

Apex Specialist

(Superbadge)

What You'll Be Doing to Earn This Superbadge

1. Automate record creation using Apex triggers
2. Synchronize Salesforce data with an external system using asynchronous REST callouts
3. Schedule synchronization using Apex code
4. Test automation logic to confirm Apex trigger side effects
5. Test integration logic using callout mocks
6. Test scheduling logic to confirm action gets queued

Concepts Tested in This Superbadge

- Apex Triggers
- Asynchronous Apex
- Apex Integration
- Apex Testing

Task 1 : Automated Record Creation

Create a Maintenance Request For each new Equipment automatically.

MaintenanceRequestHelper.apxc

```
1 public with sharing class MaintenanceRequestHelper {
2     public static void updateWorkOrders(List<Case> updWorkOrders,
3     Map<Id,Case> nonUpdCaseMap) {
4         Set<Id> validIds = new Set<Id>();
5
6         For (Case c : updWorkOrders){
7             if (nonUpdCaseMap.get(c.Id).Status != 'Closed' && c.Status ==
8             'Closed'){
9                 if (c.Type == 'Repair' || c.Type == 'Routine Maintenance'){
10                     validIds.add(c.Id);
11
12                 }
13             }
14         }
15     }
16 }
```

```

14     }
15
16     if (!validIds.isEmpty()){
17         List<Case> newCases = new List<Case>();
18         Map<Id,Case> closedCasesM = new Map<Id,Case>([SELECT Id,
Vehicle__c, Equipment__c, Equipment__r.Maintenance_Cycle__c,(SELECT
Id,Equipment__c,Quantity__c FROM Equipment_Maintenance_Items__r)
19                                     FROM Case WHERE Id
IN :validIds]);
20         Map<Id,Decimal> maintenanceCycles = new Map<ID,Decimal>();
21         AggregateResult[] results = [SELECT Maintenance_Request__c,
MIN(Equipment__r.Maintenance_Cycle__c)cycle FROM
Equipment_Maintenance_Item__c WHERE Maintenance_Request__c IN :ValidIds
GROUP BY Maintenance_Request__c];
22
23         for (AggregateResult ar : results){
24             maintenanceCycles.put((Id) ar.get('Maintenance_Request__c'),
(Decimal) ar.get('cycle'));
25         }
26
27         for(Case cc : closedCasesM.values()){
28             Case nc = new Case (
29                 ParentId = cc.Id,
30                 Status = 'New',
31                 Subject = 'Routine Maintenance',
32                 Type = 'Routine Maintenance',
33                 Vehicle__c = cc.Vehicle__c,
34                 Equipment__c =cc.Equipment__c,
35                 Origin = 'Web',
36                 Date_Reported__c = Date.Today()
37
38             );
39
40             If (maintenanceCycles.containsKey(cc.Id)){
41                 nc.Date_Due__c = Date.today().addDays((Integer)
maintenanceCycles.get(cc.Id));
42             } else {
43                 nc.Date_Due__c = Date.today().addDays((Integer)
cc.Equipment__r.maintenance_Cycle__c);
44             }
45
46             newCases.add(nc);

```

```

47         }
48
49         insert newCases;
50
51         List<Equipment_Maintenance_Item__c> clonedWPs = new
List<Equipment_Maintenance_Item__c>();
52         for (Case nc : newCases){
53             for (Equipment_Maintenance_Item__c wp :
closedCasesM.get(nc.ParentId).Equipment_Maintenance_Items__r){
54                 Equipment_Maintenance_Item__c wpClone = wp.clone();
55                 wpClone.Maintenance_Request__c = nc.Id;
56                 ClonedWPs.add(wpClone);
57
58             }
59         }
60         insert ClonedWPs;
61     }
62 }
63 }

```

MaintenanceRequest.apxt

```

1  trigger MaintenanceRequest on Case (before update, after update) {
2
3      if(Trigger.isUpdate && Trigger.isAfter){
4
5          MaintenanceRequestHelper.updateWorkOrders(Trigger.New,
Trigger.OldMap);
6
7      }
8
9  }

```

Task 2: Synchronize Salesforce data with an external system

1. Make a REST callout to an external warehouse system to get a list of equipment that needs to be updated.
2. The callout's JSON response returns the equipment records that you upsert in Salesforce.

WarehouseCalloutService.apxc :-

```
1 public with sharing class WarehouseCalloutService implements Queueable {
2     private static final String WAREHOUSE_URL = 'https://th-superbadge-
3
4     @future(callout=true)
5     public static void runWarehouseEquipmentSync(){
6         Http http = new Http();
7         HttpRequest request = new HttpRequest();
8
9         request.setEndpoint(WAREHOUSE_URL);
10        request.setMethod('GET');
11        HttpResponse response = http.send(request);
12
13        List<Product2> warehouseEq = new List<Product2>();
14
15        if (response.getStatusCode() == 200){
16            List<Object> jsonResponse =
17                (List<Object>)JSON.deserializeUntyped(response.getBody());
18            System.debug(response.getBody());
19
20            for (Object eq : jsonResponse){
21                Map<String,Object> mapJson = (Map<String,Object>)eq;
22                Product2 myEq = new Product2();
23                myEq.Replacement_Part__c = (Boolean)
24                    mapJson.get('replacement');
25                myEq.Name = (String) mapJson.get('name');
26                myEq.Maintenance_Cycle__c = (Integer)
27                    mapJson.get('maintenanceperiod');
28                myEq.Lifespan_Months__c = (Integer) mapJson.get('lifespan');
29                myEq.Cost__c = (Integer) mapJson.get('cost');
30                myEq.Warehouse_SKU__c = (String) mapJson.get('sku');
31                myEq.Current_Inventory__c = (Double)
32                    mapJson.get('quantity');
33                myEq.ProductCode = (String) mapJson.get('_id');
34                warehouseEq.add(myEq);
35            }
36        }
37    }
38 }
```

```
31         }
32
33         if (warehouseEq.size() > 0){
34             upsert warehouseEq;
35             System.debug('Your equipment was synced with the warehouse
36
37         }
38     }
39
40     public static void execute (QueueableContext context){
41         runWarehouseEquipmentSync();
42     }
43
44 }
```

open execute anonymous window (CTRL+E) and run this method

```
1 System.enqueueJob(new WarehouseCalloutService());
```

Task 3 : Schedule synchronization using Apex code

WarehouseSyncShedule.apxc :-

```
1 global with sharing class WarehouseSyncSchedule implements Schedulable{
2     global void execute(SchedulableContext ctx){
3         System.enqueueJob(new WarehouseCalloutService());
4     }
5 }
```

Create A job schedule

Job Name = WarehouseSyncScheduleJob

Apex Class = WarehouseSyncSchedule

The screenshot shows the Salesforce 'Apex Classes' setup page. At the top, there's a 'SETUP' link and the 'Apex Classes' title. Below this, there are 'Save' and 'Cancel' buttons. The 'Job Name' field is set to 'WarehouseSyncScheduleJob'. The 'Apex Class' field is set to 'WarehouseSyncSchedule'. Under the 'Schedule Apex Execution' section, the 'Frequency' is set to 'Weekly'. A dropdown menu is open, showing 'Recurs every week on' with checkboxes for all days of the week (Sunday through Saturday), all of which are checked. The 'Start' date is '6/25/2022' with a calendar icon. The 'End' date is '7/25/2022' with a calendar icon. The 'Preferred Start Time' is set to '--None--'. A note at the bottom states: 'Exact start time will depend on job queue activity.'

Task 4 : Test automation logic

Testing the generated automation logic and achieving 100% code coverage.

MaintenanceRequestHelperTest.apxc :

```
1  @istest
2  public with sharing class MaintenanceRequestHelperTest {
3
4      private static final string STATUS_NEW = 'New';
5      private static final string WORKING = 'Working';
6      private static final string CLOSED = 'Closed';
7      private static final string REPAIR = 'Repair';
8      private static final string REQUEST_ORIGIN = 'Web';
9      private static final string REQUEST_TYPE = 'Routine Maintenance';
10     private static final string REQUEST_SUBJECT = 'Testing subject';
11
12     PRIVATE STATIC Vehicle__c createVehicle(){
13         Vehicle__c Vehicle = new Vehicle__C(name = 'SuperTruck');
14         return Vehicle;
15     }
16
17     PRIVATE STATIC Product2 createEq(){
18         product2 equipment = new product2(name = 'SuperEquipment',
19                                           lifespan_months__C = 10,
20                                           maintenance_cycle__C = 10,
21                                           replacement_part__c = true);
22         return equipment;
23     }
24
25     PRIVATE STATIC Case createMaintenanceRequest(id vehicleId, id
equipmentId){
26         case cs = new case(Type=REPAIR,
27                           Status=STATUS_NEW,
28                           Origin=REQUEST_ORIGIN,
29                           Subject=REQUEST_SUBJECT,
30                           Equipment__c=equipmentId,
31                           Vehicle__c=vehicleId);
32         return cs;
33     }
34
35     PRIVATE STATIC Equipment_Maintenance_Item__c createWorkPart(id
```

```

equipmentId,id requestId){
36     Equipment_Maintenance_Item__c wp = new
Equipment_Maintenance_Item__c(Equipment__c = equipmentId,
37     Maintenance_Request__c = requestId);
38     return wp;
39 }
40
41
42 @istest
43 private static void testMaintenanceRequestPositive(){
44     Vehicle__c vehicle = createVehicle();
45     insert vehicle;
46     id vehicleId = vehicle.Id;
47
48     Product2 equipment = createEq();
49     insert equipment;
50     id equipmentId = equipment.Id;
51
52     case somethingToUpdate =
createMaintenanceRequest(vehicleId,equipmentId);
53     insert somethingToUpdate;
54
55     Equipment_Maintenance_Item__c workP =
createWorkPart(equipmentId,somethingToUpdate.id);
56     insert workP;
57
58     test.startTest();
59     somethingToUpdate.status = CLOSED;
60     update somethingToUpdate;
61     test.stopTest();
62
63     Case newReq = [Select id, subject, type, Equipment__c,
Date_Reported__c, Vehicle__c, Date_Due__c
64                     from case
65                     where status =:STATUS_NEW];
66
67     Equipment_Maintenance_Item__c workPart = [select id
68                                             from
Equipment_Maintenance_Item__c
69                                             where
Maintenance_Request__c =:newReq.Id];
70

```



```

71     system.assert(workPart != null);
72     system.assert(newReq.Subject != null);
73     system.assertEquals(newReq.Type, REQUEST_TYPE);
74     SYSTEM.assertEquals(newReq.Equipment__c, equipmentId);
75     SYSTEM.assertEquals(newReq.Vehicle__c, vehicleId);
76     SYSTEM.assertEquals(newReq.Date_Reported__c, system.today());
77 }
78
79 @istest
80 private static void testMaintenanceRequestNegative(){
81     Vehicle__C vehicle = createVehicle();
82     insert vehicle;
83     id vehicleId = vehicle.Id;
84     product2 equipment = createEq();
85     insert equipment;
86     id equipmentId = equipment.Id;
87
88     case emptyReq = createMaintenanceRequest(vehicleId,equipmentId);
89     insert emptyReq;
90
91     Equipment_Maintenance_Item__c workP = createWorkPart(equipmentId,
emptyReq.Id);
92     insert workP;
93     test.startTest();
94     emptyReq.Status = WORKING;
95     update emptyReq;
96     test.stopTest();
97
98     list<case> allRequest = [select id
99                             from case];
100
101         Equipment_Maintenance_Item__c workPart = [select id
102                                                     from
103 Equipment_Maintenance_Item__c
104                                                     where
105 Maintenance_Request__c = :emptyReq.Id];
106
107     system.assert(workPart != null);
108     system.assert(allRequest.size() == 1);
109 }
110
111 @istest
112 private static void testMaintenanceRequestBulk(){

```

```

110         list<Vehicle__C> vehicleList = new list<Vehicle__C>();
111         list<Product2> equipmentList = new list<Product2>();
112         list<Equipment_Maintenance_Item__c> workPartList = new
list<Equipment_Maintenance_Item__c>();
113         list<case> requestList = new list<case>();
114         list<id> oldRequestIds = new list<id>();
115         for(integer i = 0; i < 300; i++){
116             vehicleList.add(createVehicle());
117             equipmentList.add(createEq());
118         }
119         insert vehicleList;
120         insert equipmentList;
121         for(integer i = 0; i < 300; i++){
122             requestList.add(createMaintenanceRequest(vehicleList.get(i).id,
equipmentList.get(i).id));
123         }
124         insert requestList;
125
126         for(integer i = 0; i < 300; i++){
127             workPartList.add(createWorkPart(equipmentList.get(i).id,
requestList.get(i).id));
128         }
129         insert workPartList;
130
131         test.startTest();
132         for(case req : requestList){
133             req.Status = CLOSED;
134             oldRequestIds.add(req.Id);
135         }
136         update requestList;
137         test.stopTest();
138
139         list<case> allRequests = [select id
from case
where status =: STATUS_NEW];
140
141         list<Equipment_Maintenance_Item__c> workParts = [select
id
from
Equipment_Maintenance_Item__c
where

```

```

Maintenance_Request__c in: oldRequestIds];
146         system.assert(allRequests.size() == 300);
147     }
148 }

```

MaintenanceRequestHelper.apxc :

```

1  public with sharing class MaintenanceRequestHelper {
2      public static void updateWorkOrders(List<Case> updWorkOrders,
Map<Id,Case> nonUpdCaseMap) {
3          Set<Id> validIds = new Set<Id>();
4
5          For (Case c : updWorkOrders){
6              if (nonUpdCaseMap.get(c.Id).Status != 'Closed' && c.Status ==
'Closed'){
7                  if (c.Type == 'Repair' || c.Type == 'Routine Maintenance'){
8                      validIds.add(c.Id);
9                  }
10             }
11         }
12         if (!validIds.isEmpty()){
13             List<Case> newCases = new List<Case>();
14             Map<Id,Case> closedCasesM = new Map<Id,Case>([SELECT Id,
Vehicle__c, Equipment__c, Equipment__r.Maintenance_Cycle__c,(SELECT
Id,Equipment__c,Quantity__c FROM Equipment_Maintenance_Items__r)
15                                     FROM Case WHERE Id
IN :validIds]);
16             Map<Id,Decimal> maintenanceCycles = new Map<Id,Decimal>();
17             AggregateResult[] results = [SELECT Maintenance_Request__c,
MIN(Equipment__r.Maintenance_Cycle__c)cycle FROM
Equipment_Maintenance_Item__c WHERE Maintenance_Request__c IN :ValidIds
GROUP BY Maintenance_Request__c];
18             for (AggregateResult ar : results){
19                 maintenanceCycles.put((Id) ar.get('Maintenance_Request__c'),
(Decimal) ar.get('cycle'));
20             }
21             for(Case cc : closedCasesM.values()){
22                 Case nc = new Case (
23                     ParentId = cc.Id,
24                     Status = 'New',
25                     Subject = 'Routine Maintenance',
26                     Type = 'Routine Maintenance',

```

```

27         Vehicle__c = cc.Vehicle__c,
28         Equipment__c =cc.Equipment__c,
29         Origin = 'Web',
30         Date_Reported__c = Date.Today()
31     );
32     If (maintenanceCycles.containsKey(cc.Id)){
33         nc.Date_Due__c = Date.today().addDays((Integer)
maintenanceCycles.get(cc.Id));
34     }
35     newCases.add(nc);
36 }
37 insert newCases;
38 List<Equipment_Maintenance_Item__c> clonedWPs = new
List<Equipment_Maintenance_Item__c>();
39 for (Case nc : newCases){
40     for (Equipment_Maintenance_Item__c wp :
closedCasesM.get(nc.ParentId).Equipment_Maintenance_Items__r){
41         Equipment_Maintenance_Item__c wpClone = wp.clone();
42         wpClone.Maintenance_Request__c = nc.Id;
43         ClonedWPs.add(wpClone);
44     }
45 }
46 insert ClonedWPs;
47 }
48 }
49 }

```

MaintenanceRequest.apxt :

```

1 trigger MaintenanceRequest on Case (before update, after update) {
2     if(Trigger.isUpdate && Trigger.isAfter){
3         MaintenanceRequestHelper.updateWorkOrders(Trigger.New,
Trigger.OldMap);
4     }
5 }

```

Task 5 : Test callout logic

Test the external callout logic and achieving 100% code coverage.

WarehouseCalloutService.apxc :

```
1 public with sharing class WarehouseCalloutService {
2     private static final String WAREHOUSE_URL = 'https://th-superbadge-
3
4     //@future(callout=true)
5     public static void runWarehouseEquipmentSync(){
6
7         Http http = new Http();
8         HttpRequest request = new HttpRequest();
9         request.setEndpoint(WAREHOUSE_URL);
10        request.setMethod('GET');
11        HttpResponse response = http.send(request);
12
13        List<Product2> warehouseEq = new List<Product2>();
14
15        if (response.getStatusCode() == 200){
16            List<Object> jsonResponse =
17            (List<Object>)JSON.deserializeUntyped(response.getBody());
18            System.debug(response.getBody());
19
20            for (Object eq : jsonResponse){
21                Map<String,Object> mapJson = (Map<String,Object>)eq;
22                Product2 myEq = new Product2();
23                myEq.Replacement_Part__c = (Boolean)
24                mapJson.get('replacement');
25                myEq.Name = (String) mapJson.get('name');
26                myEq.Maintenance_Cycle__c = (Integer)
27                mapJson.get('maintenanceperiod');
28                myEq.Lifespan_Months__c = (Integer) mapJson.get('lifespan');
29                myEq.Cost__c = (Decimal) mapJson.get('lifespan');
30                myEq.Warehouse_SKU__c = (String) mapJson.get('sku');
31                myEq.Current_Inventory__c = (Double)
32                mapJson.get('quantity');
33                warehouseEq.add(myEq);
34            }
35
36            if (warehouseEq.size() > 0){
37                upsert warehouseEq;
38            }
39        }
40    }
41}
```

```

33         System.debug('Your equipment was synced with the warehouse
34         System.debug(warehouseEq);
35     }
36
37 }
38 }
39 }

```

WarehouseCalloutServiceTest.apxc :

```

1  @isTest
2
3  private class WarehouseCalloutServiceTest {
4      @isTest
5      static void testWareHouseCallout(){
6          Test.startTest();
7          // implement mock callout test here
8          Test.setMock(HTTPCalloutMock.class, new
WarehouseCalloutServiceMock());
9          WarehouseCalloutService.runWarehouseEquipmentSync();
10         Test.stopTest();
11         System.assertEquals(1, [SELECT count() FROM Product2]);
12     } }

```

WarehouseCalloutServiceMock.apxc :

```

1  @isTest
2  global class WarehouseCalloutServiceMock implements HttpCalloutMock {
3      // implement http mock callout
4      global static HttpResponse respond(HttpRequest request){
5
6          System.assertEquals('https://th-superbadge-
));
7          System.assertEquals('GET', request.getMethod());
8

```

```
9      // Create a fake response
10     HttpResponse response = new HttpResponse();
11     response.setHeader('Content-Type', 'application/json');
12     response.setBody(' [{"_id":"55d66226726b611100aaf741","replacement":false,"qu

13     response.setStatusCode(200);
14     return response;
15 }
16 }
```

Task 6 : Test scheduling logic

Test Scheduling logic and 100% code coverage.

WarehouseSyncSchedule.apxc :

```
1 global class WarehouseSyncSchedule implements Schedulable {
2     global void execute(SchedulableContext ctx) {
3
4         WarehouseCalloutService.runWarehouseEquipmentSync();
5     }
6 }
```

WarehouseSyncScheduleTest.apxc :-

```
1 @isTest
2 public class WarehouseSyncScheduleTest {
3
4     @isTest static void WarehousescheduleTest(){
5         String scheduleTime = '00 00 01 * * ?';
6         Test.startTest();
7         Test.setMock(HttpCalloutMock.class, new
WarehouseCalloutServiceMock());
8         String jobID=System.schedule('Warehouse Time To Schedule to Test',
scheduleTime, new WarehouseSyncSchedule());
9         Test.stopTest();
10        CronTrigger a=[SELECT Id FROM CronTrigger where NextFireTime >
today];
11        System.assertEquals(jobID, a.Id,'Schedule ');
12
13
14    }
15 }
```


Process Automation Specialist

(Superbadge)

What You'll Be Doing to Earn This Superbadge

1. Automate lead ownership using assignment rules
2. Enforce data integrity with formula fields and validation rules
3. Create a custom object in a master-detail relationship to a standard object
4. Define an opportunity sales process using stages, record types, and validation rules
5. Automate business processes to send emails, create related records, and submit opportunities for approval
6. Create a flow to display dynamic information on a Lightning record page
7. Create a process to evaluate and update records

Concepts Tested in This Superbadge

- Validations and Formulas
- Sales Process
- Process Builder
- Flow

Task 1 : Automate Leads

Make sure that all leads have the standard 2-character US state abbreviation in the State/Province field and either US, USA, United States, or nothing in the Country field.

Lead Validation Rule

[Back to Lead Validation Rules](#)[Help for this Page](#)

Validation Rule Detail

Edit


Clone

Rule Name	US_state	Active	<input checked="" type="checkbox"/>
Error Condition Formula	OR(AND(LEN(State) > 2, NOT(CONTAINS("AL",AK,AZ,AR,CA,CO,CT,DE,DC,FL,GA,HI,ID,IL,IN,IA,KS,KY,LA,ME,MD,MA,MI,MN,MS,MO,MT,NE,NV,NH,NJ,NM,NY,NC,ND,OH,OK,OR,PA,RI,SC,SD,TN,TX,UT,VT,VA,WA,WV,WI,WY", State))), NOT(OR(Country ="US",Country ="USA",Country ="United States", ISBLANK(Country))))		
Error Message	Not a US state or Wrong Abbreviation.	Error Location	Top of Page
Description			
Created By	Atul Kumar Singh, 6/10/2022, 10:47 PM	Modified By	Atul Kumar Singh, 6/10/2022, 10:47 PM

Edit

Clone

The Rainbow sales team should be handling all the leads that come from the web. A bunch of other leads come from our partners and from lists that we buy--all those leads are potential buyers of our assembly systems. So the Assembly System team should get those leads.

 **SETUP**

Lead Assignment Rules

Lead Assignment Rule

Assign Team

[Help for this Page](#)

Add rule entries that specify the criteria used to route leads. You can reorder rule entries on this page after you create them.

Rule Detail

Edit

Rule Name	Assign Team	Active	<input checked="" type="checkbox"/>
Created By	Atul Kumar Singh, 6/10/2022, 10:51 PM	Modified By	Atul Kumar Singh, 6/10/2022, 10:53 PM

Edit

Rule Entries

New

Reorder

Action	Order	Criteria	Assign To	Email
Edit Del	<input type="text" value="1"/>	Lead: Lead Source EQUALS Web	Rainbow Sales	<input type="checkbox"/>
Edit Del	<input type="text" value="2"/>	Lead: Lead Source NOTEQUAL TO Web	Assembly System Sales	<input type="checkbox"/>

Task 2 : Automate Accounts

Make sure that nobody can save a new account unless the shipping and billing state fields are valid US state abbreviations, and the country field is either blank or US, USA, or United States.

Account Validation Rule

[Back to Account Validation Rules](#)[Help for this Page](#)

Validation Rule Detail

EditClone

Rule NameUS_addressActive✓

Error Condition FormulaOR(AND(LEN(BillingState) > 2, NOT(CONTAINS("AL-AK-AZ-AR-CA-CO-CT-DE-DC-FL-GA-HI-ID-IL-IN-IA-KS-KY-LA-ME-MD-MA-MI-MN-MS-MO-MT-NE-NV-NH-NJ-NM-NY-NC-ND-OH-OK-OR-PA-RI-SC-SD-TN-TX-UT-VT-VA-WA-WV-WI-WY", BillingState))) AND(LEN(ShippingState) > 2, NOT(CONTAINS("AL-AK-AZ-AR-CA-CO-CT-DE-DC-FL-GA-HI-ID-IL-IN-IA-KS-KY-LA-ME-MD-MA-MI-MN-MS-MO-MT-NE-NV-NH-NJ-NM-NY-NC-ND-OH-OK-OR-PA-RI-SC-SD-TN-TX-UT-VT-VA-WA-WV-WI-WY", ShippingState))), NOT(OR(BillingCountry ="US", BillingCountry ="USA", BillingCountry ="United States", ISBLANK(BillingCountry))), NOT(OR(ShippingCountry ="US", ShippingCountry ="USA", ShippingCountry ="United States", ISBLANK(ShippingCountry))))

Error MessageYou cannot save a new account.Error LocationTop of Page

Description

Created ByAtul Kumar Singh, 6/10/2022, 11:11 PMModified ByAtul Kumar Singh, 6/10/2022, 11:11 PM

EditClone

If the account type is either "Customer - Direct" or "Customer - Channel" we don't want anybody to change the name.

Account Validation Rule

[Back to Account Validation Rules](#)[Help for this Page](#)

Validation Rule Detail

EditClone

Rule NameName_ChangeActive✓

Error Condition FormulaISCHANGED(Name) && (OR(ISPICKVAL(Type , 'Customer - Direct') , ISPICKVAL(Type , 'Customer - Channel')))

Error MessageYou cannot change the account name for customer direct and customer channel.Error LocationAccount Name

Description

Created ByAtul Kumar Singh, 6/10/2022, 11:12 PMModified ByAtul Kumar Singh, 6/10/2022, 11:12 PM

EditClone

Task 3 : Create Robot Setup Object

- For every sale we close and win, we need a setup record created that we can use for scheduling. If we delete a deal for any reason the setup record should also be deleted. We need to see the setup date and we also need a spot for the technician or the rep to make notes about the setup.
- The records can just be auto-numbered--they don't need the name of the account or anything.

SETUP > OBJECT MANAGER

Robot Setup

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Restriction Rules

Scoping Rules

Triggers

Details

Description

API Name

Robot_Setup__c

Custom

✓

Singular Label

Robot Setup

Plural Label

Robot Setups

Enable Reports

Track Activities

Track Field History

Deployment Status

Deployed

Help Settings

Standard salesforce.com Help Window

EditDelete

Fields & Relationships

7 Items, Sorted by Field Label

Quick Find

NewDeleted FieldsField DependenciesSet History Tracking

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Date	Date__c	Date		
Day of the Week	Day_of_the_Week__c	Formula (Text)		
Last Modified By	LastModifiedById	Lookup(User)		
Notes	Notes__c	Text Area(255)		
Opportunity	Opportunity__c	Master-Detail(Opportunity)		✓
Robot Setup Name	Name	Auto Number		✓

Task 4 : Create Sales Process and Validate Opportunities

The biggest deals--anything over \$100K--have to be approved before they can close. We should have a way to show on the record that the deal is approved--maybe you can add an "Approved" checkbox to the opportunity. Only system administrators like you and sales managers should be able to check it.

Opportunity Validation Rule

[Back to Opportunity Validation Rules](#) [Help for this Page](#)

Validation Rule Detail

EditClone

Rule Name	Approved_Status	Active	<input checked="" type="checkbox"/>
Error Condition Formula	IF((Amount > 100000 && Approved__c <> True && ISPICKVAL(StageName,'Closed Won')),True,False)		
Error Message	Approval for opp above 100000 need to be approved	Error Location	Top of Page
Description			
Created By	Atul Kumar Singh, 6/11/2022, 12:02 AM	Modified By	Atul Kumar Singh, 6/11/2022, 12:02 AM

EditClone

Task 5 : Automate Opportunities

Send an email to the finance group whenever an opportunity is created for a prospect or customer account, and at the same time to create a task for the account owner, but only if the account is a prospect. And when we win a deal, we want another email sent to the finance group.

SETUP

Approval Processes

Process Definition Detail

Edit

Clone

Deactivate

Process Name	Approval Status to Approved	Active	✓
Unique Name	Approval_Status_to_Approved	Next Automated Approver Determined By	Manager of Record Owner
Description			
Entry Criteria	(Opportunity: Stage EQUALS Negotiation/Review) AND (Opportunity: Amount GREATER THAN 100000)		
Record Editability	Administrator OR Current Approver	Allow Submitters to Recall Approval Requests	<input type="checkbox"/>
Approval Assignment Email Template	SALES: Opportunity Needs Approval		
Initial Submitters	User: Nushi Davoud, Opportunity Owner		
Created By	Atul Kumar Singh	Modified By	Atul Kumar Singh
	6/11/2022, 12:16 AM		6/16/2022, 11:23 PM

Initial Submission Actions

Add Existing

Add New

Action	Type	Description
	Record Lock	Lock the record from being edited
<div>Edit</div>	Field Update	Update Stage

Approval Steps

Show Actions

Edit

Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
	1	Approval Status Negotiations			User: Nushi Davoud	Final Rejection

Final Approval Actions

Add Existing

Add New

Action	Type	Description
<div>Edit</div>	Record Lock	Lock the record from being edited
<div>Edit</div> <div>Remove</div>	Field Update	Approved Checked
<div>Edit</div> <div>Remove</div>	Email Alert	Opp Approved
<div>Edit</div> <div>Remove</div>	Field Update	Update Stage

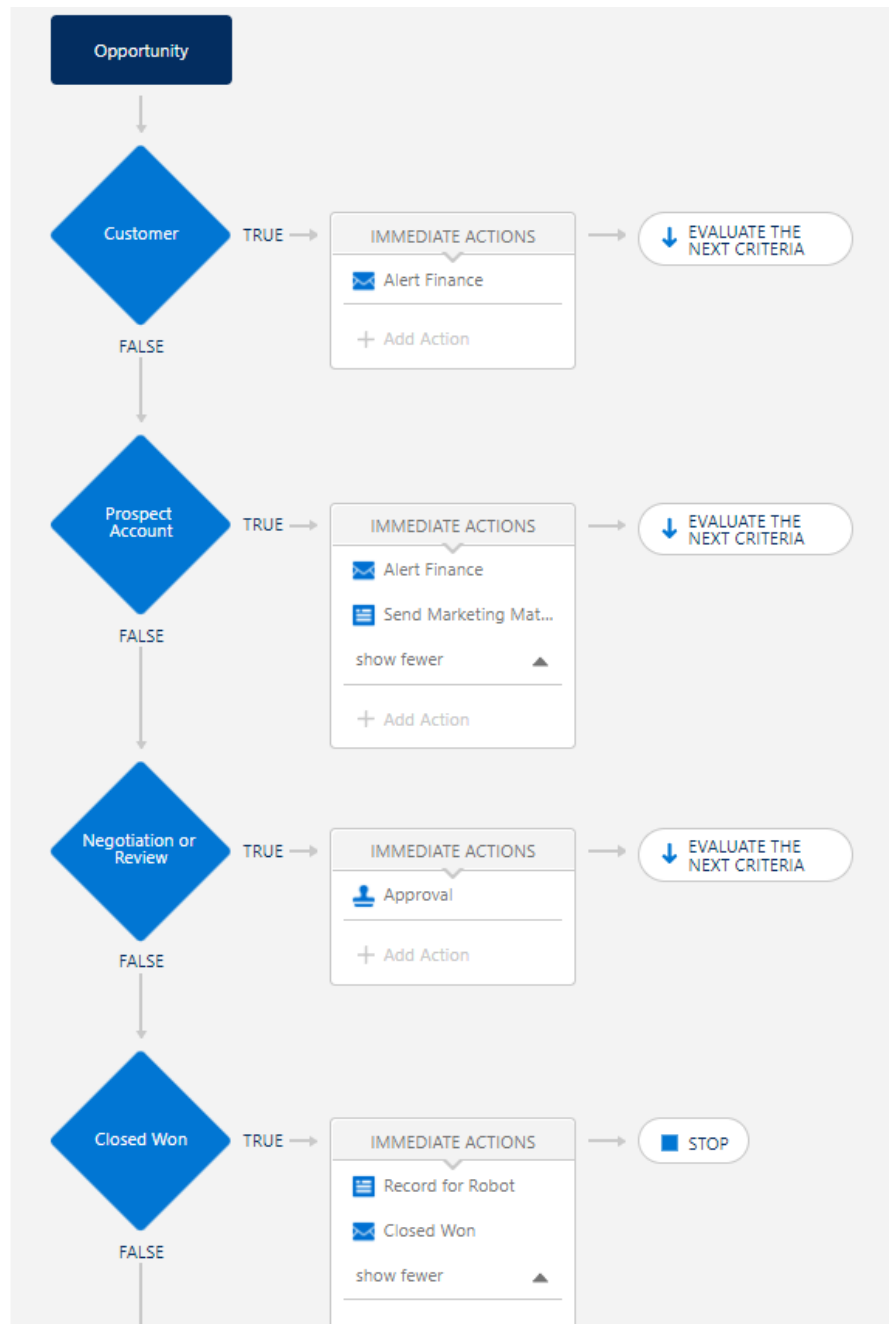
Final Rejection Actions

Add Existing

Add New

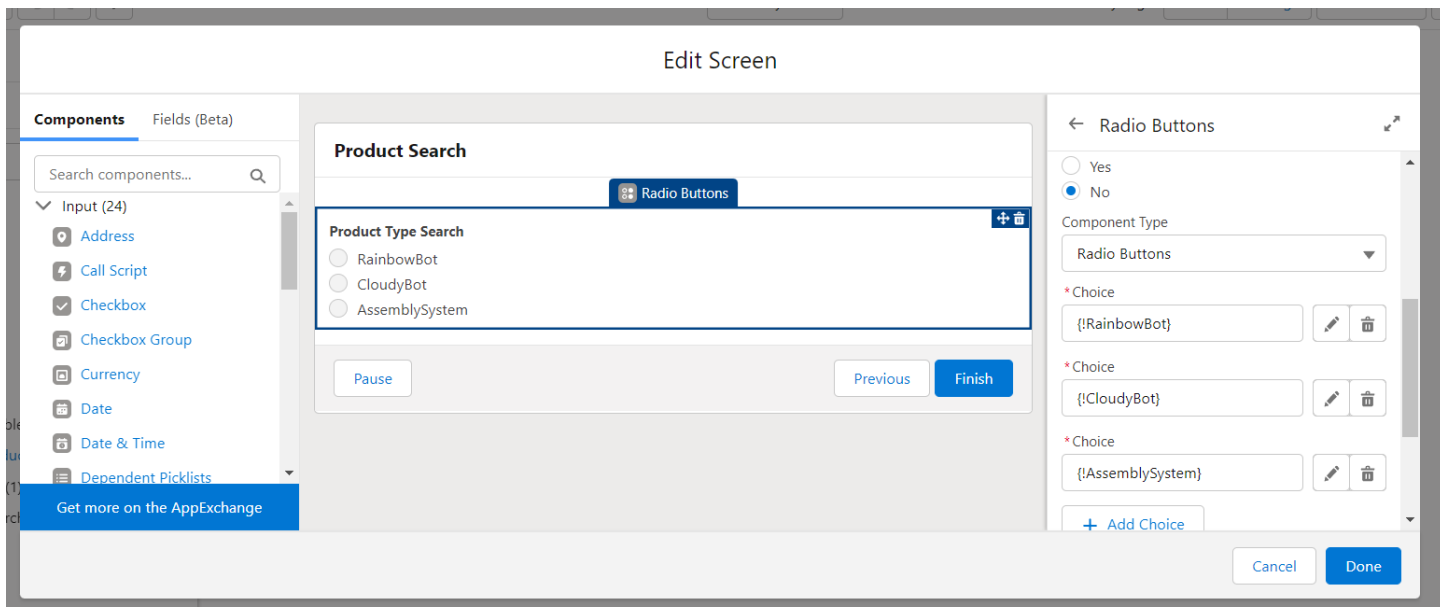
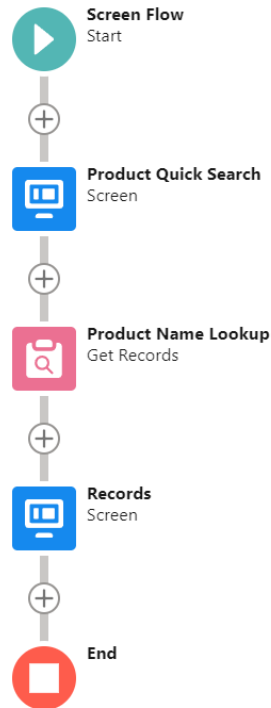
Action	Type	Description
<div>Edit</div>	Record Lock	Unlock the record for editing
<div>Edit</div> <div>Remove</div>	Email Alert	Opp Rejected
<div>Edit</div> <div>Remove</div>	Field Update	Update Stage

The opportunity owner's manager has to approve all deals over \$100,000. At this point, it should be in the Awaiting Approval stage. If the manager approves, the opportunity goes to the Closed/Won stage. If the manager doesn't approve, the opportunity should go back to Negotiation/Review stage. Changes to the record are OK at that point.



Task 6 : Create Flow for Opportunities

We need some kind of product quick search thingy or widget on the page that lets reps filter products on the type of robot they want to buy and see a list right on the opportunity record.



Task 7 : Automate Setups

We need to make sure that any robot setup date that would fall on Saturday or Sunday is set to the following Monday instead.

