Challenging Purchase Order System

Challenging Purchase Order System

Arthur Clarkson

29th October 2024

One of the most challenging tasks on our intranet is updating a widely-used method like `SavePurchaseOrder()`, as changes can easily impact various dependent areas. Each modification requires meticulous consideration of how it will affect all instances where the method is implemented. At Labman, I approach these updates with unwavering diligence, constantly assessing each line of code to ensure stability across the system.

In cases where automated testing isn't feasible due to the method's placement, I am committed to performing thorough end-to-end testing in each context where the method is used. This careful attention to detail and proactive testing demonstrates my dedication to delivering reliable, effective solutions while minimizing disruption. My determination to succeed at Labman drives me to ensure that every change enhances the system without introducing new issues.

When embarking on such a task, I develop a comprehensive plan outlining where each method is used and how I can test it. For example, on a recent feedback item, I formulated the following plan:

```
Method Name: SavePurchaseOrder()

Places Used:
- CreatePurchaseOrder
- CreatePurchaseOrderFromCreditCardPurchase
- CreatePurchaseOrderFromStandardPartsList

How to test:
- Create a purchase order via the PO page and check details
- Create a purchase order from a credit card purchase and check details
- Create a standard parts list and check each purchase order created from it and check details
```

Upon examining each method, I observed that they all create variations of the `PurchaseOrderDto` depending on the data provided, then call `SavePurchaseOrder(PurchaseOrderDto)`. My task was to add a non-nullable payment method to the purchase order, ensuring it was set properly.

For the `CreatePurchaseOrder` method, I needed to ensure that the DTO's `PaymentMethodId` field was populated. Since this method is called from a form, I added a field to the page and implemented validation to make sure it is set:

```
public void CreatePurchaseOrder(PurchaseOrderForm form)
{
    // Other validation
    var paymentMethod = GetPaymentMethodById(form.PaymentMethodId);

    if (paymentMethod == null)
    {
        throw new Exception($"Could not find payment method with ID: ({form.PaymentMethodId})");
    }

    // Additional code to populate DTO
    var dto = new PurchaseOrderDto
    {
        PaymentMethodId = form.PaymentMethodId,
        // Populate other fields
    };

    SavePurchaseOrder(dto);
}
```

For the `CreatePurchaseOrderFromCreditCardPurchase` method, I determined that I just needed to populate the DTO's payment method with the ID of the credit card payment method. Here's how I updated the code:

```
public void CreatePurchaseOrderFromCreditCardPurchase(CreditCardPurchase purchase)
{
    // Retrieve the credit card payment method
    var creditCardPaymentMethod = GetPaymentMethodByName("Credit Card");

    if (creditCardPaymentMethod == null)
    {
        throw new Exception("Credit card payment method not found.");
    }

    // Create and populate DTO
    var dto = new PurchaseOrderDto
    {
        PaymentMethodId = creditCardPaymentMethod.Id,
```

```
        // Populate other fields based on the purchase
    };

    SavePurchaseOrder(dto);
}
```

For `CreatePurchaseOrderFromStandardPartsList`, each standard parts list has a default payment method. I populated each purchase order DTO accordingly:

```
public void CreatePurchaseOrderFromStandardPartsList(StandardPartsList partsList)
{
    foreach (var item in partsList.Items)
    {
        // Create and populate DTO
        var dto = new PurchaseOrderDto
        {
            PaymentMethodId = partsList.DefaultPaymentMethodId,
            // Populate other fields based on the item
        };

        SavePurchaseOrder(dto);
    }
}
```

By updating each method and rigorously testing them in their respective contexts, I ensured that the new payment method field was correctly integrated without affecting existing functionality. This process showcases my determination to succeed at Labman by delivering high-quality work that upholds the integrity of our systems. My commitment to thorough planning, attention to detail, and proactive problem-solving contributes to the ongoing success and reliability of our projects.