SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

# APEX SPECIALIST SUPER BADGE CODES APEX TRIGGERS

### AccountAddressTrigger.axpt:

trigger AccountAddressTrigger on Account (before insert,before update) {

for(Account account:Trigger.New){ if(account.Match\_Billing\_Address c == True){ account.ShippingPostalCode = account.BillingPostalCode;

}

}

}

### ClosedOpportunityTrigger.axpt:

trigger ClosedOpportunityTrigger on Opportunity (after insert,after update) {

List<Task> tasklist = new List<Task>(); for(Opportunity opp: Trigger.New){ if(opp.StageName == 'Closed Won'){

tasklist.add(new Task(Subject = 'Follow Up Test Task',WhatId = opp.Id));

}

}

if(tasklist.size() > 0){ insert tasklist;

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

# APEX TESTING

### VerifyData.apxc:

public class VerifyDate {

public static Date CheckDates(Date date1, Date date2) { if(DateWithin30Days(date1,date2)) {

return date2;

} else {

return SetEndOfMonthDate(date1);

}

}

@TestVisible private static Boolean DateWithin30Days(Date date1, Date date2) {

//check for date2 being in the past if( date2 < date1) { return false; }

//check that date2 is within (>=) 30 days of date1

Date date30Days = date1.addDays(30); //create a date 30 days away from date1

if( date2 >= date30Days ) { return false; } else { return true; }

}

//method to return the end of the month of a given date @TestVisible private static Date SetEndOfMonthDate(Date date1) { Integer totalDays = Date.daysInMonth(date1.year(), date1.month());

Date lastDay = Date.newInstance(date1.year(), date1.month(), totalDays);

return lastDay;

}

}

TestVerifyData.apxc: @isTest

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 3

private class TestVerifyDate {

@isTest static void Test\_CheckDates\_case1(){ Date D =

VerifyDate.CheckDates(date.parse('01/01/2022'),date.parse('01/05/ System.assertEquals(date.parse('01/05/2022'), D);

}

@isTest static void Test\_CheckDates\_case2(){

Date D = VerifyDate.CheckDates(date.parse('01/01/2022'), date.parse('05/05/2022')); System.assertEquals(date.parse('01/31/2022'), D);

}

@isTest static void Test\_Within30Days\_case1(){ Boolean ﬂag = VerifyDate.DateWithin30Days(date.parse('01/01/2022'), date.parse('12/30/2021'));

System.assertEquals(false, ﬂag);

}

@isTest static void Test\_Within30Days\_case2(){ Boolean ﬂag = VerifyDate.DateWithin30Days(date.parse('01/01/2022'), date.parse('02/02/2021'));

System.assertEquals(false, ﬂag);

}

@isTest static void Test\_Within30Days\_case3(){

Boolean ﬂag = VerifyDate.DateWithin30Days(date.parse('01/01/2022'), date.parse('01/15/2022'));

System.assertEquals(true, ﬂag);

}

@isTest static void Test\_SetEndOfMonthDate(){ Date returndate =

VerifyDate.SetEndOfMonthDate(date.parse('01/01/2022'));

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 4

### RestrictContactByName.apxt:

trigger RestrictContactByName on Contact (before insert, before update) {

//check contacts prior to insert or update for invalid data For (Contact c : Trigger.New) {

if(c.LastName == 'INVALIDNAME') { //invalidname is invalid c.AddError('The Last Name "'+c.LastName+'" is not allowed for

}

}

}

### TestRestrictContactByName.apxc:

@isTest

private class TestRestrictContactByName { @isTest static void Test\_insertupdateContact(){ Contact cnt = new Contact();

cnt.LastName = 'INVALIDNAME'; Test.startTest();

Database.SaveResult result = Database.insert(cnt,false); Test.stopTest();

System.assert(!result.isSuccess()); System.assert(result.getErrors().size() > 0); System.assertEquals('The Last Name "INVALIDNAME" is not allowed result.getErrors()[0].getMessage());

}

}

### RandomContactFactory.apxc:

public class RandomContactFactory {

public static List<Contact> generateRandomContacts(Integer num\_cnts, string lastname) {

List<Contact> contacts = new List<Contact>(); for(Integer i = 0; i < num\_cnts; i++) {

Contact cnt = new Contact(FirstName = 'Test' +i,LastName =

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 5

lastname); contacts.add(cnt);

}

return contacts;

}

}

# ASYNCHRONOUS APEX

### AccountProcessor.apxc:

public class AccountProcessor { @future

public static void countContacts(List<Id> accountId\_lst) {

Map<Id,Integer> account\_cno = new Map<Id,Integer>();

List<account> account\_lst\_all = new List<account>([select id, (select id from contacts) from account]);

for(account a:account\_lst\_all) { account\_cno.put(a.id,a.contacts.size()); //populate the map

}

List<account> account\_lst = new List<account>(); / list of account that we will upsert

for(Id accountId : accountId\_lst) { if(account\_cno.containsKey(accountId)) {

account acc = new account(); acc.Id = accountId;

acc.Number\_of\_Contacts c = account\_cno.get(accountId); account\_lst.add(acc);

}

}

upsert account\_lst;

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 6

}

@isTest

### AccountProcessorTest.apxc:

public class AccountProcessorTest {

@isTest

public static void testFunc() { account acc = new account(); acc.name = 'MATW INC'; insert acc;

contact con = new contact(); con.lastname = 'Mann1'; con.AccountId = acc.Id; insert con;

contact con1 = new contact(); con1.lastname = 'Mann2'; con1.AccountId = acc.Id; insert con1;

List<Id> acc\_list = new List<Id>(); acc\_list.add(acc.Id); Test.startTest();

AccountProcessor.countContacts(acc\_list); Test.stopTest();

List<account> acc1 = new List<account>([select Number\_of\_Contacts c from account where id = :acc.id]);

system.assertEquals(2,acc1[0].Number\_of\_Contacts c);

}

}

### LeadProcessor.apxc:

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 7

global class LeadProcessor implements Database.Batchable<sObject> { global Integer count = 0;

global Database.QueryLocator start (Database.BatchableContext bc) { return Database.getQueryLocator('Select Id, LeadSource from lead');

}

global void execute (Database.BatchableContext bc,List<Lead> l\_lst) { List<lead> l\_lst\_new = new List<lead>();

for(lead l : l\_lst) {

l.leadsource = 'Dreamforce'; l\_lst\_new.add(l);

count+=1;

}

update l\_lst\_new;

}

global void ﬁnish (Database.BatchableContext bc) { system.debug('count = '+count);

}

}

### LeadProcessorTest.apxc:

@isTest

public class LeadProcessorTest {

@isTest

public static void testit() {

List<lead> l\_lst = new List<lead>(); for (Integer i = 0; i<200; i++) {

Lead l = new lead(); l.LastName = 'name'+i; l.company = 'company'; l.Status = 'somestatus'; l\_lst.add(l);

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 8

}

insert l\_lst; test.startTest();

Leadprocessor lp = new Leadprocessor(); Id batchId = Database.executeBatch(lp); Test.stopTest();

}

}

### AddPrimaryContact.apxc:

public class AddPrimaryContact implements Queueable { public contact c;

public String state;

public AddPrimaryContact(Contact c, String state) { this.c = c;

this.state = state;

}

public void execute(QueueableContext qc) { system.debug('this.c = '+this.c+' this.state = '+this.state); List<Account> acc\_lst = new List<account>([select id, name,

BillingState from account where account.BillingState = :this.state limit 200]); List<contact> c\_lst = new List<contact>();

for(account a: acc\_lst) { contact c = new contact();

c = this.c.clone(false, false, false, false); c.AccountId = a.Id;

c\_lst.add(c);

}

insert c\_lst;

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 9

}

}

@IsTest

### AddPrimaryContactTest.apxc:

public class AddPrimaryContactTest {

@IsTest

public static void testing() {

List<account> acc\_lst = new List<account>(); for (Integer i=0; i<50;i++) {

account a = new account(name=string.valueOf(i),billingstate='NY'); system.debug('account a = '+a);

acc\_lst.add(a);

}

for (Integer i=0; i<50;i++) { account a = new

account(name=string.valueOf(50+i),billingstate='CA'); system.debug('account a = '+a); acc\_lst.add(a);

}

insert acc\_lst; Test.startTest();

contact c = new contact(lastname='alex'); AddPrimaryContact apc = new AddPrimaryContact(c,'CA'); system.debug('apc = '+apc);

System.enqueueJob(apc); Test.stopTest();

List<contact> c\_lst = new List<contact>([select id from contact]); Integer size = c\_lst.size();

system.assertEquals(50, size);

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

### DailyLeadProcessor.apxc:

public class DailyLeadProcessor implements schedulable{ public void execute(schedulableContext sc) {

List<lead> l\_lst\_new = new List<lead>();

List<lead> l\_lst = new List<lead>([select id, leadsource from lead where leadsource = null]);

for(lead l : l\_lst) {

l.leadsource = 'Dreamforce'; l\_lst\_new.add(l);

}

update l\_lst\_new;

}

}

### DailyLeadProcessorTest.apxc:

@isTest

public class DailyLeadProcessorTest {

@isTest

public static void testing() {

List<lead> l\_lst = new List<lead>(); for(Integer i=0;i<200;i++) {

lead l = new lead(); l.lastname = 'lastname'+i; l.Company = 'company'+i; l\_lst.add(l);

}

insert l\_lst;

Test.startTest();

DailyLeadProcessor dlp = new DailyLeadProcessor ();

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

String jobId = System.Schedule('dailyleadprocessing','0 0 0 1 12 ?

2016',dlp);

Test.stopTest();

List<lead> l\_lst\_chk = new List<lead>([select id,leadsource from lead where leadsource != 'Dreamforce']);

System.assertequals(0,l\_lst\_chk.size());

}

}

# APEX INTEGRATION SERVICES

### AnimalLocator.apxc:

public class AnimalLocator { public class cls\_animal {

public Integer id; public String name; public String eats; public String says;

}

public class JSONOutput{ public cls\_animal animal;

//public JSONOutput parse(String json){

//return (JSONOutput) System.JSON.deserialize(json, JSONOutput.class);

//}

}

public static String getAnimalNameById (Integer id) { Http http = new Http();

HttpRequest request = new HttpRequest(); request.setEndpoint('https://th-apex-http-

callout.herokuapp.com/animals/' + id);

//request.setHeader('id', String.valueof(id)); -- cannot be used in this

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

challenge :)

request.setMethod('GET');

HttpResponse response = http.send(request); system.debug('response: ' + response.getBody());

//Map<String,Object> map\_results = (Map<String,Object>) JSON.deserializeUntyped(response.getBody());

jsonOutput results = (jsonOutput) JSON.deserialize(response.getBody(), jsonOutput.class);

//Object results = (Object) map\_results.get('animal'); system.debug('results= ' + results.animal.name);

return(results.animal.name);

}

}

@IsTest

### AnimalLocatorMock.apxc:

global class AnimalLocatorMock implements HttpCalloutMock {

global HTTPresponse respond(HTTPrequest request) { Httpresponse response = new Httpresponse(); response.setStatusCode(200);

//-- directly output the JSON, instead of creating a logic

//response.setHeader('key, value)

//Integer id = Integer.valueof(request.getHeader('id'));

//Integer id = 1;

//List<String> lst\_body = new List<String> {'majestic badger', 'ﬂuffy bunny'};

//system.debug('animal return value: ' + lst\_body[id]); response.setBody('{"animal":{"id":1,"name":"chicken","eats":"chicken

food","says":"cluck cluck"}}'); return response;

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

### AnimalLocatorTest.apxc:

@IsTest

public class AnimalLocatorTest { @isTest

public static void testAnimalLocator() { Test.setMock(HttpCalloutMock.class, new AnimalLocatorMock());

//Httpresponse response = AnimalLocator.getAnimalNameById(1); String s = AnimalLocator.getAnimalNameById(1); system.debug('string returned: ' + s);

}

}

//Generated by wsdl2apex

### ParkService.apxc:

public class ParkService {

public class byCountryResponse { public String[] return\_x;

private String[] return\_x\_type\_info = new String[]{'return',['http:](http://parks.services/%27%2Cnull%2C%270%27%2C%27-1%27%2C%27false%27)/[/parks.services/',null,'0','-1','false'};](http://parks.services/%27%2Cnull%2C%270%27%2C%27-1%27%2C%27false%27)

private String[] apex\_schema\_type\_info = new [String[]{'http:](http://parks.services/%27%2C%27false%27%2C%27false%27)/[/parks.ser](http://parks.services/%27%2C%27false%27%2C%27false%27)vices/',['false','false](http://parks.services/%27%2C%27false%27%2C%27false%27)'};

private String[] ﬁeld\_order\_type\_info = new String[]{'return\_x'};

}

public class byCountry { public String arg0;

private String[] arg0\_type\_info = new String[]{'arg0',['http:](http://parks.services/%27%2Cnull%2C%270%27%2C%271%27%2C%27false%27)/[/parks.services/',null,'0','1','false'};](http://parks.services/%27%2Cnull%2C%270%27%2C%271%27%2C%27false%27)

private String[] apex\_schema\_type\_info = new [String[]{'http:](http://parks.services/%27%2C%27false%27%2C%27false%27)/[/parks.ser](http://parks.services/%27%2C%27false%27%2C%27false%27)vices/',['false','false](http://parks.services/%27%2C%27false%27%2C%27false%27)'};

private String[] ﬁeld\_order\_type\_info = new String[]{'arg0'};

}

public class ParksImplPort {

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

public String endpoint\_x = 'https://th-apex-soap- service.herokuapp.com/service/parks';

public Map<String,String> inputHttpHeaders\_x; public Map<String,String> outputHttpHeaders\_x; public String clientCertName\_x;

public String clientCert\_x;

public String clientCertPasswd\_x; public Integer timeout\_x;

private String[] ns\_map\_type\_info = new [String[]{'http:](http://parks.services/%27)/[/parks.ser](http://parks.services/%27)vices/', 'ParkService'};

public String[] byCountry(String arg0) {

ParkService.byCountry request\_x = new ParkService.byCountry(); request\_x.arg0 = arg0;

ParkService.byCountryResponse response\_x;

Map<String, ParkService.byCountryResponse> response\_map\_x = new Map<String, ParkService.byCountryResponse>();

response\_map\_x.put('response\_x', response\_x); WebServiceCallout.invoke(

this, request\_x,

response\_map\_x,

new String[]{endpoint\_x, '',

['http:](http://parks.services/%27)/[/parks.services/',](http://parks.services/%27) 'byCountry', ['http:](http://parks.services/%27)/[/parks.services/',](http://parks.services/%27) 'byCountryResponse', 'ParkService.byCountryResponse'}

);

response\_x = response\_map\_x.get('response\_x'); return response\_x.return\_x;

}

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

### ParkLocator.apxc:

public class ParkLocator {

public static String[] country(String country){ ParkService.ParksImplPort parks = new ParkService.ParksImplPort(); String[] parksname = parks.byCountry(country);

return parksname;

}

}

### ParkLocatorTest.apxc:

@isTest

private class ParkLocatorTest{ @isTest

static void testParkLocator() { Test.setMock(WebServiceMock.class, new ParkServiceMock()); String[] arrayOfParks = ParkLocator.country('India');

System.assertEquals('Park1', arrayOfParks[0]);

}

}

### ParkServiceMock.apxc:

@isTest

global class ParkServiceMock implements WebServiceMock { global void doInvoke(

Object stub, Object request,

Map<String, Object> response, String endpoint,

String soapAction, String requestName,

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

String responseNS, String responseName, String responseType) {

ParkService.byCountryResponse response\_x = new ParkService.byCountryResponse();

List<String> lstOfDummyParks = new List<String>

{'Park1','Park2','Park3'};

response\_x.return\_x = lstOfDummyParks;

response.put('response\_x', response\_x);

}

}

### AccountManager.apxc:

@RestResource(urlMapping='/Accounts/\*/contacts') global with sharing class AccountManager {

@HttpGet

global static account getAccount() { RestRequest request = RestContext.request;

String accountId = request.requestURI.substring(request.requestURI.lastIndexOf('/')-18,

request.requestURI.lastIndexOf('/'));

List<Account> a = [select id, name, (select id, name from contacts) from account where id = :accountId];

List<contact> co = [select id, name from contact where account.id =

:accountId];

system.debug('\*\* a[0]= '+ a[0]); return a[0];

}

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

**AccountManagerTest.apxc**: @Istest(SeeAllData=true)

public class AccountManagerTest { @IsTest

public static void testaccountmanager() { RestRequest request = new RestRequest(); request.requestUri = 'https://mannharleen-dev-

ed.my.salesforce.com/services/apexrest/Accounts/00190000016cw4tAAA/c ontacts';

request.httpMethod = 'GET'; RestContext.request = request;

system.debug('test account result = '+ AccountManager.getAccount());

}

}

**Challenge 1**

# APEX SPECIALIST SUPER BADGE

### MaintenanceRequestHelper.apxc:

public with sharing class MaintenanceRequestHelper { public static void updateWorkOrders(List<Case> caseList) { List<case> newCases = new List<Case>(); Map<String,Integer> result=getDueDate(caseList);

for(Case c : caseList){ if(c.status=='closed')

if(c.type=='Repair' || c.type=='Routine Maintenance'){ Case newCase = new Case(); newCase.Status='New';

newCase.Origin='web';

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

newCase.Type='Routine Maintenance'; newCase.Subject='Routine Maintenance of Vehicle'; newCase.Vehicle c=c.Vehicle c; newCase.Equipment c=c.Equipment c; newCase.Date\_Reported c=Date.today(); if(result.get(c.Id)!=null)

newCase.Date\_Due c=Date.today()+result.get(c.Id); else

newCase.Date\_Due c=Date.today(); newCases.add(newCase);

}

}

insert newCases;

}

/

public static Map<String,Integer> getDueDate(List<case> CaseIDs){ Map<String,Integer> result = new Map<String,Integer>();

Map<Id, case> caseKeys = new Map<Id, case> (CaseIDs); List<AggregateResult> wpc=[select Maintenance\_Request r.ID cID,min(Equipment r.Maintenance\_Cycle c)cycle

from Work\_Part c where Maintenance\_Request r.ID in :caseKeys.keySet() group by Maintenance\_Request r.ID ];

for(AggregateResult res :wpc){ Integer addDays=0; if(res.get('cycle')!=null)

addDays+=Integer.valueOf(res.get('cycle')); result.put((String)res.get('cID'),addDays);

}

return result;

}

}

**MaintenanceRequest.apxt:**

trigger MaintenanceRequest on Case (before update, after update) {

/ ToDo: Call MaintenanceRequestHelper.updateWorkOrders

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 1

if(Trigger.isAfter) MaintenanceRequestHelper.updateWorkOrders(Trigger.New);

}

## Challenge 2:

### WarehouseCalloutService.apxt:

public with sharing class WarehouseCalloutService {

private static ﬁnal String WAREHOUSE\_URL = 'https://th-superbadge- apex.herokuapp.com/equipment';

@future(callout=true)

public static void runWarehouseEquipmentSync() {

//ToDo: complete this method to make the callout (using @future) to the

/ REST endpoint and update equipment on hand. HttpResponse response = getResponse(); if(response.getStatusCode() == 200)

{

List<Product2> results = getProductList(response); //get list of products from Http callout response

if(results.size() >0)

upsert results Warehouse\_SKU c; //Upsert the products in your org based on the external ID SKU

}

}

//Get the product list from the external link

public static List<Product2> getProductList(HttpResponse response)

{

List<Object> externalProducts = (List<Object>) JSON.deserializeUntyped(response.getBody()); //desrialize the json response List<Product2> newProducts = new List<Product2>();

for(Object p : externalProducts)

{

Map<String, Object> productMap = (Map<String, Object>) p; Product2 pr = new Product2();

//Map the ﬁelds in the response to the appropriate ﬁelds in the Equipment object

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

pr.Replacement\_Part c = (Boolean)productMap.get('replacement'); pr.Cost c = (Integer)productMap.get('cost');

pr.Current\_Inventory c = (Integer)productMap.get('quantity'); pr.Lifespan\_Months c = (Integer)productMap.get('lifespan') ; pr.Maintenance\_Cycle c = (Integer)productMap.get('maintenanceperiod'); pr.Warehouse\_SKU c = (String)productMap.get('sku');

pr.ProductCode = (String)productMap.get('\_id'); pr.Name = (String)productMap.get('name'); newProducts.add(pr);

}

return newProducts;

}

/ Send Http GET request and receive Http response public static HttpResponse getResponse() {

Http http = new Http();

HttpRequest request = new HttpRequest(); request.setEndpoint(WAREHOUSE\_URL); request.setMethod('GET');

HttpResponse response = http.send(request); return response;

}

}

## Challenge 3:

### WarehouseSyncSchedule.apxt

global class WarehouseSyncSchedule implements Schedulable{

/ implement scheduled code here

global void execute (SchedulableContext sc){ WarehouseCalloutService.runWarehouseEquipmentSync();

//optional this can be done by debug mode String sch = '00 00 01 \* \* ?';//on 1 pm

System.schedule('WarehouseSyncScheduleTest', sch, new WarehouseSyncSchedule());

}

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

}

## Challenge 4:

### MaintenanceRequest.apxt:

trigger MaintenanceRequest on Case (before update, after update) { if(Trigger.isUpdate && Trigger.isAfter) MaintenanceRequestHelper.updateWorkOrders(Trigger.New);

}

### InstallationTests.apxt:

@IsTest

private class InstallationTests {

private static ﬁnal String STRING\_TEST = 'TEST'; private static ﬁnal String NEW\_STATUS = 'New'; private static ﬁnal String WORKING = 'Working'; private static ﬁnal String CLOSED = 'Closed'; private static ﬁnal String REPAIR = 'Repair';

private static ﬁnal String REQUEST\_ORIGIN = 'Web';

private static ﬁnal String REQUEST\_TYPE = 'Routine Maintenance'; private static ﬁnal String REQUEST\_SUBJECT = 'AMC Spirit'; public static String CRON\_EXP = '0 0 1 \* \* ?';

static testmethod void testMaintenanceRequestNegative() { Vehicle c vehicle = createVehicle();

insert vehicle;

Id vehicleId = vehicle.Id;

Product2 equipment = createEquipment(); insert equipment;

Id equipmentId = equipment.Id;

Case r = createMaintenanceRequest(vehicleId, equipmentId); insert r;

Work\_Part c w = createWorkPart(equipmentId, r.Id);

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

insert w; Test.startTest(); r.Status = WORKING; update r; Test.stopTest();

List<case> allRequest = [SELECT Id FROM Case];

Work\_Part c workPart = [SELECT Id FROM Work\_Part c

WHERE Maintenance\_Request c =: r.Id]; System.assert(workPart != null); System.assert(allRequest.size() == 1);

}

static testmethod void testWarehouseSync() { Test.setMock(HttpCalloutMock.class, new WarehouseCalloutServiceMock()); Test.startTest();

String jobId = System.schedule('WarehouseSyncSchedule', CRON\_EXP,

new WarehouseSyncSchedule());

CronTrigger ct = [SELECT Id, CronExpression, TimesTriggered, NextFireTime FROM CronTrigger

WHERE id = :jobId];

System.assertEquals(CRON\_EXP, ct.CronExpression); System.assertEquals(0, ct.TimesTriggered); Test.stopTest();

}

private static Vehicle c createVehicle() {

Vehicle c v = new Vehicle c(Name = STRING\_TEST); return v;

}

private static Product2 createEquipment() { Product2 p = new Product2(Name = STRING\_TEST, Lifespan\_Months c = 10,

Maintenance\_Cycle c = 10,

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

Replacement\_Part c = true); return p;

}

private static Case createMaintenanceRequest(Id vehicleId, Id equipmentId)

{

Case c = new Case(Type = REPAIR, Status = NEW\_STATUS,

Origin = REQUEST\_ORIGIN, Subject = REQUEST\_SUBJECT,

Equipment c = equipmentId, Vehicle c = vehicleId); return c;

}

private static Work\_Part c createWorkPart(Id equipmentId, Id requestId) { Work\_Part c wp = new Work\_Part c(Equipment c = equipmentId, Maintenance\_Request c = requestId);

return wp;

}

}

### MaintenanceRequestHelper.apxt:

public with sharing class MaintenanceRequestHelper { public static void updateWorkOrders(List<case> caseList) { List<case> newCases = new List<case>(); Map<String,Integer> result=getDueDate(caseList); for(Case c : caseList){

if(c.status=='closed')

if(c.type=='Repair' || c.type=='Routine Maintenance'){ Case newCase = new Case(); newCase.Status='New';

newCase.Origin='web'; newCase.Type='Routine Maintenance';

newCase.Subject='Routine Maintenance of Vehicle'; newCase.Vehicle c=c.Vehicle c; newCase.Equipment c=c.Equipment c;

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

newCase.Date\_Reported c=Date.today(); if(result.get(c.Id)!=null)

newCase.Date\_Due c=Date.today()+result.get(c.Id); else

newCase.Date\_Due c=Date.today(); newCases.add(newCase);

}

}

insert newCases;

}

/

public static Map<String,Integer> getDueDate(List<case> CaseIDs){ Map<String,Integer> result = new Map<String,Integer>();

Map<Id, case> caseKeys = new Map<Id, case> (CaseIDs); List<aggregateresult> wpc=[select Maintenance\_Request r.ID cID,min(Equipment r.Maintenance\_Cycle c)cycle

from Work\_Part c where Maintenance\_Request r.ID in :caseKeys.keySet() group by Maintenance\_Request r.ID ];

for(AggregateResult res :wpc){ Integer addDays=0; if(res.get('cycle')!=null)

addDays+=Integer.valueOf(res.get('cycle')); result.put((String)res.get('cID'),addDays);

}

return result;

}

}

### MaintenanceRequestTest.apxt:

@isTest

public class MaintenanceRequestTest {

static List<case> caseList1 = new List<case>();

static List<product2> prodList = new List<product2>();

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

static List<work\_part c> wpList = new List<work\_part c>(); @testSetup

static void getData(){

caseList1= CreateData( 300,3,3,'Repair');

}

public static List<case> CreateData( Integer numOfcase, Integer numofProd, Integer numofVehicle,

String type){

List<case> caseList = new List<case>();

//Create Vehicle

Vehicle c vc = new Vehicle c(); vc.name='Test Vehicle';

upsert vc;

//Create Equiment

for(Integer i=0;i<numofProd;i++){ Product2 prod = new Product2(); prod.Name='Test Product'+i; if(i!=0)

prod.Maintenance\_Cycle c=i; prod.Replacement\_Part c=true; prodList.add(prod);

}

upsert prodlist;

//Create Case

for(Integer i=0;i< numOfcase;i++){ Case newCase = new Case(); newCase.Status='New'; newCase.Origin='web';

if( math.mod(i, 2) ==0) newCase.Type='Routine Maintenance'; else

newCase.Type='Repair';

newCase.Subject='Routine Maintenance of Vehicle' +i; newCase.Vehicle c=vc.Id;

if(i<numofProd)

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

newCase.Equipment c=prodList.get(i).ID; else

newCase.Equipment c=prodList.get(0).ID; caseList.add(newCase);

}

upsert caseList;

for(Integer i=0;i<numofProd;i++){ Work\_Part c wp = new Work\_Part c(); wp.Equipment c =prodlist.get(i).Id ;

wp.Maintenance\_Request c=caseList.get(i).id; wplist.add(wp) ;

}

upsert wplist; return caseList;

}

public static testmethod void testMaintenanceHelper(){ Test.startTest();

getData();

for(Case cas: caseList1) cas.Status ='Closed'; update caseList1; Test.stopTest();

}

}

## Challenge 5:

### WarehouseCalloutServiceTest.apxt:

@IsTest

private class WarehouseCalloutServiceTest {

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

/ implement your mock callout test here @isTest

static void testWareHouseCallout(){ Test.setMock(HttpCalloutMock.class, new WarehouseCalloutServiceMock()); WarehouseCalloutService.runWarehouseEquipmentSync();

}

}

### WarehouseCalloutServiceMock.apxt:

@isTest

public class WarehouseCalloutServiceMock implements HTTPCalloutMock

{

/ implement http mock callout

public HTTPResponse respond (HttpRequest request){ HttpResponse response = new HTTPResponse(); response.setHeader('Content-type','application/json');

response.setBody('[{"\_id":"55d66226726b611100aaf741","replacement":fals e,"quantity":5,"name":"Generator 1000 kW","maintenanceperiod":365,"lifespan":120,"cost":5000,"sku":"100003"},{"\_id"

:"55d66226726b611100aaf742","replacement":true,"quantity":183,"name":"Co oling Fan","maintenanceperiod":0,"lifespan":0,"cost":300,"sku":"100004"},{"\_id":"55d 66226726b611100aaf743","replacement":true,"quantity":143,"name":"Fuse 20A","maintenanceperiod":0,"lifespan":0,"cost":22,"sku":"100005"}]'); response.setStatusCode(200);

return response;

}

}

## Challenge 6:

@isTest

### WarehouseSyncScheduleTest.apxt:

private class WarehouseSyncScheduleTest {

public static String CRON\_EXP = '0 0 0 15 3 ? 2022'; static testmethod void testjob(){

SPSGP-12153-Salesforce Developer Catalyst

Self-Learning & Super Badges 2

MaintenanceRequestTest.CreateData( 5,2,2,'Repair'); Test.startTest(); Test.setMock(HttpCalloutMock.class, new WarehouseCalloutServiceMock());

String joBID= System.schedule('TestScheduleJob', CRON\_EXP, new WarehouseSyncSchedule());

/ List<Case> caselist = [Select count(id) from case where case] Test.stopTest();

}

}