (z178\_sales\_report) 8. This program creates a report by joining z178\_lineitems and z178\_product tables. 9. Reports provides details about customer, product, quantity ordered, price of the product, order date, total amount of cost per line item and finally a sum of total amount of cost per line items below the Total Amount Column. |
| Other | *Program and class lists.*Z178\_CART\_MAINT\_FINALPROJECT*Classes: ZCL178\_SHOPPING\_CART\_FINALPROJ and* z178\_CX\_SHOPPING\_CARTZ178\_PRODUCT\_MAINT\_FINALPROJECz178\_sales\_report |

# Functional Design Assumptions

|  |  |
| --- | --- |
| Assumptions |  |

# Identified Issues

|  |  |
| --- | --- |
| Issues | No open issues. |

# Additional information (OPTIONAL)

|  |  |
| --- | --- |
| Other relevant information | While testing the program, please use the .xls excel file (.xlsx format might cause issues) with a single worksheet and product names capitalized. Please run the product upload on virtual desktop. |

# 

# Figure: Z178\_CART\_MAINT\_FINALPROJECT program execution view

# 

# Figure: Z178\_PRODUCT\_MAINT\_FINALPROJECprogram execution view

# 

# Figure: Z178\_Sales\_Report program execution view

# CODES:

# Z178\_CART\_MAINT\_FINALPROJECT

# \*&---------------------------------------------------------------------\* \*& Report  Z178\_CART\_MAINT \*& \*&---------------------------------------------------------------------\* \*& \*& \*&---------------------------------------------------------------------\* REPORT Z178\_CART\_MAINT\_FINALPROJECT. tables: z178\_lineitems, z178\_cart. \*&---------------------------------------------------------------------\* \*& Report  Z185\_CART\_MAINT \*& \*&---------------------------------------------------------------------\* \*& \*& \*&---------------------------------------------------------------------\* " Selection screen SELECTION-SCREEN BEGIN OF BLOCK cart WITH FRAME TITLE text-001. PARAMETERS: p\_cust TYPE z178\_cart-customer OBLIGATORY, " Customer ID             p\_prod TYPE z178\_cart-product,             p\_qty TYPE z178\_cart-quantity.  " Product ID SELECTION-SCREEN END OF BLOCK cart. \* Parameters for action screen SELECTION-SCREEN BEGIN OF BLOCK action WITH FRAME TITLE text-002. PARAMETERS: p\_view  TYPE boolean RADIOBUTTON GROUP act,             p\_add   TYPE boolean RADIOBUTTON GROUP act,             p\_rem   TYPE boolean RADIOBUTTON GROUP act,             p\_clear TYPE boolean RADIOBUTTON GROUP act,             p\_order TYPE boolean RADIOBUTTON GROUP act. SELECTION-SCREEN END OF BLOCK action. Initialization. " Data declaration DATA: lo\_cart TYPE REF TO zcl178\_shopping\_cart\_finalproj,       lt\_cart TYPE lo\_cart->yt\_cart,       rp\_cart TYPE lo\_cart->AT\_cart. AT SELECTION-SCREEN.     TRY. " Cart object initialization CREATE OBJECT lo\_cart EXPORTING ip\_customer = p\_cust. CATCH z178\_CX\_SHOPPING\_CART. MESSAGE e000(38) WITH 'Customer not found'(006). ENDTRY. START-OF-SELECTION.  " Action if view cart items is true IF p\_view = abap\_true.   " Get customer products from the cart   lt\_cart = lo\_cart->get\_cart( ).   " Get output reference of the ALV table   cl\_salv\_table=>factory( IMPORTING r\_salv\_table = DATA(gr\_alv)                           CHANGING t\_table = lt\_cart ).   " Display the result as ALV table   gr\_alv->display( ). " Action if add production into the cart is true ELSEIF p\_add = abap\_true.   " Add product to the cart   lo\_cart->add\_product( ip\_product = p\_prod ip\_quantity = p\_qty ).   WRITE 'Product added'. " Acition if remove product condition is true ELSEIF p\_rem = abap\_true.   " Remove product from the cart   lo\_cart->remove\_product( p\_prod ).   WRITE 'Product removed from the cart'. " Action if clear products from the cart is true ELSEIF p\_clear = abap\_true.   " Clear product from the cart   lo\_cart->clear\_cart( ).   WRITE 'Cart cleared'. " Action if the place order from cart is true ELSEIF p\_order = abap\_true. " Data declaration data: itab type standard table of z178\_cart with header line, wa\_tab type z178\_lineitems, ld\_ev\_guid\_16   TYPE GUID\_16 . select \* into table itab from z178\_cart where customer = p\_cust. " Call BAPI function 'GUID\_CREATE' to create an unique order number CALL FUNCTION 'GUID\_CREATE'   IMPORTING     ev\_guid\_16 =                 ld\_ev\_guid\_16. " Loop at the internal table to insert data into z178\_lineitems loop at itab into wa\_tab. z178\_lineitems-customer = wa\_tab-customer. z178\_lineitems-product = wa\_tab-product. z178\_lineitems-quantity = wa\_tab-quantity. z178\_lineitems-OrderNo = ld\_ev\_guid\_16. z178\_lineitems-Date\_Created = sy-datum. modify z178\_lineitems. endloop. " Delete from the cart after converting into order delete from z178\_cart where customer = p\_cust.   WRITE 'Order Placed'. ENDIF. \*ELSEIF p\_report = abap\_true. \* \* \*  " Get customer products from the cart \*  rp\_cart = lo\_cart->get\_report( ). \* \*  " Get output reference of the ALV table \*  cl\_salv\_table=>factory( IMPORTING r\_salv\_table = DATA(lr\_alv) \*                          CHANGING t\_table = rp\_cart ). \* \*  " Display the result as ALV table \*  lr\_alv->display( ).

# ZCL178\_SHOPPING\_CART\_FINALPROJ

# class ZCL178\_SHOPPING\_CART\_FINALPROJ definition   public   final   create public . public section.   types:     "This is defining the structure of our table we will use to track products     BEGIN OF y\_cart,              product     TYPE z178\_product-product,              description TYPE z178\_product\_txt-description,              quantity    TYPE z178\_cart-quantity,              uom         TYPE z178\_product-uom,              price       TYPE z178\_product-price,              Total       Type Z178\_TOTAL\_AMOUNT,            END OF y\_cart.         Types:    BEGIN OF l\_cart,              OrderNo    TYPE z178\_lineitems-OrderNo,              customer    TYPE SCUSTOM-ID,              product     TYPE z178\_product-product,              quantity    TYPE z178\_cart-quantity,              price       TYPE z178\_product-price,              Date\_Created        TYPE z178\_lineitems-Date\_Created,              Total       Type Z178\_TOTAL\_AMOUNT,            END OF l\_cart .   types:     yt\_cart TYPE STANDARD TABLE OF y\_cart WITH KEY product,     AT\_cart TYPE STANDARD TABLE OF l\_cart WITH KEY orderNo.       "These are the prototypes of our methods we will define in the next section       "In chapter 10 we have added checking of the customer to see if it is in the       "SCUSTOM table   methods CONSTRUCTOR     importing       !IP\_CUSTOMER type SCUSTOM-ID     raising       Z178\_CX\_SHOPPING\_CART.   methods ADD\_PRODUCT     importing       !IP\_PRODUCT type Z178\_PRODUCT-PRODUCT       !IP\_QUANTITY type Z178\_CART-QUANTITY optional .   methods REMOVE\_PRODUCT     importing       !IP\_PRODUCT type Z178\_PRODUCT-PRODUCT .   methods CLEAR\_CART .       "This has to be yt\_cart referring to the actual table       "I had put y\_cart - directly to the type and received errors       "This is the shopping cart method prototype that returns what is in the current cart   methods GET\_CART     returning       value(RT\_CART) type YT\_CART . \*  methods GET\_Report \*    returning \*      value(RP\_CART) type AT\_CART . \* \* \*   methods GET\_Report\_All \*    returning \*      value(RP\_CART) type AT\_CART .   PROTECTED SECTION.   PRIVATE SECTION.     "This is accessible from the methods within this class but not outside of this class     DATA: d\_customer TYPE scustom-id. ENDCLASS. CLASS ZCL178\_SHOPPING\_CART\_FINALPROJ IMPLEMENTATION. \* <SIGNATURE>---------------------------------------------------------------------------------------+ \* | Instance Public Method ZCL178\_SHOPPING\_CART\_FINALPROJ->ADD\_PRODUCT \* +-------------------------------------------------------------------------------------------------+ \* | [--->] IP\_PRODUCT                     TYPE        Z178\_PRODUCT-PRODUCT \* | [--->] IP\_QUANTITY                    TYPE        Z178\_CART-QUANTITY(optional) \* +--------------------------------------------------------------------------------------</SIGNATURE>   METHOD ADD\_PRODUCT.     "Here we add a new product to the cart via passed parameter     DATA: ls\_cart TYPE z178\_cart.     ls\_cart-customer = d\_customer.     ls\_cart-product = ip\_product.     IF ip\_quantity IS NOT INITIAL.       ls\_cart-quantity = ip\_quantity.     ELSE.       ls\_cart-quantity = 1.     ENDIF.     MODIFY z178\_cart FROM ls\_cart.   ENDMETHOD. \* <SIGNATURE>---------------------------------------------------------------------------------------+ \* | Instance Public Method ZCL178\_SHOPPING\_CART\_FINALPROJ->CLEAR\_CART \* +-------------------------------------------------------------------------------------------------+ \* +--------------------------------------------------------------------------------------</SIGNATURE>   METHOD CLEAR\_CART.     "Here we clear the cart for the customer - no actual passing needed     DELETE FROM z178\_cart WHERE customer = d\_customer.   ENDMETHOD. \* <SIGNATURE>---------------------------------------------------------------------------------------+ \* | Instance Public Method ZCL178\_SHOPPING\_CART\_FINALPROJ->CONSTRUCTOR \* +-------------------------------------------------------------------------------------------------+ \* | [--->] IP\_CUSTOMER                    TYPE        SCUSTOM-ID \* | [!CX!] Z178\_CX\_SHOPPING\_CART \* +--------------------------------------------------------------------------------------</SIGNATURE>   METHOD CONSTRUCTOR.     "Here we establish that this instance of the class if for a particular customer     "d\_customer = ip\_customer.     "Here we actually look up the customer to     "See if they are in the customer table     "If they are not we raise our custom error     SELECT SINGLE id     INTO @d\_customer     FROM scustom     WHERE id = @ip\_customer.     IF sy-subrc <> 0.       RAISE EXCEPTION TYPE Z178\_CX\_SHOPPING\_CART.     ENDIF.   ENDMETHOD. \* <SIGNATURE>---------------------------------------------------------------------------------------+ \* | Instance Public Method ZCL178\_SHOPPING\_CART\_FINALPROJ->GET\_CART \* +-------------------------------------------------------------------------------------------------+ \* | [<-()] RT\_CART                        TYPE        YT\_CART \* +--------------------------------------------------------------------------------------</SIGNATURE>   METHOD GET\_CART.     "Our get\_cart method looks into our table and returns     "the contents of our cart to a table that is the same type     "as what we defined above in our class.     SELECT c~product,           q~description,           quantity,           uom,           price,           ( quantity \* price ) as Total     FROM z178\_cart AS c     INNER JOIN z178\_product AS p       ON c~product EQ p~product       inner join Z178\_product\_txt as q       on c~product EQ q~product       WHERE c~customer = @d\_customer     INTO TABLE @rt\_cart.   ENDMETHOD. \* <SIGNATURE>---------------------------------------------------------------------------------------+ \* | Instance Public Method ZCL178\_SHOPPING\_CART\_FINALPROJ->REMOVE\_PRODUCT \* +-------------------------------------------------------------------------------------------------+ \* | [--->] IP\_PRODUCT                     TYPE        Z178\_PRODUCT-PRODUCT \* +--------------------------------------------------------------------------------------</SIGNATURE>   METHOD REMOVE\_PRODUCT.     "Here we remove a product from the cart via passed parameter     DATA: ls\_cart TYPE z178\_cart.     ls\_cart-customer = d\_customer.     ls\_cart-product = ip\_product.     DELETE z178\_cart FROM ls\_cart.   ENDMETHOD. ENDCLASS.

# z178\_product\_maint\_finalprojec.

# REPORT z178\_product\_maint\_finalprojec. TYPE-POOLS truxs. TABLES: z178\_product, z178\_product\_txt. " Selection Screein to Input Production Information SELECTION-SCREEN BEGIN OF BLOCK product WITH FRAME TITLE text-001. PARAMETERS: p\_prod  TYPE z178\_product-product,  " Product name             p\_desc  TYPE z178\_product\_txt-description LOWER CASE, " Product Description             p\_price TYPE z178\_product-price,    " Product Price             p\_curr  TYPE z178\_product-currency, " Product Currency             p\_uom   TYPE z178\_product-uom.      " Product Unit of Measurement SELECTION-SCREEN END OF BLOCK product. " Selection Screen for actions SELECTION-SCREEN BEGIN OF BLOCK action WITH FRAME TITLE text-002. PARAMETERS: p\_upd    TYPE boolean RADIOBUTTON GROUP act,    " Action for product add/update             p\_rem    TYPE boolean RADIOBUTTON GROUP act,    " Action for removing prduct             p\_dis    TYPE boolean RADIOBUTTON GROUP act,    " Action for displaying product list             p\_upload TYPE boolean RADIOBUTTON GROUP act.    " Action for uploading product SELECTION-SCREEN END OF BLOCK action. " Selection screen for uploading product SELECTION-SCREEN BEGIN OF BLOCK upload WITH FRAME TITLE text-005. PARAMETERS: p\_file TYPE  rlgrap-filename.   " .xls file location PARAMETERS: p\_head TYPE char01 DEFAULT 'X'. " Check it the file has header SELECTION-SCREEN END OF BLOCK upload. " Declaring data types for internal table and work area DATA: s\_product     TYPE z178\_product, s\_product\_txt TYPE z178\_product\_txt. s\_product\_txt-product = p\_prod. s\_product\_txt-description = p\_desc. s\_product\_txt-language = sy-langu. s\_product-product = p\_prod. s\_product-price = p\_price. s\_product-currency = p\_curr. s\_product-uom = p\_uom. " Declaring data types for internal table and work area TYPES: BEGIN OF t\_datatab,          product     TYPE z178\_zproductid,          price       TYPE z178\_prod\_price,          currency    TYPE z178\_product\_currency,          uom         TYPE z178\_product\_uom,          description TYPE z178\_zproduct\_desc,        END OF t\_datatab. DATA: it\_datatab TYPE STANDARD TABLE OF t\_datatab,       wa\_datatab TYPE t\_datatab. DATA: it\_raw TYPE truxs\_t\_text\_data. \* " Action if the product add/update radio button selected IF p\_upd = abap\_true.   MODIFY z178\_product FROM s\_product.   MODIFY z178\_product\_txt FROM s\_product\_txt.   WRITE: 'Update Completed'(003). " Action if the product remove radio button selected ELSEIF p\_rem = abap\_true.   DELETE FROM z178\_product\_txt WHERE product = p\_prod.   DELETE z178\_product FROM s\_product.   WRITE: 'Delete Completed'(004). " Action if the product display radio button selected ELSEIF p\_dis = abap\_true.   SELECT z178\_product~product, description, price, currency, uom   FROM z178\_product INNER JOIN z178\_product\_txt   ON z178\_product~product = z178\_product\_txt~product   WHERE language = @sy-langu   INTO TABLE @DATA(t\_products).   cl\_salv\_table=>factory(   IMPORTING r\_salv\_table = DATA(gr\_alv)   CHANGING t\_table = t\_products ).   gr\_alv->display( ). " Action if the product upload radio button selected ELSEIF p\_upload = abap\_true.   CALL FUNCTION 'TEXT\_CONVERT\_XLS\_TO\_SAP' " Call BAPI function for uploading excel file     EXPORTING \*     I\_FIELD\_SEPERATOR    =       i\_line\_header        = p\_head       " Check header       i\_tab\_raw\_data       = it\_raw       " WORK TABLE       i\_filename           = p\_file       " File Name     TABLES       i\_tab\_converted\_data = it\_datatab[]  "ACTUAL DATA     EXCEPTIONS                             " Error handling if upload fails       conversion\_failed    = 1       OTHERS               = 2. " Error message display if error occurs   IF sy-subrc <> 0.     MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno             WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.   ENDIF. " Loop into internal table to upload data into z178\_product and z178\_product\_txt table   LOOP AT it\_datatab INTO wa\_datatab.     WRITE:/ wa\_datatab-product, wa\_datatab-price, wa\_datatab-currency, wa\_datatab-uom, wa\_datatab-description.     z178\_product-product = wa\_datatab-product.     z178\_product-price = wa\_datatab-price.     z178\_product-currency = wa\_datatab-currency.     z178\_product-uom = wa\_datatab-uom.     z178\_product\_txt-product =  wa\_datatab-product.     z178\_product\_txt-description = wa\_datatab-description.     z178\_product\_txt-language = sy-langu.     MODIFY z178\_product.     MODIFY z178\_product\_txt.   ENDLOOP. Write:/ 'DATA FROM EXCEL FILE SUCCESSFULLY UPLOADED'. ENDIF. " SELECT SCREEN FOR REQUESTING EXCEL FILE BY CALLING BAPI FUNCTION 'F4\_FILENAME' AT SELECTION-SCREEN ON VALUE-REQUEST FOR p\_file.   CALL FUNCTION 'F4\_FILENAME'     EXPORTING       field\_name = 'P\_FILE'     IMPORTING       file\_name  = p\_file. \* \* \*START-OF-SELECTION. \*  " Convert Excel Data to SAP internal Table Data \* \* \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\* END-OF-SELECTION. \*END-OF-SELECTION.

# Z178\_SALES\_REPORT

# \*&---------------------------------------------------------------------\* \*& Report  Z178\_SALES\_REPORT \*& \*&---------------------------------------------------------------------\* \*& This report will show the sales orders line items and total sum of sale \*& \*&---------------------------------------------------------------------\* REPORT z178\_sales\_report LINE-SIZE 150       " Length of the line LINE-COUNT 50(3)    " Number of Line per page NO STANDARD PAGE HEADING. TABLES: z178\_cart, z178\_lineitems, z178\_product, z178\_product\_txt. \* Table structure TYPES:    BEGIN OF l\_cart,             orderno      TYPE z178\_lineitems-orderno,             customer     TYPE scustom-id,             product      TYPE z178\_product-product,             quantity     TYPE z178\_cart-quantity,             price        TYPE z178\_product-price,             date\_created TYPE z178\_lineitems-date\_created,             total        TYPE z178\_total\_amount,           END OF l\_cart . DATA: wa\_r TYPE l\_cart,       it\_r TYPE STANDARD TABLE OF l\_cart. \* Table Structure for getting total sum of sales TYPES:    BEGIN OF sum\_cart,             total TYPE z178\_total\_amount,           END OF sum\_cart . DATA: wa\_sum TYPE sum\_cart,       it\_sum TYPE STANDARD TABLE OF sum\_cart. \* Line at the top of page TOP-OF-PAGE.   WRITE:/ 'Z178 SHOPPING CART SALES REPORT' COLOR 7.   ULINE.   WRITE:/ 'ORDER NUMBER' COLOR 1,   38 'CUSTOMER' COLOR 2,   53 'PRODUCTS' COLOR 3,   73 'QUANTITITY' COLOR 4,   93 'PRICE'      COLOR 6,   113 'DATE'       COLOR 7,   133 'TOTAL PRICE' COLOR 5.   ULINE.   END-OF-PAGE. \* Query to get data into the sales report START-OF-SELECTION.   SELECT            c~orderno,            c~customer,            c~product,            quantity,            price,            c~date\_created,            ( quantity \* price ) AS total      FROM z178\_lineitems AS c      INNER JOIN z178\_product AS p        ON c~product EQ p~product \*      inner join Z178\_product\_txt as q \*      on c~product EQ q~product        ORDER BY c~customer \*      WHERE c~customer = @d\_customer   INTO TABLE @it\_r. \* Loop at the internal table to write data   LOOP AT it\_r INTO wa\_r.     WRITE:/  wa\_r-orderno,     38 wa\_r-customer,     54 wa\_r-product,     64 wa\_r-quantity,     84 wa\_r-price,     114  wa\_r-date\_created,     134 wa\_r-total.   ENDLOOP. END-OF-SELECTION.   ULINE. \* SQL statement to find sum total of sale   SELECT             SUM( quantity \* price ) AS total       FROM z178\_lineitems AS c       INNER JOIN z178\_product AS p         ON c~product EQ p~product         INNER JOIN z178\_product\_txt AS q         ON c~product EQ q~product \*      WHERE c~customer = @d\_customer   INTO TABLE @it\_sum.   LOOP AT it\_sum INTO wa\_sum.     WRITE:/ 'SHOPPING CART REPORT HAS BEEN CREATED' COLOR 7,     134 wa\_sum-total.   ENDLOOP.   ULINE.   SKIP.