ITCS473 Project Assignment 2 Testing for Agile Software Project

Unit Testing

Test Suite 1: testHandleAddToCart – ISP Interface-based ECC
Goal: Test the functionality of adding an item to the cart

1. Identify testable functions
    - Testable function: handleAddToCart()
2. Identify parameters, return types, return values, and exceptional behavior
    - Parameters:
        - product: Product | null,
        - selectedAddOns: AddOn[],
        - quantity: number,
        - displayTotalPrice: number,
        - addToCart: (item: CartItemType) => number, // returns cart length or updated cart
        - category = “”
        - sweetness = ""
        - type = ""
    - Return type: object
    - Return value:
        - { success: false, message: "Product not found" } if the product is null
        - { success: true, cartLength: number } if adding a product to the cart succeeds, where cartLength represents the updated cart size.
    - Exceptional behavior:
        - If the product to be added to the cart is null, it will return { success: false, message: "Product not found" }
        - If the quantity of the product to be added to the cart is invalid (0 or negative), it will return { success: false, message: "Product quantity is invalid" }
        - If the total price of the product to be added to the cart is invalid (0 or negative), it will return { success: false, message: "Product total price is invalid" }
3. Model the input domain

- Develop characteristics
    - C1: The product is null
    - C2: The selectedAddOns are empty
    - C3: The value of quantity
    - C4: The value of displayTotalPrice
    - C5: The category is empty
    - C6: The sweetness is empty

- C7: The type is empty

- Partition characteristic

Characteristic	b1	b2	b3
C1: The product is null	True	False	
C2: The selectedAddOns are empty	True	False	
C3: The value of quantity	Quantity <= 0	Quantity > 0	Quantity is not a number
C4: The value of displayTotalPrice	displayTotalPrice <= 0	displayTotalPrice > 0	displayTotalPrice is not a number
C5: The category is empty	True	False	
C6: The sweetness is empty	True	False	
C7: The type is empty	True	False	

- Identify (possible) value

Characteristic	b1	b2	b3
C1: The product is null	null	product = { id: 101, Drink_Name: "Americano", Description: "A rich espresso with added water for a smooth coffee.", Price: { hotPrice: 50, coldPrice: 55 }, DrinkType: "Hot/Cold", Tag: ["Coffee"],	

		isRecommended: true, img_src: "/images/americano.png", category: "Beverage", AddOns: [{ name: "Oat Milk", price: 10 }], };	
C2: The selectedAddOns are empty	[{}]	[{ name: "Oat Milk", price: 10 }]	
C3: The value of quantity	0	1	NaN
C4: The value of displayTotalPrice	0	60	NaN
C5: The category is empty	“”	"Beverage"	
C6: The sweetness is empty	“”	"50%"	
C7: The type is empty	“”	"Hot"	

4. Combine partitions to define test requirements
    - ECC Approach
        - Test requirements:
        - Number of tests = 3

(C1b1, C2b1, C3b1, C4b1, C5b1, C6b1, C7b1) =
product = null;
selectedAddOns = [{}];
quantity = 0;
displayTotalPrice = 0;

```
        sweetness = "";
        type = “”;
(C1b2, C2b2, C3b2, C4b2, C5b2, C6b2, C7b2) =
        product = {
           id: 101,
           Drink_Name: "Americano",
           Description: "A rich espresso with added water for a smooth coffee.",
           Price: { hotPrice: 50, coldPrice: 55 },
           DrinkType: "Hot/Cold",
           Tag: ["Coffee"],
           isRecommended: true,
           img_src: "/images/americano.png",
           category: "Beverage",
           AddOns: [{ name: "Oat Milk", price: 10 }],
         };
        selectedAddOns = [{ name: "Oat Milk", price: 10 }];
        quantity = 1;
        displayTotalPrice = 60;
        sweetness = "50%";
        type = “Hot”;
(C1b2, C2b2, C3b3, C4b3, C5b2, C6b2, C7b2) =
        product = {
           id: 101,
           Drink_Name: "Americano",
           Description: "A rich espresso with added water for a smooth coffee.",
           Price: { hotPrice: 50, coldPrice: 55 },
           DrinkType: "Hot/Cold",
           Tag: ["Coffee"],
           isRecommended: true,
           img_src: "/images/americano.png",
           category: "Beverage",
           AddOns: [{ name: "Oat Milk", price: 10 }],
         };
        selectedAddOns = [{ name: "Oat Milk", price: 10 }];
        quantity = NaN;
        displayTotalPrice = NaN;
        sweetness = "50%";
        type = “Hot”;
```

5. Derive test value:

Test Case	Test Value	Expected result

T1 (C1b1, C2b1, C3b1, C4b1, C5b1, C6b1, C7b1)	product = null;
	selectedAddOns = [{}];
	quantity = 0;
	displayTotalPrice = 0;
	sweetness = "";
	type = “”;
		{ success: false, message: "Product not found" }

T2 (C1b2, C2b2, C3b2, C4b2, C5b2, C6b2, C7b2)	product = {
	id: 101,
	Drink_Name: "Americano",
	Description: "A rich espresso with added water for a smooth coffee.",
	Price: { hotPrice: 50, coldPrice: 55 },
	DrinkType: "Hot/Cold",
	Tag: ["Coffee"],
	isRecommended: true,
	img_src: "/images/americano.png",
	category: "Beverage",
	AddOns: [{ name: "Oat Milk", price: 10 }],
	};
	selectedAddOns = [{ name: "Oat Milk", price: 10 }];
	quantity = 1;
	displayTotalPrice = 60;
	sweetness = "50%";
	type = “Hot”;
		{ success: true, cartLength: 1 }

T3 (C1b2, C2b2, C3b3, C4b3, C5b2, C6b2, C7b2)	product = { id: 101, Drink_Name: "Americano", Description: "A rich espresso with added water for a smooth coffee.", Price: { hotPrice: 50, coldPrice: 55 }, DrinkType: "Hot/Cold",	{ success: false, message: "Product quantity is invalid" }

	Tag: ["Coffee"],
	isRecommended: true,
	img_src:
	"/images/americano.png",
	category: "Beverage",
	AddOns: [{ name: "Oat
	Milk", price: 10 }],
	};
	selectedAddOns = [{ name:
	"Oat Milk", price: 10 }];
	quantity = NaN;
	displayTotalPrice = NaN;
	sweetness = "50%";
	type = “Hot”;	

Test Suite 2: testHandleSortChange – ISP Interface-based ECC
Goal: Test the functionality of searching for an item

1. Identify testable functions
    - Testable function: handleSortChange ()
2. Identify parameters, return types, return values, and exceptional behavior
    - Parameters: Product[] updatedProducts, String sortOrder
    - Return type: Product[]
    - Return value: updatedProducts unsorted or sorted by price in ascending or descending.
    - Exceptional behavior:
3. Model the input domain

- Develop characteristics
    - C1: length of updatedProducts array
    - C2: sortOder value
    - C3 = hotPrice value
    - C4 = coldPrice value

- Partition characteristic

Characteristic	b1	b2	b3
C1 = length of updatedProducts	Empty	Single Product	Multiple Products
C2 = sortOder value	Ascending	Descending	Invalid value
C3 = hotPrice value	Valid	Invalid	
C4 = coldPrice value	Valid	Invalid	

- Identify (possible) value

Characteristic	b1	b2	b3
C1 = length of updatedProducts	[ ]	[{ id: 1, name: "Dirty", description: "A rich espresso shot served over	[{ id: 1, name: "Dirty", description: "A rich espresso shot served over

		cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}]	cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and

			steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}]
C2 = sortOder value	"Price Low to High"	"Price High to Low"	"Invalid value"
C3 = hotPrice value	50	-	
C4 = coldPrice value	60	-	

4. Combine partitions to define test requirements
    - BCC Approach
        - Base choice: C1b3, C2b1, C3b1, C4b1
        - Test requirements:
        - Number of tests = 7

(C1b3, C2b1, C3b1, C4b1) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]},
{ id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]},
{ id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")

(C1b3, C2b1, C3b1, C4b2) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor

contrast.", hotPrice: "85", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]},
{ id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]},
{ id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")

(C1b3, C2b1, C3b2, C4b1) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "-", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]},
{ id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "-", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]},
{ id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "-", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")

(C1b3, C2b2, C3b1, C4b1) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]},
{ id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]},
{ id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price High to Low")

(C1b3, C2b3, C3b1, C4b1) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor

contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]},
{ id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]},
{ id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Invalid value")

(C1b1, C2b1, C3b1, C4b1) = Infeasible

(C1b2, C2b1, C3b1, C4b1) = handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "50", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}], "Price Low to High")

5. Derive test value:

Test Case	Test Value	Expected result
T1 (C1b3, C2b1, C3b1, C4b1)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold",	[{ id: 2, ... }, { id: 3, ... }, { id: 1, ... }]

	isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")	
T2 (C1b3, C2b1, C3b1, C4b2)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "-", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")	[{ id: 2, ... }, { id: 3, ... }, { id: 1, ... }]

T3 (C1b3, C2b1, C3b2, C4b1)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "-", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "-", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "-", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price Low to High")	[{ id: 2, ... }, { id: 3, ... }, { id: 1, ... }]
T4 (C1b3, C2b2, C3b1, C4b1)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made	[{ id: 1, ... }, { id: 3, ... }, { id: 2, ... }]

	by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Price High to Low")	
T5 (C1b3, C2b3, C3b1, C4b1)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "85", coldPrice: "90", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}, { id: 2, name: "Americano", description: "An Americano is made by diluting an espresso shot with hot water for a smooth, robust coffee.", hotPrice: "55", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "americano.png", Tag: ["Coffee"]}, { id: 3, name: "Latte", description: "A smooth blend of espresso and steamed milk with a creamy finish.", hotPrice: "60", coldPrice: "65", category: "Drink",	[{ id: 1, ... }, { id: 2, ... }, { id: 3, ... }]

	TypeOfDrinks: "Hot/Cold", isRecommended: false, image: "latte.png", Tag: ["Coffee", "Milk"]}], "Invalid value")	
T6 (C1b2, C2b3, C3b1, C4b1)	handleSortChange ([{ id: 1, name: "Dirty", description: "A rich espresso shot served over cold milk, resulting in a bold flavor contrast.", hotPrice: "50", coldPrice: "60", category: "Drink", TypeOfDrinks: "Hot/Cold", isRecommended: true, image: "latte.png", Tag: ["Coffee","Recommend"]}], "Price Low to High")	[{ id: 1, ... }]

Test Suite 3: testHandleAddOnClick – ISP Functionality-based ACoC
Goal: Test the functionality of toggling the selection state of an add-on in the selected AddOns

Identify testable functions
- Testable function: handleAddOnClick()

Identify parameters, return types, return values, and exceptional behavior
- Parameters:
  - selectedAddOn: AddOn, // new add-on
  - selectedAddOns: AddOn[] // array of current selected add-ons
- Return type: array
- Return value:
  - If the add-on is already selected (exists in the list), it removes it from the selectedAddOns and returns the new list of selectedAddOns.
  - If the add-on is not already selected, it adds it to the selectedAddOns and returns the new list of selectedAddOns.
- Exceptional behavior: -

Model the input domain

- Develop characteristics
  - C1: The initial state of the selectedAddOns list is initially empty.
  - C2: The selectedAddOn already exists in the selectedAddOns list

- Partition characteristic

Characteristic	b1	b2
C1 = The initial state of selectedAddOns list is initially empty	True	False
C2 = The selectedAddOn already exists in the selectedAddOns list	True	False

- Identify (possible) value

Characteristic	b1	b2
C1 = The initial state of selectedAddOns list is initially	selectedAddOns = []	selectedAddOns = [{ name: "Oat Milk",

empty		price: 15 }]
C2 = The selectedAddOn already exists in the selectedAddOns list	selectedAddOn = { name: "Oat Milk", price: 15 } selectedAddOns = [{ name: "Oat Milk", price: 15 }]	selectedAddOn = { name: "Brown Sugar Jelly", price: 15 } selectedAddOns = [{ name: "Oat Milk", price: 15 }]

Combine partitions to define test requirements

- ACoC Approach
  - Test requirements:
  - Number of tests = 2*2 = 4

(C1b1, C2b1) = Infeasible
(C1b1, C2b2) =
selectedAddOn = { name: "Oat Milk", price: 15 }
selectedAddOns = [];
(C1b2, C2b1) =
selectedAddOn = { name: "Oat Milk", price: 15 }
selectedAddOns = [{ name: "Oat Milk", price: 15 }]
(C1b2, C2b2) =
selectedAddOn = { name: "Brown Sugar Jelly", price: 15 }
selectedAddOns = [{ name: "Oat Milk", price: 15 }]

Derive test value:

Test Case	Test Value	Expected result
T2 (C1b1, C2b2)	selectedAddOn = { name: "Oat Milk", price: 15 } selectedAddOns = [];	[{ name: "Oat Milk", price: 15 }]
T3 (C1b2, C2b1)	selectedAddOn = { name: "Oat Milk", price: 15 } selectedAddOns = [{ name: "Oat Milk", price: 15 }]	[ ] (Remove the add-on from the list)
T4 (C1b2, C2b2)	selectedAddOn = { name: "Brown Sugar Jelly", price: 15 }	[{ name: "Oat Milk", price: 15 }, { name: "Brown Sugar Jelly",

	selectedAddOns = [{ name: "Oat Milk", price: 15 }]	price: 15 }]