

# Evidencia de reparación de bugs con TDD

Para cada uno de los bug que elegimos con el equipo, vamos a especificar:

- Código de software reparado.
- Código de casos de prueba
- Evidencia de ejecución de casos de prueba

## Introducir un número de trackeo inexistente muestra un mensaje de error del backend

[Issue](#)

### Código de software reparado

#### Backend:

```
public Purchase GetPurchaseByTrackingCode(string trackingCode)
{
    if (string.IsNullOrEmpty(trackingCode))
        throw new InvalidResourceException($"Tracking Code is can't be empty");

    Purchase trackedPurchase = _purchasesRepository.GetOneDetailByExpression(p =>
p.TrackingCode == trackingCode);
    if (trackedPurchase == null)
        throw new InvalidResourceException($"Tracking code does not exist");

    return trackedPurchase;
}
```

#### Frontend:

```
getPurchaseByTrackingCode(code: string): Observable<PurchaseResponse> {
    const url = `${this.url}/tracking?code=${code}`;
    return this.http.get<PurchaseResponse>(url, {headers: this.getHttpHeaders() })
    .pipe(
        tap(),
        catchError(this.handleError<PurchaseResponse>
            (`Get purchase by tracking code "${code}"`))
    );
}
```

## Código de casos de prueba

```
[TestMethod]
[ExpectedException(typeof(InvalidResourceException))]
public void Get_Tracking_Invalid_Code_InvalidResourceException()
{
    string invalidTrackingCode = "InvalidTrackingCode";
    _purchaseRepository
        .Setup(service => service.GetOneDetailByExpression(p => p.TrackingCode ==
invalidTrackingCode))
        .Returns((Purchase)null);

    var result = _purchasesManager.GetPurchaseByTrackingCode(invalidTrackingCode);
}
```

## Evidencia de ejecución de casos de prueba

### Fase Roja:

Serie de pruebas finalizada: 292 pruebas (Superadas: 291; Con errores: 1; Omitidas: 0) ejecutadas en 3 s

Prueba	Duración	Rasgos	M
✗ Get_Tracking_Invalid_Code_InvalidResourceException	1 ms		T...
✓ Reject_Purchase_Fail_Drug_Not_Found	1 ms		
✓ Reject_Purchase_Fail_Pharmacy_Not_Found	< 1 ms		
✓ Reject_Purchase_Fail_Purchase_Detail_Not_Found	< 1 ms		
✓ Reject_Purchase_Fail_Purchase_Not_Found	< 1 ms		
✓ Reject_Purchase_Ok	3 ms		
✓ RoleManagerTest (1)	2 ms		
✓ StockManagerTest (25)	28 ms		
✓ UnitMeasureManagerTest (1)	1 ms		
✓ UsersManagerTests (12)	7 ms		

**Resumen de los detalles de la prueba**

✗ Get\_Tracking\_Invalid\_Code\_InvalidResourceException  
Origen: PurchasesManagerTests.cs línea 693  
Duración: 1 ms

Mensaje:  
Test method did not throw expected exception PharmaGo.Exceptions.InvalidResourceException.

### Fase verde:

Serie de pruebas finalizada: 292 pruebas (Superadas: 292; Con errores: 0; Omitidas: 0) ejecutadas en 2,2 s

Prueba	Duración	Rasgos	M
✓ Get_Tracking_Invalid_Code_InvalidResourceException	< 1 ms		
✓ Reject_Purchase_Fail_Drug_Not_Found	1 ms		
✓ Reject_Purchase_Fail_Pharmacy_Not_Found	< 1 ms		
✓ Reject_Purchase_Fail_Purchase_Detail_Not_Found	< 1 ms		
✓ Reject_Purchase_Fail_Purchase_Not_Found	< 1 ms		
✓ Reject_Purchase_Ok	2 ms		
✓ RoleManagerTest (1)	1 ms		
✓ StockManagerTest (25)	32 ms		
✓ UnitMeasureManagerTest (1)	2 ms		
✓ UsersManagerTests (12)	10 ms		

**Resumen de los detalles de la prueba**

✓ Get\_Tracking\_Invalid\_Code\_InvalidResourceException  
Origen: PurchasesManagerTests.cs línea 693  
Duración: < 1 ms

Se pueden agregar cantidad negativa de medicamentos al carrito

[Issue](#)

Código de software reparado

```
public Purchase CreatePurchase(Purchase purchase)
{
    string validEmail = @"^([w\.\-]+)@([w\.-]+)(\.\([w\]{2,3}\))+)$";
    Regex rgEmail = new(validEmail);
    if (string.IsNullOrEmpty(purchase.BuyerEmail) || !rgEmail.IsMatch(purchase.BuyerEmail))
        throw new InvalidResourceException("Invalid Email");

    if ((purchase.details == null || purchase.details.Count == 0))
        throw new InvalidResourceException("The list of items can't be empty");

    if (purchase.PurchaseDate == DateTime.MinValue)
        throw new InvalidResourceException("The purchase date is a mandatory field");

    decimal total = 0;
    foreach (var detail in purchase.details)
    {
        int pharmacyId = detail.Pharmacy.Id;
        if (pharmacyId <= 0)
            throw new ResourceNotFoundException($"Pharmacy Id is a mandatory field");

        var pharmacy = _pharmaciesRepository.GetOneByExpression(x => x.Id == pharmacyId);
        if (pharmacy is null)
            throw new ResourceNotFoundException($"Pharmacy {detail.Pharmacy.Id} not found");

        if (detail.Quantity <= 0)
            throw new InvalidResourceException("All items quantity should be bigger than zero");

        string drugCode = detail.Drug.Code;
        var drug = pharmacy.Drugs.FirstOrDefault(x => x.Code == drugCode && x.Deleted == false);
        if (drug is null)
            throw new ResourceNotFoundException($"Drug {drugCode} not found in Pharmacy {pharmacy.Name}");

        detail.Pharmacy = pharmacy;
        total = total + (drug.Price * detail.Quantity);
        detail.Price = drug.Price;
        detail.Drug = drug;
        detail.Status = PENDING;
    }
    purchase.TotalAmount = total;
    purchase.TrackingCode = generateTrackingCode();
    _purchasesRepository.InsertOne(purchase);
    _purchasesRepository.Save();

    return purchase;
}
```

## Código de casos de prueba

```
[TestMethod]
You, 15 hours ago • Add required unit tests ...
0 references | Run Test | Debug Test
public void Create_Purchase_Zero_Quantity()
{
    //Arrange
    pharmacy.Drugs = new List<Drug>();
    pharmacy.Drugs.Add(drug1);
    pharmacy.Drugs.Add(drug2);
    purchaseDetail = new List<PurchaseDetail> {
        new PurchaseDetail{Id = 1, Quantity = 0, Price = new decimal(100), Drug = drug1, Pharmacy = pharmacy},
        new PurchaseDetail{Id = 2, Quantity = 51, Price = new decimal(250), Drug = drug2, Pharmacy = pharmacy2 }
    };
    purchase.details = purchaseDetail;

    _pharmacyRespository.Setup(y => y.GetOneByExpression(x => x.Id == 1)).Returns(pharmacy);

    // Act & Assert
    var exception = Assert.ThrowsException<InvalidResourceException>(() => _purchasesManager.CreatePurchase(purchase));

    // Assert the exception message
    Assert.AreEqual("All items quantity should be bigger than zero", exception.Message);
}

[TestMethod]
0 references | Run Test | Debug Test
public void Create_Purchase_Negative_Quantity()
{
    //Arrange
    pharmacy.Drugs = new List<Drug>();
    pharmacy.Drugs.Add(drug1);
    pharmacy.Drugs.Add(drug2);
    purchaseDetail = new List<PurchaseDetail> {
        new PurchaseDetail{Id = 1, Quantity = -10, Price = new decimal(100), Drug = drug1, Pharmacy = pharmacy},
        new PurchaseDetail{Id = 2, Quantity = 51, Price = new decimal(250), Drug = drug2, Pharmacy = pharmacy2 }
    };
    purchase.details = purchaseDetail;

    _pharmacyRespository.Setup(y => y.GetOneByExpression(x => x.Id == 1)).Returns(pharmacy);

    // Act & Assert
    var exception = Assert.ThrowsException<InvalidResourceException>(() => _purchasesManager.CreatePurchase(purchase));

    // Assert the exception message
    Assert.AreEqual("All items quantity should be bigger than zero", exception.Message);
}
}
```

## Evidencia de ejecución de casos de prueba

**Fase Roja:**

Test Explorer

Test run finished: 293 Tests (291 Passed, 2 Failed, 0 Skipped) run in 11.9 sec

Test	Duration	Traits
PharmaGo.Test (293)	11.1 sec	
PharmaGo.Test.BusinessLogic.Test (158)	3.1 sec	
DrugManagerTests (27)	2.2 sec	
InvitationManagerTest (28)	337 ms	
LoginManagerTests (11)	50 ms	
PharmacyManagerTests (20)	44 ms	
PresentationManagerTest (1)	4 ms	
PurchasesManagerTests (32)	274 ms	
Approve_Purchase_Fail_Drug_Not_Found	26 ms	
Approve_Purchase_Fail_No_Stock	3 ms	
Approve_Purchase_Fail_Pharmacy_Not_Found	2 ms	
Approve_Purchase_Fail_Purchase_Detail_Not_Found	2 ms	
Approve_Purchase_Fail_Purchase_Not_Found	5 ms	
Approve_Purchase_Ok	9 ms	
Create_Purchase_Fail_Empty_Email	8 ms	
Create_Purchase_Fail_Invalid_Drug	6 ms	
Create_Purchase_Fail_Invalid_Email	1 ms	
Create_Purchase_Fail_Invalid_Pharmacy	2 ms	
Create_Purchase_Fail_Invalid_Pharmacy_Id	4 ms	
Create_Purchase_Fail_Invalid_Purchase_Date	1 ms	
Create_Purchase_Fail_Invalid_Quantity	4 ms	
Create_Purchase_Fail_No_Items	1 ms	
Create_Purchase_Fail_Null_Items	1 ms	
Create_Purchase_Negative_Quantity	110 ms	
Create_Purchase_Ok	8 ms	
Create_Purchase_Zero_Quantity	3 ms	
Get_All_Purchases_By_Date	12 ms	
Get_All_Purchases_By_Date_Invalid_End_Date	4 ms	
Get_All_Purchases_By_Date_Invalid_Start_Date	6 ms	
Get_All_Purchases_By_Date_No_Start_Date_And_No_End_Date	3 ms	
Get_All_Purchases_By_Date_Start_Date_Bigger_Than_End_Date	6 ms	
Get_Purchase_By_TrackingCode_Fail	2 ms	
Get_Purchase_By_TrackingCode_Ok	2 ms	
Get_Purchases_All_Ok	6 ms	
Get_Purchases_All_Ok__	4 ms	
Reject_Purchase_Fail_Drug_Not_Found	9 ms	
Reject_Purchase_Fail_Pharmacy_Not_Found	6 ms	
Reject_Purchase_Fail_Purchase_Detail_Not_Found	2 ms	
Reject_Purchase_Fail_Purchase_Not_Found	3 ms	
Reject_Purchase_Ok	13 ms	
RoleManagerTest (1)	6 ms	
StockManagerTest (25)	132 ms	
ApproveStockRequest_ShouldUpdateDrugStock	28 ms	
ApproveStockRequest_WithApprovedStatus_ShouldReturnException	5 ms	

**Group Summary**

PharmaGo.Test

Tests in group : 293

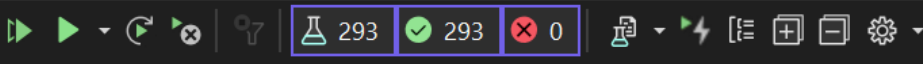
Total Duration: 11.1 sec

**Outcomes**

291 Passed

2 Failed

Fase Verde:

		
Test run finished: 293 Tests (293 Passed, 0 Failed, 0 Skipped) run in 6.9 sec		
Test	Duration	Traits
PharmaGo.Test (293)	6.2 sec	
PharmaGo.Test.BusinessLogic.Test (158)	693 ms	
DrugManagerTests (27)	336 ms	
InvitationManagerTest (28)	86 ms	
LoginManagerTests (11)	34 ms	
PharmacyManagerTests (20)	27 ms	
PresentationManagerTest (1)	5 ms	
PurchasesManagerTests (32)	92 ms	
Approve_Purchase_Fail_Drug_Not_Found	15 ms	
Approve_Purchase_Fail_No_Stock	1 ms	
Approve_Purchase_Fail_Pharmacy_Not_Found	1 ms	
Approve_Purchase_Fail_Purchase_Detail_Not_Found	1 ms	
Approve_Purchase_Fail_Purchase_Not_Found	1 ms	
Approve_Purchase_Ok	5 ms	
Create_Purchase_Fail_Empty_Email	6 ms	
Create_Purchase_Fail_Invalid_Drug	7 ms	
Create_Purchase_Fail_Invalid_Email	< 1 ms	
Create_Purchase_Fail_Invalid_Pharmacy	1 ms	
Create_Purchase_Fail_Invalid_Pharmacy_Id	1 ms	
Create_Purchase_Fail_Invalid_Purchase_Date	1 ms	
Create_Purchase_Fail_Invalid_Quantity	2 ms	
Create_Purchase_Fail_No_Items	< 1 ms	
Create_Purchase_Fail_Null_Items	< 1 ms	
Create_Purchase_Negative_Quantity	2 ms	
Create_Purchase_Ok	5 ms	
Create_Purchase_Zero_Quantity	1 ms	
Get_All_Purchases_By_Date	5 ms	
Get_All_Purchases_By_Date_Invalid_End_Date	1 ms	
Get_All_Purchases_By_Date_Invalid_Start_Date	1 ms	
Get_All_Purchases_By_Date_No_Start_Date_And_No_End_Date	3 ms	
Get_All_Purchases_By_Date_Start_Date_Bigger_Than_End_Date	5 ms	
Get_Purchase_By_TrackingCode_Fail	2 ms	

El sistema permite agregar cantidades negativas a una stock request

## Issue

## Código de software reparado

```
71         User existEmployee = null;
72         if (stockRequest.Details == null) throw new InvalidResourceException("Invalid stock details.");
73         if (stockRequest.Details.Count == 0) throw new InvalidResourceException("Invalid stock details.");
74 +         foreach (StockRequestDetail requestDetail in stockRequest.Details)
75 +         {
76 +             if(requestDetail.Quantity <= 0)
77 +             {
78 +                 throw new InvalidResourceException("Invalid stock details. Drug quantity can't be less than 1");
79 +             }
80 +         }
81         var session = _sessionRepository.GetOneByExpression(session => session.Token == new Guid(token));
82         if (session == null) throw new InvalidResourceException("Invalid session from employee.");
83         var userId = session.UserId;
```

## Código de casos de prueba

```
47 // [TestMethod]
48 // [ExpectedException(typeof(InvalidResourceException))]
49 // public void GetStock_MultipleFilters_ShouldReturnDeserializedStockRequest_By_One
50 // {
51 //     Assert.AreEqual(2, result.Count());
52 // }
53
54 [TestMethod]
55 [ExpectedException(typeof(InvalidResourceException))]
56 public void CreateStockRequest_WithDrugWithZeroQuantity_ShouldReturnException()
57 {
58     var drug = new Drug() { Id = 1, Code = "732324" };
59     User employee = new User() { Id = 1, UserName = "jcastro" };
60     var stockRequest = new StockRequest()
61     {
62         Id = 1,
63         Status = Domain.Enums.StockRequestStatus.Pending,
64         Employee = employee,
65         Details = new List<StockRequestDetail>()
66         {
67             new StockRequestDetail() { Id = 1, Drug = drug, Quantity = 50 },
68             new StockRequestDetail() { Id = 1, Drug = drug, Quantity = 0 }
69         },
70         RequestDate = DateTime.Now
71     };
72     //Act
73     _stockRequestManager.CreateStockRequest(stockRequest, token);
74 }
75
76 [TestMethod]
77 [ExpectedException(typeof(InvalidResourceException))]
78 public void CreateStockRequest_WithDrugWithNegativeQuantity_ShouldReturnException()
79 {
80     var drug = new Drug() { Id = 1, Code = "732324" };
81     User employee = new User() { Id = 1, UserName = "jcastro" };
82     var stockRequest = new StockRequest()
83     {
84         Id = 1,
85         Status = Domain.Enums.StockRequestStatus.Pending,
86         Employee = employee,
87         Details = new List<StockRequestDetail>()
88         {
89             new StockRequestDetail() { Id = 1, Drug = drug, Quantity = 50 },
90             new StockRequestDetail() { Id = 1, Drug = drug, Quantity = -10 }
91         },
92         RequestDate = DateTime.Now
93     };
94     //Act
95     _stockRequestManager.CreateStockRequest(stockRequest, token);
96 }
97
98 }
```

## Evidencia de ejecución de casos de prueba

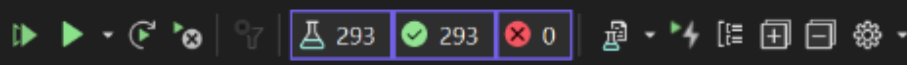
### Fase Roja:

▶ ▶ ▶ ↺ ✕ | 🔍 | 🧪 293 ✅ 291 ❌ 2 | 🧪 ⚡ ⚙️ + - ⚙️

Test	Duration	Traits
PharmaGo.Test (293)	4.6 sec	
PharmaGo.Test.BusinessLogic.Test (158)	507 ms	
DrugManagerTests (27)	214 ms	
InvitationManagerTest (28)	49 ms	
LoginManagerTests (11)	14 ms	
PharmacyManagerTests (20)	13 ms	
PresentationManagerTest (1)	2 ms	
PurchasesManagerTests (30)	64 ms	
RoleManagerTest (1)	2 ms	
StockManagerTest (27)	132 ms	
ApproveStockRequest_ShouldUpdateDrugStock	11 ms	
ApproveStockRequest_WithApprovedStatus_ShouldReturnException	1 ms	
ApproveStockRequest_WithDeletedDrugs_ShouldReturnException	1 ms	
ApproveStockRequest_WithInvalidStockRequest_ShouldReturnException	1 ms	
ApproveStockRequest_WithRejectedStatus_ShouldReturnException	1 ms	
CreateStockRequest_ShouldCreateStockRequest	3 ms	
CreateStockRequest_WithADrugWithNegativeQuantity_ShouldReturnE...	83 ms	
CreateStockRequest_WithADrugWithZeroQuantity_ShouldReturnExcep...	1 ms	
CreateStockRequest_WithCeroDetails_ShouldReturnException	1 ms	
CreateStockRequest_WithInvalidDrugs_ShouldReturnException	1 ms	
CreateStockRequest_WithInvalidEmployee_ShouldReturnException	1 ms	
CreateStockRequest_WithNulEmployee_SouldReturnException	2 ms	
CreateStockRequest_WithNullDetails_ShouldReturnException	< 1 ms	
GetStock_WithInvalidEmployee_ShouldReturnException_By_Owner	3 ms	
GetStock_WithInvalidUser_ShouldReturnException_By_Owner	2 ms	
GetStock_WithNullEmployee_ShouldReturnException_By_Owner	< 1 ms	
GetStock_WithNullFilters_ShouldReturnIEnumerableStockRequest_By_...	5 ms	
GetStockByEmployee_WithAllFilters_ShouldReturnIEnumerableStockRe...	7 ms	
GetStockByEmployee_WithCodeAndStatus_ShouldReturnIEnumerableS...	2 ms	
GetStockByEmployee_WithDateAndCode_ShouldReturnIEnumerableSt...	1 ms	
GetStockByEmployee_WithDateAndStatus_ShouldReturnIEnumerableSt...	1 ms	
GetStockByEmployee_WithInvalidEmployee_ShouldReturnException	1 ms	
GetStockByEmployee_WithNullEmployee_ShouldReturnException	< 1 ms	
GetStockByEmployee_WithNullFilters_ShouldReturnIEnumerableStockR...	1 ms	
RejectStockRequest_WithApprovedStatus_ShouldReturnException	1 ms	
RejectStockRequest_WithInvalidStockRequest_ShouldReturnException	< 1 ms	
RejectStockRequest_WithRejectedStatus_ShouldReturnException	1 ms	
UnitMeasureManagerTest (1)	4 ms	
UsersManagerTests (12)	13 ms	
PharmaGo.Test.DataAccess.Test (38)	3.9 sec	
PharmaGo.Test.WebApi.Test (97)	276 ms	



## Fase Verde:

Test Explorer		
		
Test run finished: 293 Tests (293 Passed, 0 Failed, 0 Skipped) run in 4.3 sec		
Test	Duration	Traits
▶ ✓ PharmacyManagerTests (20)	17 ms	
▶ ✓ PresentationManagerTest (1)	1 ms	
▶ ✓ PurchasesManagerTests (30)	90 ms	
▶ ✓ RoleManagerTest (1)	11 ms	
▲ ✓ StockManagerTest (27)	82 ms	
✓ ApproveStockRequest_ShouldUpdateDrugStock	18 ms	
✓ ApproveStockRequest_WithApprovedStatus_ShouldReturnException	2 ms	
✓ ApproveStockRequest_WithDeletedDrugs_ShouldReturnException	2 ms	
✓ ApproveStockRequest_WithInvalidStockRequest_ShouldReturnException	2 ms	
✓ ApproveStockRequest_WithRejectedStatus_ShouldReturnException	1 ms	
✓ CreateStockRequest_ShouldCreateStockRequest	4 ms	
✓ CreateStockRequest_WithADrugWithNegativeQuantity_ShouldReturnE...	1 ms	
✓ CreateStockRequest_WithADrugWithZeroQuantity_ShouldReturnExcep...	< 1 ms	
✓ CreateStockRequest_WithCeroDetails_ShouldReturnException	< 1 ms	
✓ CreateStockRequest_WithInvalidDrugs_ShouldReturnException	6 ms	
✓ CreateStockRequest_WithInvalidEmployee_ShouldReturnException	2 ms	
✓ CreateStockRequest_WithNulEmployee_SouldReturnException	2 ms	
✓ CreateStockRequest_WithNullDetails_ShouldReturnException	< 1 ms	
✓ GetStock_WithInvalidEmployee_ShouldReturnException_By_Owner	2 ms	
✓ GetStock_WithInvalidUser_ShouldReturnException_By_Owner	2 ms	
✓ GetStock_WithNullEmployee_ShouldReturnException_By_Owner	1 ms	
✓ GetStock_WithNullFilters_ShouldReturnIEnumerableStockRequest_By_...	8 ms	
✓ GetStockByEmployee_WithAllFilters_ShouldReturnIEnumerableStockRe...	10 ms	
✓ GetStockByEmployee_WithCodeAndStatus_ShouldReturnIEnumerableS...	2 ms	
✓ GetStockByEmployee_WithDateAndCode_ShouldReturnIEnumerableSt...	1 ms	
✓ GetStockByEmployee_WithDateAndStatus_ShouldReturnIEnumerableSt...	1 ms	
✓ GetStockByEmployee_WithInvalidEmployee_ShouldReturnException	2 ms	
✓ GetStockByEmployee_WithNullEmployee_ShouldReturnException	2 ms	
✓ GetStockByEmployee_WithNullFilters_ShouldReturnIEnumerableStockR...	5 ms	
✓ RejectStockRequest_WithApprovedStatus_ShouldReturnException	3 ms	
✓ RejectStockRequest_WithInvalidStockRequest_ShouldReturnException	1 ms	
✓ RejectStockRequest_WithRejectedStatus_ShouldReturnException	2 ms	
▶ ✓ UnitMeasureManagerTest (1)	3 ms	
▶ ✓ UsersManagerTests (12)	26 ms	
▶ ✓ PharmaGo.Test.DataAccess.Test (38)	2.6 sec	

