# **Protocoles de test**

*Utilisez le template ci-dessous pour remplir le document – remplacez les mots en italique.*

**Objectif Test :** Vérifier les spécifications de validation du système de gestion de l'inventaire

**Test 1 : Vérification du nom du produit**

**TestMissingName()**

Cas de spécification : Le produit doit porter un nom

**Étape 1 (Arrange) :**

Préparer un produit avec un nom vide :

product.Name = null;

product.Price = "9.9";

product.Stock = "1";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est MissingName :

Assert.False(isValid);

Assert.Equal("MissingName", GetFirstErrorMessage(product));

**Test 2 : Vérification de la présence du prix**

**TestMissingPrice()**

Cas de spécification : Le champ Prix est obligatoire

**Étape 1 (Arrange) :**

Préparer un produit sans prix :

product.Name = "Test Product";

product.Stock = "1";

product.Price = null;

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est MissingPrice :

Assert.False(ValidateModel(isValid));

Assert.Equal("MissingPrice", GetFirstErrorMessage(product));

**Test 3 : Vérification que le prix est un nombre décimal**

**TestPriceNotANumber()**

Cas de spécification : Le prix doit être un nombre décimal

**Étape 1 (Arrange) :**

Préparer un produit avec un prix non numérique :

product.Name = "Test Product";

product.Price = "abc";

product.Stock = "3";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est PriceNotANumber :

Assert.False(isValid);

Assert.Equal("PriceNotANumber", GetFirstErrorMessage(product));

**Test 4 : Vérification que le prix est supérieur à zéro**

**TestPriceNotGreaterThanZero()**

Cas de spécification : Le prix doit être supérieur à zéro

**Étape 1 (Arrange) :**

Préparer un produit avec un prix de zéro ou moins :

product.Name = "Test Product";

product.Price = "0";

product.Stock = "1";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est PriceNotGreaterThanZero :

Assert.False(isValid);

Assert.Equal("PriceNotGreaterThanZero", GetFirstErrorMessage(product));

**Test 5 : Vérification de la présence de la quantité**

**TestMissingQuantity()**

Cas de spécification : Le champ Quantité est obligatoire

**Étape 1 (Arrange) :**

Préparer un produit sans quantité :

product.Name = "Test Product";

product.Stock = null;

product.Price = "42";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est MissingQuantity :

Assert.False(isValid);

Assert.Equal("MissingQuantity", GetFirstErrorMessage(product));

**Test 6 : Vérification que la quantité est un nombre entier**

**TestQuantityNotAnInteger()**

Cas de spécification : La quantité doit être un nombre entier

**Étape 1 (Arrange) :**

Préparer un produit avec une quantité non entière :

product.Name = "Test Product";

product.Stock = "1.5";

product.Price = "19";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est QuantityNotAnInteger :

Assert.False(isValid);

Assert.Equal("QuantityNotAnInteger", GetFirstErrorMessage(product));

**Test 7 : Vérification que la quantité est supérieure à zéro**

**TestQuantityNotGreaterThanZero()**

Cas de spécification : La quantité doit être supérieure à zéro

**Étape 1 (Arrange) :**

Préparer un produit avec une quantité de zéro ou moins :

product.Name = "Test Product";

product.Price = "7";

product.Stock = "0";

**Étape 2 (Act) :**

Lancer la validation du produit :

var isValid = ValidateModel(product);

**Étape 3 (Assert) : Vérification**

Vérifier que le message d'erreur est QuantityNotGreaterThanZero :

Assert.False(isValid);

Assert.Equal("QuantityNotGreaterThanZero", GetFirstErrorMessage(product));

Objectif Test : Vérification complète des spécifications de validation du système de gestion de l'inventaire