Let's try to create FULL SAMPLE CYCLE OF: BRD, TRD, TC, RTM, UNIT TEST!

The business requirement. These requirements are often less technical and more focused on business needs. Here's a simplified business requirement table based on the provided technical requirements related to the e-commerce platform's checkout process:

ID	Requirement Description	Business Objectives and Benefits	Priority	Status	Owner/Assigned To
BR1	Seamless Shopping Cart and Checkout Experience	Provide users with an intuitive and efficient shopping cart and checkout process.	High	Pending	Product Manager
BR2	Support for Multiple Payment Methods	Allow users to choose from various payment options, enhancing convenience.	High	Pending	Product Manager
BR3	Apply and Validate Discount Codes	Enable users to apply and receive discounts, promoting customer loyalty.	High	Pending	Marketing Team
BR4	Email Confirmation after Purchase	Send email confirmations to users, ensuring transparency and order acknowledgment.	High	Pending	Customer Support Team
BR5	Error Handling and User Notifications	Provide a user-friendly experience with clear error messages and notifications.	High	Pending	UX/UI Team
BR6	Performance Optimization during High Traffic Conditions	Ensure system performance under heavy traffic, preventing disruptions.	High	Pending	Infrastructure Team
BR7	Inventory Management for Out-of-Stock Items	Enhance customer satisfaction by preventing purchases of unavailable items.	High	Pending	Inventory Manager
BR8	Secure Handling of Invalid Shipping Addresses	Improve address validation to minimize shipping issues and enhance order accuracy.	High	Pending	Logistics Team
BR9	Integration with External Payment Gateway for Credit Card Payments	Ensure secure and reliable credit card transactions through an external gateway.	High	Pending	Finance Team

In this business requirement table:

- **Requirement ID:** A unique identifier for each business requirement.
- **Requirement Description:** A brief description of the business objective or functionality.
- **Business Objectives and Benefits:** The high-level goals and benefits associated with fulfilling the business requirement.

- **Priority:** The importance or urgency assigned to the business requirement.
- **Status:** Indicates the current state of the business requirement (e.g., Pending, In Progress).
- **Owner/Assigned To:** The individual or team responsible for overseeing or implementing the business requirement.

Below is a technical requirement table derived from the provided business requirement table. This table outlines specific technical functionalities and criteria necessary to meet each business requirement:

ID	Requirement Description	Acceptance Criteria	Dependencies	Implementation Notes	Testing Guidelines	Priority	Status	Owner/Assigned To
TR1	Seamless Shopping Cart and Checkout Experience	Users can add products to the cart and view the cart contents.	Product Database, UI	Ensure proper integration with the product database.	Test adding products, updating quantities, and viewing the cart.	High	Pending	Development Team
TR2	Support for Multiple Payment Methods	The system should allow users to choose from various payment methods.	Payment Gateways	Integrate with different payment gateways (e.g., credit card, PayPal).	Test different payment methods and ensure successful transactions.	High	Pending	Development Team
TR3	Apply and Validate Discount Codes	Users can apply discount codes during checkout.	Discount Service	Validate discount codes and calculate correct order totals.	Test applying valid and invalid discount codes during checkout.	High	Pending	Development Team
TR4	Email Confirmation after Purchase	Users should receive an email confirmation after completing a purchase.	Email Service	Ensure integration with the email service for order confirmations.	Verify that users receive accurate order confirmation emails.	High	Pending	Development Team
TR5	Error Handling and User Notifications	The system should provide clear error messages and	Logging, UI	Implement logging for errors and display user-friendly messages.	Test scenarios that trigger errors and ensure users receive	High	Pending	Development Team

ID	Requirement Description	Acceptance Criteria	Dependencies	Implementation Notes	Testing Guidelines	Priority	Status	Owner/Assigned To
		notifications to users.			appropriate notifications.			
TR6	Performance Optimization during High Traffic Conditions	The system should handle high traffic without slowdowns or errors.	Load Balancing, Scalability	Optimize server infrastructure for scalability and performance.	Perform load testing under high traffic conditions to ensure system stability.	High	Pending	Infrastructure Team
TR7	Inventory Management for Out-of-Stock Items	The system should prevent checkout for out-of-stock items.	Product Database	Integrate with inventory management to track product availability.	Test scenarios where users attempt to purchase out-of-stock items.	High	Pending	Development Team
TR8	Secure Handling of Invalid Shipping Addresses	Users should be prompted for a valid shipping address during checkout.	Address Validation Service	Implement validation for shipping addresses.	Test entering invalid shipping addresses and verify the system response.	High	Pending	Development Team
TR9	Integration with External Payment Gateway for Credit Card Payments	Credit card payments should be securely processed via an external gateway.	External Payment Gateway	Ensure proper integration with the selected external payment gateway.	Test credit card payments and verify transactions on the external gateway.	High	Pending	Development Team

In this technical requirement table:

- **Requirement ID:** Corresponds to the Business Requirement ID for easy reference.
- **Requirement Description:** Aligns with the business requirement, providing a detailed technical perspective.
- **Acceptance Criteria:** Clear and measurable criteria to determine if the technical requirement has been implemented correctly.
- **Dependencies:** External factors or components that the technical requirement relies on.
- **Implementation Notes:** Guidance or specific instructions for the development team regarding how the requirement should be implemented.
- **Testing Guidelines:** Instructions for the testing team on how to verify and validate the technical requirement.
- **Priority:** Indicates the priority of the technical requirement.
- **Status:** Reflects the current status of the technical requirement.

- **Owner/Assigned To:** The individual or team responsible for implementing or overseeing the technical requirement.

Below is a comprehensive test case table for the provided technical requirements. Each test case includes relevant details such as Test Case ID, Test Case Description, Test Steps, Expected Result, Priority, Status, and Owner/Assigned To:

Test Case Table:

Test Case ID	Test Case Description	Test Steps	Expected Result	Priority	Status	Owner/Assigned To
TC1	Seamless Shopping Cart and Checkout Experience	1. Navigate to the product catalog.	Product is added to the cart with correct quantity.	High	Pending	Development Team
		2. Select a product and add it to the cart.	Cart displays the selected product accurately.			
		3. Navigate to the shopping cart.	Cart shows the added product.			
		4. Verify the added product in the cart.	Cart displays the accurate product and quantity.			
TC2	Support for Multiple Payment Methods	1. Proceed to checkout after adding products.	System offers various payment methods (e.g., credit card, PayPal).	High	Pending	Development Team
		2. Choose different payment methods.	Successful transactions using different payment methods.			
TC3	Apply and Validate Discount Codes	1. Add products to the cart and proceed to checkout.	Enter a valid discount code.	High	Pending	Development Team
		2. Enter a valid discount code.	Discount is applied correctly to the order total.			
		3. Attempt to apply an invalid discount code.	System rejects the invalid discount code.			

Test Case ID	Test Case Description	Test Steps	Expected Result	Priority	Status	Owner/Assigned To
TC4	Email Confirmation after Purchase	1. Complete the checkout process.	Check the associated email account for an order confirmation email.	High	Pending	Development Team
TC5	Error Handling and User Notifications	1. Trigger an error during checkout.	Clear error message is displayed.	High	Pending	Development Team
		2. Complete a successful transaction.	Users receive appropriate notifications for successful transactions.			
TC6	Performance Optimization during High Traffic Conditions	1. Simulate high traffic on the platform.	System handles high traffic effectively with minimal slowdowns.	High	Pending	Infrastructure Team
		2. Monitor system response times.				
TC7	Inventory Management for Out-of-Stock Items	1. Attempt to purchase an out-of-stock item.	System prevents checkout for out-of-stock items.	High	Pending	Development Team
TC8	Secure Handling of Invalid Shipping Addresses	1. Enter an invalid shipping address.	System prompts for a valid shipping address.	High	Pending	Development Team
TC9	Integration with External Payment Gateway for Credit Card Payments	1. Choose credit card as the payment method.	Credit card payments are securely processed via the external gateway.	High	Pending	Development Team
		2. Enter valid credit card information.	Verify the transaction details on the external payment gateway.			

In this table:

- **Test Case ID:** Unique identifier for each test case.
- **Test Case Description:** Describes the nature of the test case.

- **Test Steps:** Steps to execute the test case.
- **Expected Result:** The expected outcome when the test case is executed.
- **Priority:** The priority assigned to the test case.
- **Status:** The current status of the test case.
- **Owner/Assigned To:** The individual or team responsible for the test case.

Traceability Matrix:

Test Case ID	Business Requirement ID(s)	Technical Requirement ID(s)
TC1	BR1, BR2	TR1
TC2	BR2	TR2
TC3	BR3	TR3
TC4	BR4	TR4
TC5	BR5	TR5
TC6	BR6	TR6
TC7	BR7	TR7
TC8	BR8	TR8
TC9	BR9	TR9

In this matrix:

- **Test Case ID:** Unique identifier for each test case.
- **Business Requirement ID(s):** Corresponds to the Business Requirement ID(s) associated with each test case.
- **Technical Requirement ID(s):** Corresponds to the Technical Requirement ID(s) associated with each test case.

ShoppingCartApp.cs

```
using System;
using System.Collections.Generic;

public class ShoppingCart
{
    private List<string> cartItems;

    public ShoppingCart()
    {
        cartItems = new List<string>();
    }

    public void AddToCart(string product)
    {
        cartItems.Add(product);
    }

    public List<string> ViewCart()
    {
        return cartItems;
```

```
    }

    public bool Checkout()
    {
        // Simulate successful checkout
        return true;
    }

    public bool ApplyDiscountCode(string code)
    {
        // Simulate discount code validation
        return code == "VALIDCODE";
    }
}
```

Unit tests:

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using System.Collections.Generic;

[TestClass]
public class ShoppingCartTests
{
    [TestMethod]
    public void Test_AddToCart()
    {
        // Arrange
        var shoppingCart = new ShoppingCart();

        // Act
```

```
    shoppingCart.AddToCart("ProductA");

    // Assert
    CollectionAssert.Contains(shoppingCart.ViewCart(), "ProductA");
}

[TestMethod]
public void Test_Checkout_Success()
{
    // Arrange
    var shoppingCart = new ShoppingCart();

    // Act
    var result = shoppingCart.Checkout();

    // Assert
    Assert.IsTrue(result);
}

[TestMethod]
public void Test_ApplyDiscountCode_ValidCode()
{
    // Arrange
    var shoppingCart = new ShoppingCart();

    // Act
    var result = shoppingCart.ApplyDiscountCode("VALIDCODE");

    // Assert
    Assert.IsTrue(result);
```

```
    }

    [TestMethod]
    public void Test_ApplyDiscountCode_InvalidCode()
    {
        // Arrange
        var shoppingCart = new ShoppingCart();

        // Act
        var result = shoppingCart.ApplyDiscountCode("INVALIDCODE");

        // Assert
        Assert.IsFalse(result);
    }
}
```

Program.cs

```
class Program
{
    static void Main(string[] args)
    {
        // Instantiate the ShoppingCart class
        var shoppingCart = new ShoppingCart();

        // Add products to the cart
        shoppingCart.AddToCart("ProductA");
        shoppingCart.AddToCart("ProductB");

        // View the cart
```

```
Console.WriteLine("Cart Contents:");
foreach (var item in shoppingCart.ViewCart())
{
    Console.WriteLine(item);
}

// Apply discount code
string discountCode = "VALIDCODE";
bool discountApplied = shoppingCart.ApplyDiscountCode(discountCode);

if (discountApplied)
{
    Console.WriteLine($"Discount code '{discountCode}' applied successfully.");
}
else
{
    Console.WriteLine($"Invalid discount code: '{discountCode}'.");
}

// Perform checkout
bool checkoutResult = shoppingCart.Checkout();

if (checkoutResult)
{
    Console.WriteLine("Checkout successful!");
}
else
{
    Console.WriteLine("Checkout failed. Please try again.");
}
```

```
    }
}
```