

# SYSTEM FOR RECORDING STOCK INDIVIDUAL ASSIGNMENT

#### COMP40003

### **Software Development and Application Modelling-2**

#### **CF2321COM**

Prepared by:

Nimsha Fernando - CB011990

Date of submission: 29/8/2023

Instructor: Mr. Upul Rathnayake

# **Table of Contents**

Test Plan	n Table3
Diagram	S5
• U	Ise case diagram for the entire application:
• A	ctivity diagram for use case 1: Add new item of stock
• A	ctivity diagram for use case 2: Add quantity to a stock item7
• S	equence diagram for use case 1: Add new item of stock
• S	equence diagram for use case 2: Add quantity to a stock item 9
• U	IML class diagram for use case 1 and 210

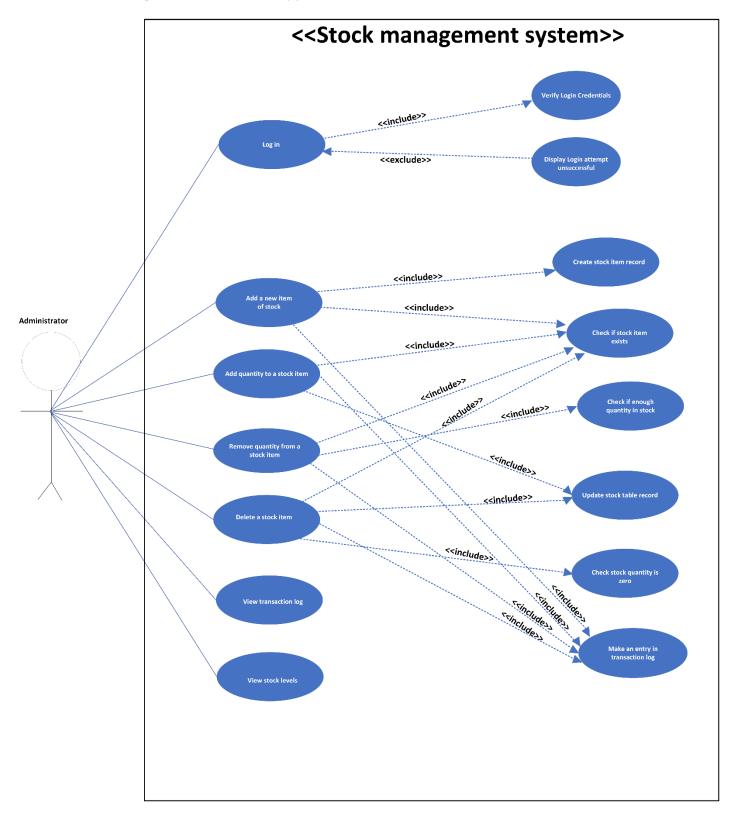
## Test Plan Table

Test Case #	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
1	Ensure that decimal values for stock quantity are not accepted	Stock code: NRF Stock quantity: 0.5	The system should reject decimal quantities	"Please input valid integer for Stock Quantity "	Pass
2	Verify response when entering 0 for stock quantity	Stock code: DBY04, Stock name: glasses Stock quantity: 0	The system should reject a quantity of zero	"Quantity must be a positive number"	Pass
3	Verify response when the stock name field is left empty.	Stock code: BNV3, Stock name: [Empty] Stock quantity: 6	The system should request a valid stock name	"Stock name cannot be empty."	Pass
4	Confirm proper handling of text input in the stock quantity field	Stock code: DBY04, Stock name: glasses Stock quantity: gfg	The system should request a valid integer input	"Please input valid integer for Stock Quantity "	Pass
5	Confirm that a new stock item can be added to the database	Stock code: XYZ01, Stock name: bottle, Stock quantity: 5	The item should be added to the database	"item successfully added"	Pass
6	Ensure that adding a stock item with an existing code is prevented	Stock code: XYZ01, Stock name: bottle, Stock quantity: 10	The system should reject the addition	"A stock item with same stock code or name already exists"	Pass
7	Verify response when adding a different stock name to an existing code	Stock code: XYZ01, Stock name: crayons Stock quantity: 2	The system should reject the use of same stock code to store a stock item with a different name	"error:Duplicate entry 'XYZ01' "	Pass
8	Confirm that the system prevents stock/quantity from being added when stock code field is empty	Stock code: [Empty] Stock name: butter, Stock quantity: 6	The system should request a valid stock code	"Stock code cannot be empty"	Pass

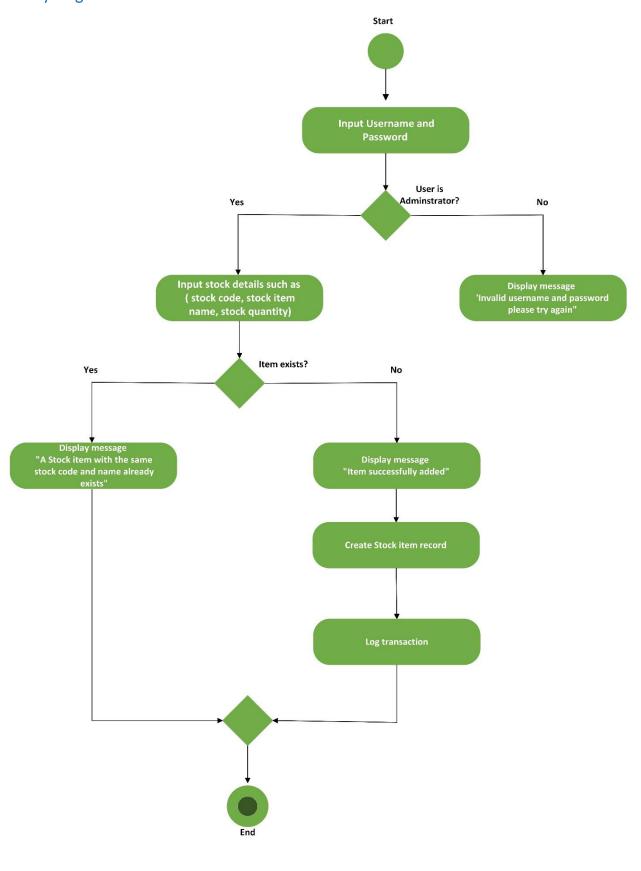
9	Verify response	Stock code: AX20	The system should	No data is	Pass
	when adding	Stock quantity: 10	notify that the item	inserted in to	
	quantity to a non-		does not exist	the database	
	existent stock				
	item				
10	Confirm that	Stock code: XYZ01	The quantity should	"Quantity	Pass
	quantity can be	Stock quantity: 4	be added to the	successfully	
	added to an		existing item	added"	
	existing stock item				
11	Validate	Username: admin	User should be	Logged in	Pass
	successful login	Password:	granted access to the	successfully	
	with valid	nimsha123	system		
	username and				
	password				

# **Diagrams**

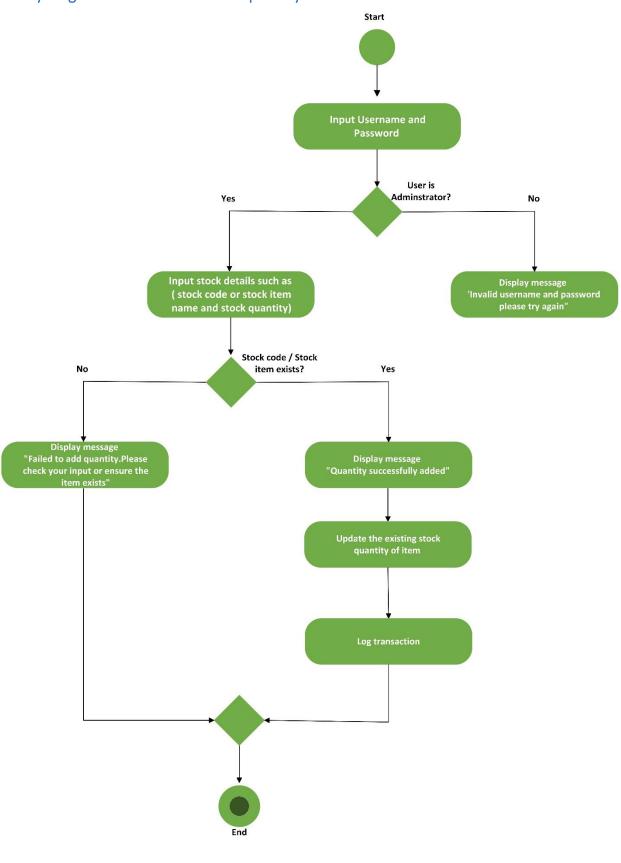
Use case diagram for the entire application:



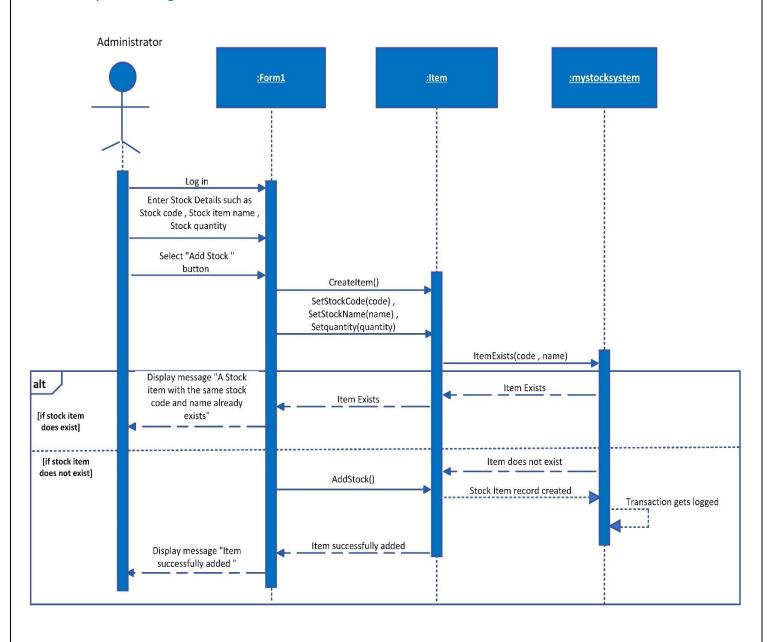
### Activity diagram for use case 1: Add new item of stock



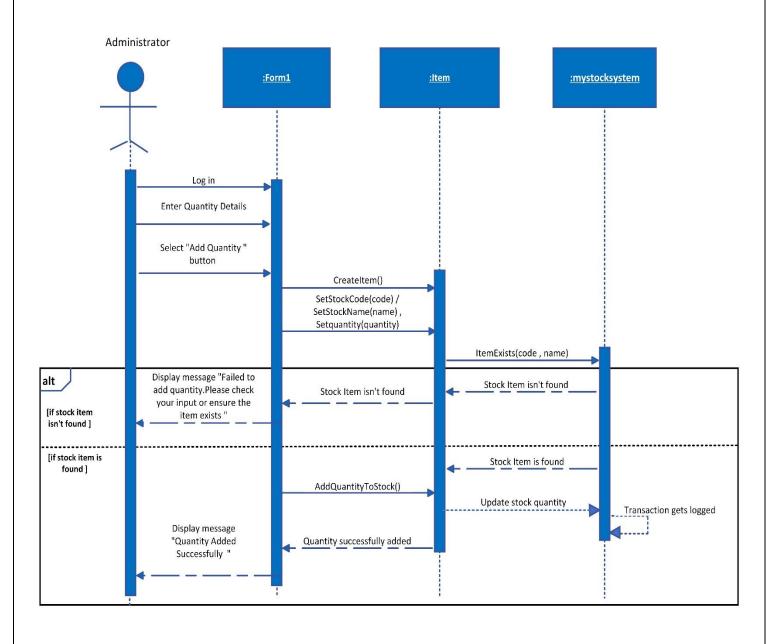
# Activity diagram for use case 2: Add quantity to a stock item



#### Sequence diagram for use case 1: Add new item of stock



### Sequence diagram for use case 2: Add quantity to a stock item



# UML class diagram for use case 1 and 2 Form1.cs - connection: MySqlConnection - connectionString: string + Form1() + AddQuantityButton\_Click() + Createltem(): Item + AddStockButton\_Click() + button\_Click() + label2\_Click() 1 0.1 Transaction.cs DateTime: DateTime - Action: string StockCode: string - Name: string - StockCode: string - Quantity: int - StockItemName: string - QuantityChanged: int - connectionString: string - NewQuantityInStock: int - connectionString: string + SetStockCode(TheStockCode: string) + GetStockCode(): string + SetStockName(TheStockName: string) - GetDateTime(): DateTime + SetDateTime(value: DateTime) + GetStockName(): string + GetAction(): string + SetAction(value: string) + SetQuantity(TheQuantity: int) + GetQuantity(): int + AddStock(): bool + GetStockCode(): string + SetStockCode(value: string) + GetStockItemName(): string + ItemExists(stockCode: string, stockName: string): bool + GetStockName(stockCode: string): string + SetStockItemName(value: string) + AddQuantityToStock(): bool + GetQuantityChanged(): int + SetQuantityChanged(value: int) + GetNewQuantityInStock(): int + GetStockQuantity(stockCode: string): int + UpdateStockQuantity(stockCode: string, newQuantity: int): bool + GetStockNameByCode(stockCode: string), string + LogTransaction(action: string, stockCode: string, stockName: string, quantityAdded: + SetNewQuantityInStock(value: int) + LogTransaction(action: string, stockCode: string, stockName: string, quantityAdded: int, int, newQuantityInStock: int) newQuantityInStock: int): bool