public void ValidateTheResponseWhenAttributesAreUpdatedThroughBlkEditPricing(

String advertiser, String agency, String AE, String calender,

String startPeriodValuelocal, String endPeriodValuelocal,

String packages, String updQty, String updUcbd

) throws InterruptedException {

java.util.function.Function<String, String> cleanCurrency = val -> val == null ? ""

: val.replace("$", "").replace(",", "").replace("%", "").trim();

DigitalPlannerHeaderPage digitalHeader = new DigitalPlannerHeaderPage(getPageBrowser());

digitalHeader.createOnlyDigitalPlanner(

digital\_PlanName + "AdvancedPricingAllocation\_CCF",

AutoConfigPlannerHeader.digitalOrderType, "", advertiser, "", AE,

calender, startPeriodValuelocal, endPeriodValuelocal, ""

);

digitalHeader.clickSaveHeader();

PlannerDigitalWorkspacePage digitalWorkspace = new PlannerDigitalWorkspacePage(getPageBrowser());

digitalWorkspace.navigateToWorkSpaceTab();

String[] packageNames = packages.split(";");

digitalWorkspace.addPackageInChooserDigital(packageNames[0]);

String lineNumber = digitalWorkspace.getProductLineNo(packageNames[0]);

digitalWorkspace.expandPackageLineItem(lineNumber);

List<String> childLines = digitalWorkspace.getPackageChildLineListNo(lineNumber);

String childLine1 = childLines.get(0);

final String[] bulkActions = AutoConfigDigitalPlannerWorkSpace.bulkActionsUi.split(";");

String[] ucbd = updUcbd.split(";");

String[] qty = updQty.split(";");

// Save original value for unchanged assertion

String originalChildBudgetValue = digitalWorkspace.getDigitalProductLineTextFieldvalue("desiredBudget", 2);

// ========== 1. Update Client Budget ==========

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.clientBudget, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.desiredBudget, AutoConfigDigitalPlannerWorkSpace.clientBudgetInputValue

);

String newBudgetValue = AutoConfigDigitalPlannerWorkSpace.clientBudgetInputValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("desiredBudget", 1)),

cleanCurrency.apply(newBudgetValue),

"Parent client budget does not match after bulk edit"

);

digitalWorkspace.expandPackageLineItem(lineNumber);

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean clientBudgetDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='desiredBudget\_input' and @aria-label='Client Budget']");

System.out.println("'Client Budget' bulk edit input for child is " + (clientBudgetDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), clientBudgetDisabled, "Client Budget field is not disabled for child line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("desiredBudget", 2)),

cleanCurrency.apply(originalChildBudgetValue),

"Child client budget should remain unchanged after bulk edit"

);

// ========== 2. Update UCBD ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.pricingTabUcbd, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.unitCostBeforeDiscount, ucbd[0]

);

String newUCBDValue = ucbd[0];

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("ucbd", 2)),

cleanCurrency.apply(newUCBDValue),

"Child UCBD does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean ucbdDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='ucbd\_input' and @aria-label='Unit cost before Discounts']");

System.out.println("'UCBD' bulk edit input for parent is " + (ucbdDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), ucbdDisabled, "UCBD should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 3. Update Desired Quantity ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.desireQuantity, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.desireQuantityInput, AutoConfigDigitalPlannerWorkSpace.desireQuantityInputValue

);

String newDesiredQty = AutoConfigDigitalPlannerWorkSpace.desireQuantityInputValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("desiredQuantity", 2)),

cleanCurrency.apply(newDesiredQty),

"Child desired quantity does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean desiredQtyDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='desiredQuantity\_input' and @aria-label='Desired Quantity']");

System.out.println("'Desired Quantity' bulk edit input for parent is " + (desiredQtyDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), desiredQtyDisabled, "Desired Quantity should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 4. Update NUC ==========

digitalWorkspace.expandPackageLineItem(lineNumber);

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.NUC, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.NUC, AutoConfigDigitalPlannerWorkSpace.NUCValue

);

String newNUC = AutoConfigDigitalPlannerWorkSpace.NUCValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("netUnitCost", 2)),

cleanCurrency.apply(newNUC),

"Child NUC does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean nucDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='netUnitCost\_input' and @aria-label='Net Unit Cost']");

System.out.println("'NUC' bulk edit input for parent is " + (nucDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), nucDisabled, "NUC should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 5. Update Quantity ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.quantity, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.quantity, qty[0]

);

String newQtyValue = qty[0];

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("quantity", 2)),

cleanCurrency.apply(newQtyValue),

"Child quantity does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean quantityDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='quantity\_input' and @aria-label='Quantity']");

System.out.println("'Quantity' bulk edit input for parent is " + (quantityDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), quantityDisabled, "Quantity should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 6. Update Production Quantity ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.prodQuantity, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.prodQty, AutoConfigDigitalPlannerWorkSpace.prodQuantityValue

);

String newProdQty = AutoConfigDigitalPlannerWorkSpace.prodQuantityValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("productionQuantity", 2)),

cleanCurrency.apply(newProdQty),

"Child production quantity does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean prodQtyDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='productionQuantity\_input' and @aria-label='Production Quantity']");

System.out.println("'Production Quantity' bulk edit input for parent is " + (prodQtyDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), prodQtyDisabled, "Production Quantity should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 7. Update SOV% ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.SOV, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.sov, AutoConfigDigitalPlannerWorkSpace.sovValue

);

String newSOVValue = AutoConfigDigitalPlannerWorkSpace.sovValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("sov", 2)),

cleanCurrency.apply(newSOVValue),

"Child SOV% does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean sovDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='sov\_input' and @aria-label='SOV%']");

System.out.println("'SOV%' bulk edit input for parent is " + (sovDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), sovDisabled, "SOV% should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 8. Update Production Buffer ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.prodBuff, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.prodBuffer, AutoConfigDigitalPlannerWorkSpace.prodBuffInuput

);

String newProdBuffer = AutoConfigDigitalPlannerWorkSpace.prodBuffInuput;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

getCustomeVerification().verifyEquals(getSoftAssert(),

cleanCurrency.apply(digitalWorkspace.getDigitalProductLineTextFieldvalue("productionBuffer", 2)),

cleanCurrency.apply(newProdBuffer),

"Child production buffer does not match after bulk edit"

);

digitalWorkspace.openBulkEditSelectingLineNo(lineNumber);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

boolean prodBufferDisabled = digitalWorkspace.isBulkEditInputDisabledByXpathWithRetry("//input[@id='productionBuffer\_input' and @aria-label='Production Buffer']");

System.out.println("'Production Buffer' bulk edit input for parent is " + (prodBufferDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), prodBufferDisabled, "Production Buffer should be disabled for parent line in bulk edit");

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.cancelButton);

// ========== 9. Update Discount (Disc01 Gross): ADVANCED EDIT child value check, parent disabled check ==========

digitalWorkspace.openBulkEditSelectingLineNo(childLine1);

digitalWorkspace.navigateToTabsInBulkEdit(AutoConfigDigitalPlannerWorkSpace.advanceEditPricingTab);

digitalWorkspace.selectBulkEditAction(AutoConfigDigitalPlannerWorkSpace.disc01AppliedGross, bulkActions[0]);

digitalWorkspace.enterValueBulkEditPricingField(

AutoConfigDigitalPlannerWorkSpace.disc01AppliedGrossInput, AutoConfigDigitalPlannerWorkSpace.disc01AppliedGrossValue

);

String newDiscValue = AutoConfigDigitalPlannerWorkSpace.disc01AppliedGrossValue;

digitalWorkspace.clickOnSaveorCancelInBulk(AutoConfigDigitalPlannerWorkSpace.saveButton);

Thread.sleep(3000);

digitalWorkspace.expandPackageLineItem(lineNumber);

// Child: advanced edit value check using field constant

digitalWorkspace.navigateToAdvanceEditPage(childLine1);

String advEditDiscChild = cleanCurrency.apply(

digitalWorkspace.getAdvanceEditLineDetailsFieldValues(AutoConfigDigitalPlannerWorkSpace.disc01AppliedGrossInput)

);

System.out.println("'Disc01 Gross' in advanced edit on child is " + advEditDiscChild);

getCustomeVerification().verifyEquals(getSoftAssert(), cleanCurrency.apply(newDiscValue), advEditDiscChild, "Child discount value in advanced edit does not match after bulk edit");

digitalWorkspace.clickOnAdvEditCancelButton();

// Parent: advanced edit disabled check using field constant/label

digitalWorkspace.navigateToAdvanceEditPage(lineNumber);

boolean discAdvEditDisabled = digitalWorkspace.isAdvanceEditInputDisabledByFieldKey(AutoConfigDigitalPlannerWorkSpace.disc01AppliedGrossInput);

System.out.println("'Disc01 Gross' input for parent advanced edit is " + (discAdvEditDisabled ? "DISABLED" : "ENABLED"));

getCustomeVerification().verifyTrue(getSoftAssert(), discAdvEditDisabled, "Disc01 Gross should be disabled in advanced edit for parent");

digitalWorkspace.clickOnAdvEditCancelButton();

digitalWorkspace.waitForDuration(10000);

getCustomeVerification().assertAll(getSoftAssert());

}

// Enhanced helper method for bulk edit input disabled check with retry logic

public boolean isBulkEditInputDisabledByXpathWithRetry(String xpath) {

try {

Locator input = getPageBrowser().locator(xpath);

input.waitFor(new Locator.WaitForOptions().setTimeout(5000));

if (input.count() == 0) {

System.out.println("Bulk edit input not found for xpath: " + xpath);

return false;

}

String disabledAttr = input.getAttribute("disabled");

boolean isDisabled = disabledAttr != null;

System.out.println("Bulk edit input [xpath=" + xpath + "] disabled? " + isDisabled);

return isDisabled;

} catch (Exception e) {

System.out.println("Error checking bulk edit input disabled state for xpath: " + xpath + ", error: " + e.getMessage());

return false;

}

}

// Enhanced helper method for advanced edit input disabled check

public boolean isAdvanceEditInputDisabledByFieldKey(String fieldKeyOrInputName) {

try {

// Multiple locator strategies for finding the discount input in advanced edit

String[] locatorStrategies = {

"//input[contains(@aria-label,'Disc\_01\_Gross') and contains(@aria-label,'Applied To Gross')]",

"//input[contains(@aria-label,'Disc\_01\_Gross [Applied To Gross]')]",

"//input[contains(@aria-label,'Disc\_01\_Gross')]",

"//input[contains(@id,'" + fieldKeyOrInputName + "')]",

"//input[contains(@id,'discount') and contains(@aria-label,'Gross')]"

};

Locator input = null;

for (String strategy : locatorStrategies) {

input = getPageBrowser().locator(strategy);

try {

input.waitFor(new Locator.WaitForOptions().setTimeout(3000));

if (input.count() > 0) {

System.out.println("Found advanced edit input using strategy: " + strategy);

break;

}

} catch (Exception e) {

// Continue to next strategy

continue;

}

}

if (input == null || input.count() == 0) {

System.out.println("Advanced edit input not found for field: " + fieldKeyOrInputName);

return false;

}

String disabledAttr = input.getAttribute("disabled");

boolean isDisabled = disabledAttr != null;

System.out.println("Advanced edit [fieldKey=" + fieldKeyOrInputName + "] disabled? " + isDisabled);

return isDisabled;

} catch (Exception e) {

System.out.println("Error checking advanced edit input disabled state for field: " + fieldKeyOrInputName + ", error: " + e.getMessage());

return false;

}

}