**MaintenanceRequestHelperTest**

@istest

public with sharing class MaintenanceRequestHelperTest {

private static final string STATUS\_NEW = '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 = 'Testing subject';

PRIVATE STATIC Vehicle\_\_c createVehicle(){

Vehicle\_\_c Vehicle = new Vehicle\_\_C(name = 'SuperTruck');

return Vehicle;

}

PRIVATE STATIC Product2 createEq(){

product2 equipment = new product2(name = 'SuperEquipment',

lifespan\_months\_\_C = 10,

maintenance\_cycle\_\_C = 10,

replacement\_part\_\_c = true);

return equipment;

}

PRIVATE STATIC Case createMaintenanceRequest(id vehicleId, id equipmentId){

case cs = new case(Type=REPAIR,

Status=STATUS\_NEW,

Origin=REQUEST\_ORIGIN,

Subject=REQUEST\_SUBJECT,

Equipment\_\_c=equipmentId,

Vehicle\_\_c=vehicleId);

return cs;

}

PRIVATE STATIC Equipment\_Maintenance\_Item\_\_c createWorkPart(id equipmentId,id requestId){

Equipment\_Maintenance\_Item\_\_c wp = new Equipment\_Maintenance\_Item\_\_c(Equipment\_\_c = equipmentId,

Maintenance\_Request\_\_c = requestId);

return wp;

}

@istest

private static void testMaintenanceRequestPositive(){

Vehicle\_\_c vehicle = createVehicle();

insert vehicle;

id vehicleId = [vehicle.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbHdOLUZBR05GdmNRZDAtOHNodmlUYlVzUG1qZ3xBQ3Jtc0ttd29tTFhuSE9BVnc3WUhxNEE2MTR4Rng5QzkxYlpkVERHZEpCTG93YVlrNVRxTmtZZjdGejV1cnRCeVBJNGdvV3B2Q29vZGgtY29wODA4VlVMdWpOeFFNREIyYTVZVXBtZ1hPZXB6Wk9qRG5heWF4QQ&q=http%3A%2F%2Fvehicle.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg);

Product2 equipment = createEq();

insert equipment;

id equipmentId = [equipment.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbFZPZE9KZ3ZUVTN4Z1I5MGpTTkxjeXExMmdYQXxBQ3Jtc0tsNUE0Q0cyRTZsSzktSW5JZERIMElDNUVrUDFtczBxeXBNTy1mUmFhMFozQ0ltVDJnc1pxajF3em5NeC1yR0o1OHlib2d5dTB2RXdwNWhDRHF5bTBuRldkRTFZZnpHbzVJX2xvRUM0cjZ0UEJLVF95aw&q=http%3A%2F%2Fequipment.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg);

case somethingToUpdate = createMaintenanceRequest(vehicleId,equipmentId);

insert somethingToUpdate;

Equipment\_Maintenance\_Item\_\_c workP = createWorkPart(equipmentId,somethingToUpdate.id);

insert workP;

test.startTest();

somethingToUpdate.status = CLOSED;

update somethingToUpdate;

test.stopTest();

Case newReq = [Select id, subject, type, Equipment\_\_c, Date\_Reported\_\_c, Vehicle\_\_c, Date\_Due\_\_c

from case

where status =:STATUS\_NEW];

Equipment\_Maintenance\_Item\_\_c workPart = [select id

from Equipment\_Maintenance\_Item\_\_c

where Maintenance\_Request\_\_c =:newReq.Id];

system.assert(workPart != null);

system.assert(newReq.Subject != null);

system.assertEquals(newReq.Type, REQUEST\_TYPE);

SYSTEM.assertEquals(newReq.Equipment\_\_c, equipmentId);

SYSTEM.assertEquals(newReq.Vehicle\_\_c, vehicleId);

SYSTEM.assertEquals(newReq.Date\_Reported\_\_c, system.today());

}

@istest

private static void testMaintenanceRequestNegative(){

Vehicle\_\_C vehicle = createVehicle();

insert vehicle;

id vehicleId = [vehicle.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbkFJeEYxc1V6VTFWdzdNVmVlMVBpVWkzeUtrZ3xBQ3Jtc0ttWEtnaGQxN2pjTDMyTFVpRDRFeUFLY040V0JCT1o0MUNmaEk5cGI5RWRxUEZicG1MeUw0UVVnZWxlNlZ1RERUUTVlUjc2R0RrNFZOX2t1eHhLYTNuOTMtU2UxVXlUS2d3YTNmeV9YZ3Z2b2tPeUFIZw&q=http%3A%2F%2Fvehicle.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg);

product2 equipment = createEq();

insert equipment;

id equipmentId = [equipment.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbi1JSE5jemRYamR2dzQ2NTB0TWdiaDhHQy04UXxBQ3Jtc0tsMnNta0pFT2FaV3JOMURGZlpKRkJsY1Q3M1hoOFNKZVV3Ty1xdkNEV2RkSXFLWUZnTWh2U1QxSnhVeFhLRlU2YTV3M3dlOTY3QWtIekpOODRfbWtEZmJaaDliT05kTDBiOFM3N1FGcndTeW91anE3VQ&q=http%3A%2F%2Fequipment.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg);

case emptyReq = createMaintenanceRequest(vehicleId,equipmentId);

insert emptyReq;

Equipment\_Maintenance\_Item\_\_c workP = createWorkPart(equipmentId, emptyReq.Id);

insert workP;

test.startTest();

emptyReq.Status = WORKING;

update emptyReq;

test.stopTest();

list<case> allRequest = [select id

from case];

Equipment\_Maintenance\_Item\_\_c workPart = [select id

from Equipment\_Maintenance\_Item\_\_c

where Maintenance\_Request\_\_c = :emptyReq.Id];

system.assert(workPart != null);

system.assert(allRequest.size() == 1);

}

@istest

private static void testMaintenanceRequestBulk(){

list<Vehicle\_\_C> vehicleList = new list<Vehicle\_\_C>();

list<Product2> equipmentList = new list<Product2>();

list<Equipment\_Maintenance\_Item\_\_c> workPartList = new list<Equipment\_Maintenance\_Item\_\_c>();

list<case> requestList = new list<case>();

list<id> oldRequestIds = new list<id>();

for(integer i = 0; i < 300; i++){

vehicleList.add(createVehicle());

equipmentList.add(createEq());

}

insert vehicleList;

insert equipmentList;

for(integer i = 0; i < 300; i++){

requestList.add(createMaintenanceRequest(vehicleList.get(i).id, equipmentList.get(i).id));

}

insert requestList;

for(integer i = 0; i < 300; i++){

workPartList.add(createWorkPart(equipmentList.get(i).id, requestList.get(i).id));

}

insert workPartList;

test.startTest();

for(case req : requestList){

req.Status = CLOSED;

oldRequestIds.add(req.Id);

}

update requestList;

test.stopTest();

list<case> allRequests = [select id

from case

where status =: STATUS\_NEW];

list<Equipment\_Maintenance\_Item\_\_c> workParts = [select id

from Equipment\_Maintenance\_Item\_\_c

where Maintenance\_Request\_\_c in: oldRequestIds];

system.assert(allRequests.size() == 300);

}

}

**MaintenanceRequestHelper**

public with sharing class MaintenanceRequestHelper {

public static void updateworkOrders(List<Case> updWorkOrders, Map<Id,Case> nonUpdCaseMap) {

Set<Id> validIds = new Set<Id>();

For (Case c : updWorkOrders){

if (nonUpdCaseMap.get(c.Id).Status != 'Closed' && c.Status == 'Closed'){

if (c.Type == 'Repair' || c.Type == 'Routine Maintenance'){

validIds.add(c.Id);

}

}

}

if (!validIds.isEmpty()){

List<Case> newCases = new List<Case>();

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)

FROM Case WHERE Id IN :validIds]);

Map<Id,Decimal> maintenanceCycles = new Map<ID,Decimal>();

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];

for (AggregateResult ar : results){

maintenanceCycles.put((Id) ar.get('Maintenance\_Request\_\_c'), (Decimal) ar.get('cycle'));

}

for(Case cc : closedCasesM.values()){

Case nc = new Case (

ParentId = [cc.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqa0RiR2ZNeGNtYkJXelRVUUdzd2ZaYzVGNXNlQXxBQ3Jtc0trLUo3cWZkVGpiNUdhY253dzNkbzNJOWVJajhjeEMyVmY3RW9PRVh3X1ZfSklQdmZ3b3JvNzA1UWhEQXg1cU1vbDdmX0ZyUWhXVm1yN1RNVUFnb1F3Q3czN3pvQ0lNeVYxWU81SzlJN2tVR05Sekg4NA&q=http%3A%2F%2Fcc.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg),

Status = 'New',

Subject = 'Routine Maintenance',

Type = 'Routine Maintenance',

Vehicle\_\_c = cc.Vehicle\_\_c,

Equipment\_\_c =cc.Equipment\_\_c,

Origin = 'Web',

Date\_Reported\_\_c = Date.Today()

);

If (maintenanceCycles.containskey(cc.Id)){

nc.Date\_Due\_\_c = Date.today().addDays((Integer) maintenanceCycles.get(cc.Id));

}

newCases.add(nc);

}

insert newCases;

List<Equipment\_Maintenance\_Item\_\_c> clonedWPs = new List<Equipment\_Maintenance\_Item\_\_c>();

for (Case nc : newCases){

for (Equipment\_Maintenance\_Item\_\_c wp : closedCasesM.get(nc.ParentId).Equipment\_Maintenance\_Items\_\_r){

Equipment\_Maintenance\_Item\_\_c wpClone = wp.clone();

wpClone.Maintenance\_Request\_\_c = [nc.Id](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbUc2UFpld2NEeG5mTFVpZVNsZnVVc1ZSVGZUZ3xBQ3Jtc0tudHJ2VWp4S3lKRHl4YVZWM0p2R2VCTHhzT1N3bWo2OG45TXNsTnQ4bWt6eHNJb01Id0loVGlEbFp1ck00NG5VQ0paRzJraE5MQVhWWlpNUmdJRGM2UV9BMzQxakRQZ0tybFhYNGVtLUtySENkX1pWTQ&q=http%3A%2F%2Fnc.id%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg);

ClonedWPs.add(wpClone);

}

}

insert ClonedWPs;

}

}

}

**MaintenanceRequest**

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

if(Trigger.isUpdate && Trigger.isAfter){

MaintenanceRequestHelper.updateWorkOrders([Trigger.New](https://www.youtube.com/redirect?event=comments&redir_token=QUFFLUhqbE5wY3VQb2dmM3Q2VTZkdjRuTGw3ajQyRkFpUXxBQ3Jtc0ttbkdlX01Jd3BwU0libko1T1ZGSkFmUmZfVGVBNUNOZGhoM1ZLejIxcE4yU1ZxZW0tTUs3TTdLeDZTWkJvZG9fRnltS1R3akpKRmJqR2gzX2JXSEtVSVJBeTh4OE5MUFJ2MEg0Z3ZnNXk4R2lxclgwbw&q=http%3A%2F%2Ftrigger.new%2F&stzid=UgzLbrAdydQfYqkhoUR4AaABAg), Trigger.OldMap);

}

}