Zijad Lendo 19166

White box testiranje – izvještaj

Metoda getFilteredProducts(List<Product> products) u Filter:

public List<Product> getFilteredProducts(List<Product> products)

{

for (int i = 0; i < products.Count; i++)

{

var product = products[i];

if ((minPrice != null && product.Price < minPrice) || (maxPrice != null && product.Price > maxPrice)) {

products.Remove(product);

i--;

continue;

}

if (manufacturers != null)

{

bool found = false;

foreach (var man in manufacturers)

{

if (product.Manufacturer.Equals(man))

{

found = true;

break;

}

}

if (!found)

{

products.Remove(product);

i--;

continue;

}

}

if (categories != null)

{

bool found = false;

foreach(var cat in categories)

{

if (product.Category.Equals(cat))

{

found = true;

break;

}

}

if (!found)

{

products.Remove(product);

i--;

continue;

}

}

}

if (sortStrategy != null)

{

products = sortStrategy.sortProducts(products);

}

return products;

}

Testovi za white box:

private readonly List<Product> productsOriginal = new ()

{

new() { Price = 10, Manufacturer = "Volvo", Category = "Sedan" },

new() { Price = 100, Manufacturer = "Mercedes", Category = "Hatchback" },

new() { Price = 9, Manufacturer = "Volvo", Category = "Sedan" },

new() { Price = 101, Manufacturer = "Volvo", Category = "Hatchback" },

new() { Price = 10, Manufacturer = "Volkswagen", Category = "Sedan" },

new() { Price = 10, Manufacturer = "Volvo", Category = "Coupe" }

};

private bool IsFiltered(int? min, int? max, List<String>? manufacturers, List<String>? categories, ISortStrategy? sortStrategy)

{

var productsExpected = (

from product in productsOriginal

where min == null || product.Price >= min

where max == null || product.Price <= max

where manufacturers == null || manufacturers.Contains(product.Manufacturer)

where categories == null || categories.Contains(product.Category)

select product

).ToList();

if (sortStrategy != null)

{

productsExpected = sortStrategy.sortProducts(productsExpected);

}

var filter = new Filter(min, max, manufacturers, categories, sortStrategy);

return Enumerable.SequenceEqual(productsExpected, filter.getFilteredProducts(productsOriginal));

}

[TestMethod]

public void MinFilterTest()

{

int min = 10;

Assert.IsTrue(IsFiltered(min, null, null, null, null));

}

[TestMethod] public void MaxFilterTest()

{

int max = 100;

Assert.IsTrue(IsFiltered(null, max, null, null, null));

}

[TestMethod]

public void ManufacturerFilterTest()

{

var manufacturers = new List<string> { "Mercedes", "Volvo" };

Assert.IsTrue(IsFiltered(null, null, manufacturers, null, null));

}

[TestMethod]

public void CategoryFilterTest()

{

var categories = new List<string> { "Sedan", "Hatchback" };

Assert.IsTrue(IsFiltered(null, null, null, categories, null));

}

[TestMethod]

public void AlphabeticalSortFilterTest()

{

var sortStrategy = new AlphabeticalStrategy();

Assert.IsTrue(IsFiltered(null, null, null, null, sortStrategy));

}

[TestMethod]

public void HighestPriceSortFilterTest()

{

var sortStrategy = new HighestPriceStrategy();

Assert.IsTrue(IsFiltered(null, null, null, null, sortStrategy));

}

[TestMethod]

public void LowestPriceSortFilterTest()

{

var sortStrategy = new LowestPriceStrategy();

Assert.IsTrue(IsFiltered(null, null, null, null, sortStrategy));

}

Analiza: testovi postuju line coverage, condition coverage i path coverage sa svim prolazećim.