Código de los casos de prueba

Product.Feature

```
Feature: Product

Scenario: Create new product successfully
    Given The product code is $<code>
    And the name is $<name>
    And the description is $<description>
    And the price is $<price>
    And a certain pharmacy
    When I Click the Create button
    Then The product is created shoud be $<isCreated>
    And A message in the lower part on the screen appears
    And Added to the database
 Examples:
    | code    | name | description | price | isCreated |
    | 12345   | coca   | Dale sabor a tu vida          | 80    | true      |
    | 12345   | Lorem ipsum dolor sit amet co.   | Dale sabor a tu vida          | 80    |
true      |
    | 12345   | coca   | Lorem ipsum dolor sit amet consectetur adipiscing elit malesuada
vulp.          | 80    | true      |
    | 12345   | coca   | Dale sabor a tu vida          | 80.50    | true      |


Scenario: Error creating new product
    Given The product code is $<code>
    And the name is $<name>
    And the description is $<description>
    And the price is $<price>
    And a certain pharmacy
    When I Click the Create button
    Then The product is created shoud be $<isCreated>
    And A message in the lower part on the screen appears
 Examples:
    | code    | name | description | price | isCreated |
    | 2     | coca   | Dale sabor a tu vida          | 80    | false     |
    | a     | coca   | Dale sabor a tu vida          | 80    | false     |
```

```
    | -5     | coca    | Dale sabor a tu vida          | 80    | false     |
    | 123456| coca    | Dale sabor a tu vida          | 80    | false     |
    | @      | coca    | Dale sabor a tu vida          | 80    | false     |
    |        | coca    | Dale sabor a tu vida          | 80    | false     |
    | 12345 | Lorem ipsum dolor sit amet con.   | Dale sabor a tu vida          | 80    |
false  |
    | 12345 | @$&     | Dale sabor a tu vida          | 80    | false     |
    | 12345 |         | Dale sabor a tu vida          | 80    | false     |
    | 12345 | coca    | Lorem ipsum dolor sit amet consectetur adipiscing elit accumsan
libero.          | 80    | false  |
    | 12345 | coca    | "$@                           | 80    | false     |
    | 12345 | coca    |                               | 80    | false     |
    | 12345 | coca    | Dale sabor a tu vida          | -5    | false     |
    | 12345 | coca    | Dale sabor a tu vida          | asd   | false     |
    | 12345 | coca    | Dale sabor a tu vida          | @$%   | false     |
    | 12345 | coca    | Dale sabor a tu vida          |       | false     |

Scenario: Delete a product succesfully
    Given The product code is $<code>
    When I click the delete button
    Then the product should be deleted from the database
 Examples:
    | code    |
    | AWXS    |

Scenario: Invalid Code deleting a product
    Given The product code is $<code>
    When I click the delete button
    Then The product is not deleted
    And an error message is returned to the employee
 Examples:
    | code    |
    | 12345   |
    | 123456   |
    | 123457   |
    | 123458   |

Scenario: A non-employee deleting a product
    Given The product code is $<code>
    When A non-employee user touches a delete button
```

```
    Then The product is not deleted
    And an Authorization error message is returned to the employee
 Examples:
    | code    |
    | 12345   |
    | 123456   |
    | 123457   |
    | 123458   |

Scenario: Get all drugs
    When I enter the product page
    Then All products are shown to me
```

ProductStepDefinition.cs

```
[Binding]
    public class ProductSteps
    {
        private Product _product = new Product();
        private bool _isProductCreated;
        private Pharmacy _pharmacy = new Pharmacy() { Id = 1, Name = "pharmacy", Address =
"address", Users = new List<User>() };
        private Mock<IRepository<Product>> _productRepository;
        private Mock<IRepository<Pharmacy>> _pharmacyRepository;
        private Mock<IRepository<Session>> _sessionRepository;
        private Mock<IRepository<User>> _userRepository;
        private IProductManager _productManager;
        private string _mockToken = "c80da9ed-1b41-4768-8e34-b728cae25d2f";

        #region Scenario: Create new product successfully
        [Given("The product code is (.*)")]
        public void GivenAValidProductCodeInPharmacy(string code)
        {
            _product.Code = code;
        }

        [Given("the name is (.*)")]
```

```
public void GivenAValidProductName(string name)
{
    _product.Name = name;
}

[Given("the description is (.*)")]
public void GivenAValidProductDescription(string description)
{
    _product.Description = description;
}

[Given("the price is (.*)")]
public void GivenAValidProductPrice(string price)
{
    if (decimal.TryParse(price, out decimal parsedPrice))
    {
        _product.Price = parsedPrice;
    }
}

[Given("a certain pharmacy")]
public void GivenAValidPharmacy()
{
    _product.Pharmacy = _pharmacy;
}

[When("I Click the Create button")]
public void WhenClickTheCreateButton()
{
    // Llama al metodo ValidOrFail para verificar si el producto es valido
    try
    {
        _product.ValidOrFail();
        _isProductCreated = true;
    }
    catch (InvalidResourceException)
    {
        // Maneja la excepcion si la validacion falla
        _isProductCreated = false;
    }
```

```
        }

        [Then("The product is created shoud be (.*)")]
        public void ThenTheProductIsCreatedSuccessfully(string isCreated)
        {
            _userRepository = new Mock<IRepository<User>>();
            _sessionRepository = new Mock<IRepository<Session>>();
            _productRepository = new Mock<IRepository<Product>>();
            _pharmacyRepository = new Mock<IRepository<Pharmacy>>();
            _productManager =  new ProductManager(_productRepository.Object,
_pharmacyRepository.Object, _sessionRepository.Object, _userRepository.Object);
            if (bool.TryParse(isCreated, out bool parsedIsCreated))
            {
                _productManager.Create(_product, _mockToken);
                try
                {
                    if (parsedIsCreated)
                    {
                        Assert.True(_isProductCreated);
                    }
                    else
                    {
                        Assert.False(_isProductCreated);
                    }
                }
                catch (InvalidResourceException)
                {
                    if (!parsedIsCreated)
                    {
                        Assert.True(_isProductCreated);
                    }
                    else
                    {
                        Assert.False(_isProductCreated);
                    }
                }

            }

        }
```

```
        [Then("A message in the lower part on the screen appears")]
        public void ThenAMessageInTheLowerPartOnTheScreenAppears()
        {
            // No aplica, solo al frontend
        }

        [Then("Added to the database")]
        public void ThenAddedToTheDatabase()
        {
            // No se necesita una base de datos real, asi que no se realiza ninguna accion
aqui
        }
        #endregion

        #region Delete a product
        private Product productTest = new Product()
        {
            Code = "AWS",
            Name = "TestName",
            Deleted = false,
            Description = "Test drug description",
            Id = 1,
            Pharmacy = new Pharmacy(),
            Price = 1000,
            Stock = 100
        };

        [When(@"I click the delete button")]
        public void WhenIClickTheDeleteButton()
        {
            _product = productTest;
        }

        [Then(@"the product should be deleted from the database")]
        public void ThenTheProductShouldBeDeletedFromTheDatabase()
        {
            var productToDelete = _product;
            _userRepository = new Mock<IRepository<User>>();
            _sessionRepository = new Mock<IRepository<Session>>();
```

```
            _productRepository = new Mock<IRepository<Product>>(MockBehavior.Strict);
            _pharmacyRepository = new Mock<IRepository<Pharmacy>>();
            _productManager = new ProductManager(_productRepository.Object,
_pharmacyRepository.Object, _sessionRepository.Object, _userRepository.Object);

            _productRepository.Setup(p =>
p.GetOneDetailByExpression(It.IsAny<Expression<Func<Product,
bool>>>())).Returns(productToDelete);
            _productRepository.Setup(p => p.DeleteOne(It.IsAny<Product>()));

            _productManager.Delete(productToDelete, _mockToken);

            _productRepository.VerifyAll();
        }

        [Then(@"The product is not deleted")]
        public void ThenTheProductIsNotDeleted()
        {
            var productToDelete = _product;
            _userRepository = new Mock<IRepository<User>>();
            _sessionRepository = new Mock<IRepository<Session>>();
            _productRepository = new Mock<IRepository<Product>>(MockBehavior.Strict);
            _pharmacyRepository = new Mock<IRepository<Pharmacy>>();
            _productManager = new ProductManager(_productRepository.Object,
_pharmacyRepository.Object, _sessionRepository.Object, _userRepository.Object);

            _productRepository.Setup(p =>
p.GetOneDetailByExpression(It.IsAny<Expression<Func<Product,
bool>>>())).Returns(productToDelete);

            try
            {
                _productManager.Delete(productToDelete, _mockToken);
            } catch (Exception e)
            {
                _productRepository.VerifyAll();
            }

        }
```

```
        [Then(@"an error message is returned to the employee")]
        public void ThenAnErrorMessageIsReturnedToTheEmployee()
        {
            var productToDelete = _product;
            _userRepository = new Mock<IRepository<User>>();
            _sessionRepository = new Mock<IRepository<Session>>();
            _productRepository = new Mock<IRepository<Product>>(MockBehavior.Strict);
            _pharmacyRepository = new Mock<IRepository<Pharmacy>>();
            _productManager = new ProductManager(_productRepository.Object,
_pharmacyRepository.Object, _sessionRepository.Object, _userRepository.Object);

            _productRepository.Setup(p =>
p.GetOneDetailByExpression(It.IsAny<Expression<Func<Product,
bool>>>())).Returns((Product)null);

            Assert.Throws<ResourceNotFoundException>(() =>
_productManager.Delete(productToDelete, _mockToken));
        }

        #region A non-employee deleting a product
        private string nonEmployeeToken = "";
        [When(@"A non-employee user touches a delete button")]
        public void WhenANon_EmployeeUserTouchesADeleteButton()
        {
            nonEmployeeToken = "59C25ED6-1090-49FD-ADF2-DE24E1E932F0";
        }

        [Then(@"an Authorization error message is returned to the employee")]
        public void ThenAnAuthorizationErrorMessageIsReturnedToTheEmployee()
        {
            //aplica solo a llamadas http
        }
        #endregion

        #region Get all drugs
        [When(@"I enter the product page")]
        public void WhenIEnterTheProductPage()
        {
        }
```

```
        [Then(@"All products are shown to me")]
        public void ThenAllProductsAreShownToMe()
        {
            //front
        }

        #endregion
        #endregion

    }
```

ModifyProduct.Feature

```
Feature: Modify product

Scenario: Modify a product successfully
Given An existing product
When I change a $<variable> with the $<value>
And Click the Modify button
Then The product is modified $<isModified>
And A message in the lower part on the screen appears
And Updated in the database
Examples:
|variable|value| isModified |
|name   |coca1|true|
|description|Dale sabor a tu vida|true|
|price  |50|true|
|stock  |30|true|


Scenario: Invalid product name modifying existing product
Given An existing product
When I change a $<variable> with the $<value>
And Click the Modify button
Then The product is modified $<isModified>
And A message in the lower part on the screen appears
And Database is not updated
Examples:
|variable|value| isModified |
```

```
|name| !$% | false |
|name|  | false |


Scenario: Invalid product description modifying existing product
Given An existing product
When I change a $<variable> with the $<value>
And Click the Modify button
Then The product is modified $<isModified>
And A message in the lower part on the screen appears
And Database is not updated
Examples:
|variable         |value| isModified |
|description      | !$%| false |
|description      | | false |

Scenario: Invalid product price modifying existing product
Given An existing product
When I change a $<variable> with the $<value>
And Click the Modify button
Then The product is modified $<isModified>
And A message in the lower part on the screen appears
And Database is not updated
Examples:
|variable |value    | isModified |
| price   |  -10    | false |
| price   |  abc    | false |
| price   |  !$%    | false |

Scenario: Invalid product stock modifying existing product
Given An existing product
When I change a $<variable> with the $<value>
And Click the Modify button
Then The product is modified $<isModified>
And A message in the lower part on the screen appears
And Database is not updated
Examples:
| variable|value| isModified |
| stock   | -5| false |
| stock   | abc| false |
```

```
| stock    |   | false |
```

ModifyProductStepDefinition.cs

```
public class ModifyProductStepDefinition
    {
        [Binding]
        public class ProductSteps
        {
            private Pharmacy _pharmacy = new Pharmacy() { Id = 1, Name = "pharmacy",
Address = "address", Users = new List<User>() };
            private Product _productBackup;
            private bool _isProductUpdated;
            private Product _product;
            private Mock<IRepository<Product>> _productRepository;
            private Mock<IRepository<Pharmacy>> _pharmacyRepository;
            private Mock<IRepository<Session>> _sessionRepository;
            private Mock<IRepository<User>> _userRepository;
            private IProductManager _productManager;
            private string _mockToken = "c80da9ed-1b41-4768-8e34-b728cae25d2f";

            #region Scenario: Modify new product successfully

            [Given("An existing product")]
            public void GivenAValidProductCodeInPharmacy()
            {
                _product = new Product() { Code = "1", Name = "Coca", Description = "Dale
sabor a tu vida", Price = 50, Stock = 10, Deleted = false, Id = 1, Pharmacy = _pharmacy};
                _productBackup = new Product() { Code = "1", Name = "Coca", Description =
"Dale sabor a tu vida", Price = 50, Stock = 10, Deleted = false, Id = 1, Pharmacy =
_pharmacy };
            }

            [When("I change a (.*) with the (.*)")]
            public void WhenIChangeTheValue(string variable, string value)
            {
                if(variable=="name")
                {
```

```
                    _product.Name = value;
                }
                else if (variable == "description")
                {
                    _product.Description = value;
                }
                else if (variable == "price")
                {
                    if (decimal.TryParse(value, out decimal parsedPrice))
                    {
                        _product.Price = parsedPrice;
                    }
                }
                else if (variable == "stock")
                {
                    if (int.TryParse(value, out int parsedStock))
                    {
                        _product.Stock = parsedStock;
                    }
                }
            }
            [When("Click the Modify button")]
            public void WhenClickTheModifyButton()
            { }

            [Then("The product is modified (.*)")]
            public void ThenTheProductIsUpdatedSuccessfully(string isModified)
            {
                _userRepository = new Mock<IRepository<User>>();
                _sessionRepository = new Mock<IRepository<Session>>();
                _productRepository = new Mock<IRepository<Product>>();
                _pharmacyRepository = new Mock<IRepository<Pharmacy>>();
                _productManager = new ProductManager(_productRepository.Object,
_pharmacyRepository.Object, _sessionRepository.Object, _userRepository.Object);
                if (bool.TryParse(isModified, out bool parsedIsUpdated))
                {
                    _productManager.Update(_product.Code, _product);
                    try
                    {
                        if (parsedIsUpdated)
```

```
                    {
                        Assert.True(_isProductUpdated);
                    }
                    else
                    {
                        Assert.False(_isProductUpdated);
                    }
                }
                catch (InvalidResourceException)
                {
                    if (!parsedIsUpdated)
                    {
                        Assert.True(_isProductUpdated);
                    }
                    else
                    {
                        Assert.False(_isProductUpdated);
                    }
                }
            }
        }
        [Then(@"Database is not updated")]
        public void ThenDatabaseIsNotUpdated()
        {
        }
        [Then(@"Updated in the database")]
        public void ThenUpdatedInTheDatabase()
        {
        }
        #endregion
    }

}
```