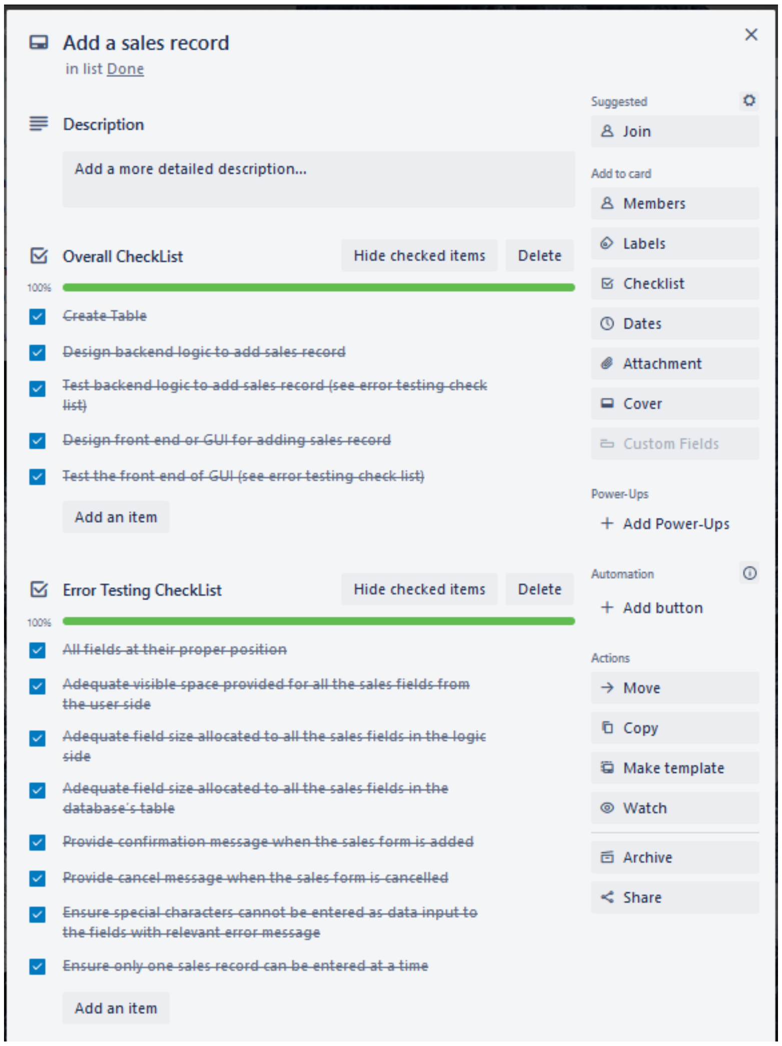
SWE20001: Managing Software Projects

Name: S M Ragib Rezwan

ID: 103172423

Tutor: Naveed Ali | 12.30pm Tuesday | EN310

In 71-72D, I had stated that the team will ensure quality of "Add Sales Record" using ISO25010 characteristic of Usability's sub-characteristic of "User Error Protection". Thus, in figure [1], I have shown the trello board's card for add sales record with complete checklist. There you can see that all check lists have been marked as completed alongside the error testing ones *(which had been mentioned in task 72D).*



Add a sales record

in list Done

Description

Add a more detailed description...

Overall CheckList

100%

- ✓ ~~Create Table~~
- ✓ ~~Design backend logic to add sales record~~
- ✓ ~~Test backend logic to add sales record (see error testing check list)~~
- ✓ ~~Design front end or GUI for adding sales record~~
- ✓ ~~Test the front end of GUI (see error testing check list)~~

Error Testing CheckList

100%

- ✓ ~~All fields at their proper position~~
- ✓ ~~Adequate visible space provided for all the sales fields from the user side~~
- ✓ ~~Adequate field size allocated to all the sales fields in the logic side~~
- ✓ ~~Adequate field size allocated to all the sales fields in the database's table~~
- ✓ ~~Provide confirmation message when the sales form is added~~
- ✓ ~~Provide cancel message when the sales form is cancelled~~
- ✓ ~~Ensure special characters cannot be entered as data input to the fields with relevant error message~~
- ✓ ~~Ensure only one sales record can be entered at a time~~

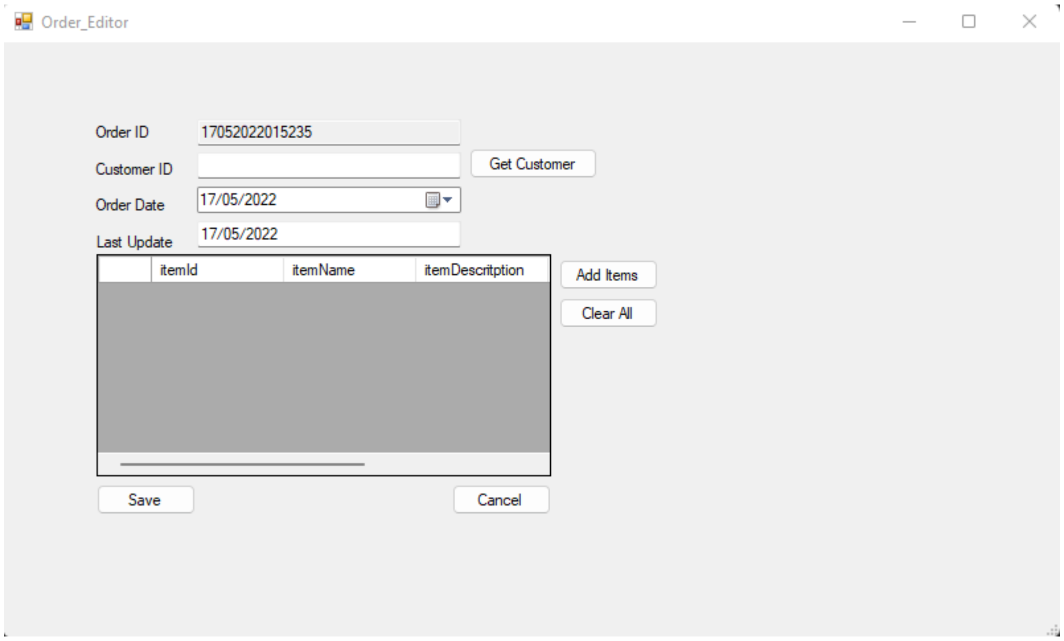
[Figure 1: showing the Trello board card for "add sales record"]

Furthermore, while ticking through the checking for error testing, we had also taken relevant screenshots as evidence and have detailed what they mean for each point.

//must add relevant code sides for all parts (like code for dialogue box size and stuff---ie pic of that properties thing for any button or test size or text on button)!! Or else it would be accepted!!

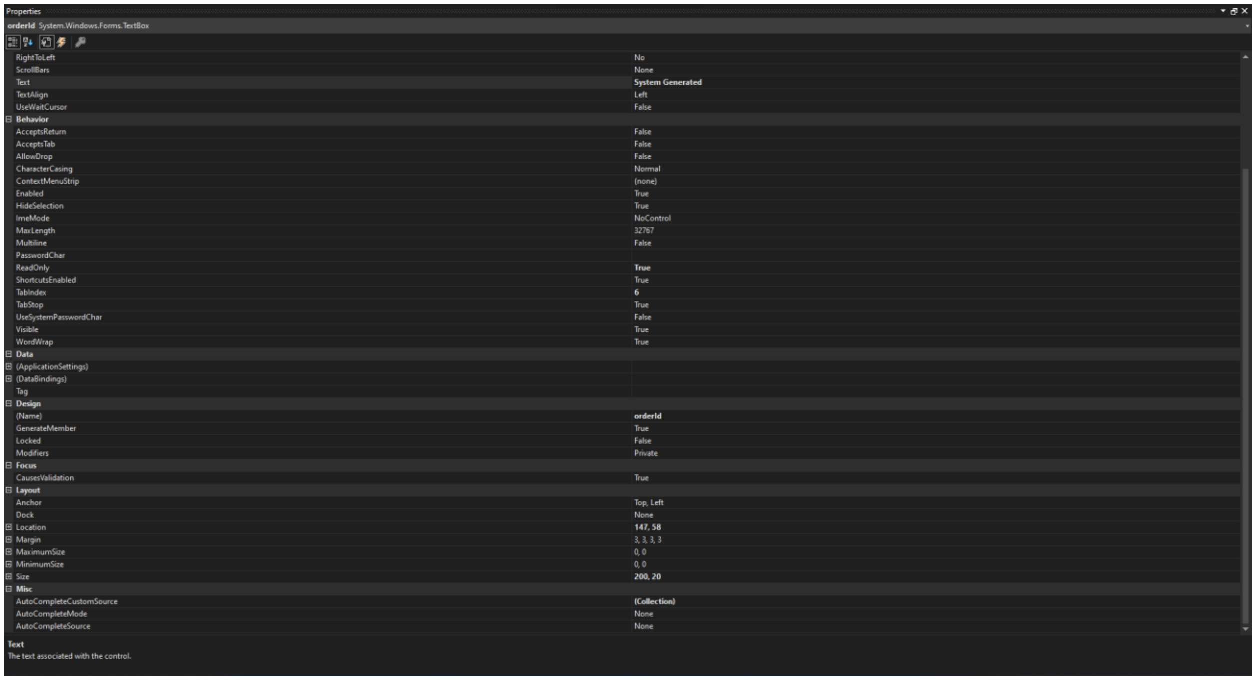
Point 1: Checking all Fields at Proper position:

We checked this in both the GUI and also the backend code parts and ensured that each text box had proper field name beside it (so that user know which data he or she is entering) and that they had proper positioning and spacing [see figure2.1 for box positioning, size and setup details]. This can be seen in figure[2]. Keep in mind, here we have used order and sales interchangeable in the code (as they mean the same thing) and kept the final display name as order as it sounded more professional and user friendly than sales.



Order ID	17052022015235
Customer ID	
Order Date	17/05/2022
Last Update	17/05/2022

[Figure [2]: Picture of GUI of "Add Sales record" (aka Order editor)]



```
Properties
orderId  System.Windows.Forms.TextBox
   RightToLeft                 No
   ScrollBars                  None
   Text                        System Generated
   TextAlign                   Left
   UseWaitCursor               False
⊟ Behavior
   AcceptsReturn               False
   AcceptsTab                  False
   AllowDrop                   False
   CharacterCasing             Normal
   ContextMenuStrip            (none)
   Enabled                     True
   HideSelection               True
   ImeMode                     NoControl
   MaxLength                   32767
   Multiline                   False
   PasswordChar
   ReadOnly                    True
   ShortcutsEnabled            True
   TabIndex                    6
   TabStop                     True
   UseSystemPasswordChar       False
   Visible                     True
   WordWrap                    True
⊟ Data
⊞ (ApplicationSettings)
⊞ (DataBindings)
   Tag
⊟ Design
   (Name)                      orderId
   GenerateMember              True
   Locked                      False
   Modifiers                   Private
⊟ Focus
   CausesValidation            True
⊟ Layout
   Anchor                      Top, Left
   Dock                        None
⊞ Location                     147, 58
⊞ Margin                       3, 3, 3, 3
⊞ MaximumSize                  0, 0
⊞ MinimumSize                  0, 0
⊞ Size                         200, 20
⊟ Misc
   AutoCompleteCustomSource    (Collection)
   AutoCompleteMode            None
   AutoCompleteSource          None
Text
The text associated with the control.
```

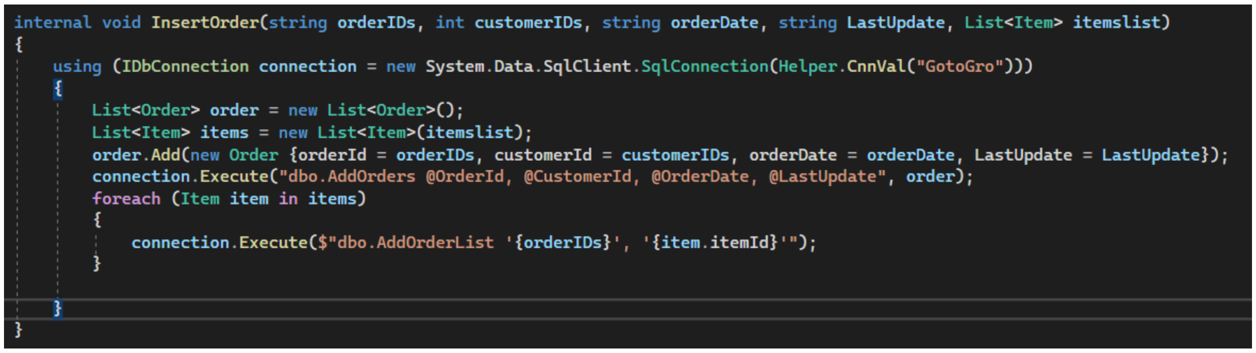
[Figure [2.1]: Picture of configuration for “orderId” textbox set up]

Point 2: Checking adequate visible space allocated to user side:

This has been tested using the same method as Point 1. From there (figure [2]), we can see that there is more than enough space for each fields. So when user enters the data, they will be able to see the full information that has been entered at a glance and will not need to scroll through the text box.

Point 3-4: Checking adequate field sizes allocated to each field on methods and database:

Here we had done this on the code side[3.1] and database itself by using the default variable type with no explicit size restrictions on the fields. This means the fields will always be able to store the complete data and not be truncated (ie putting in a fragment of data due to lack of space instead of full data) in any way what so ever. Thus no field will have any missing data in any way.



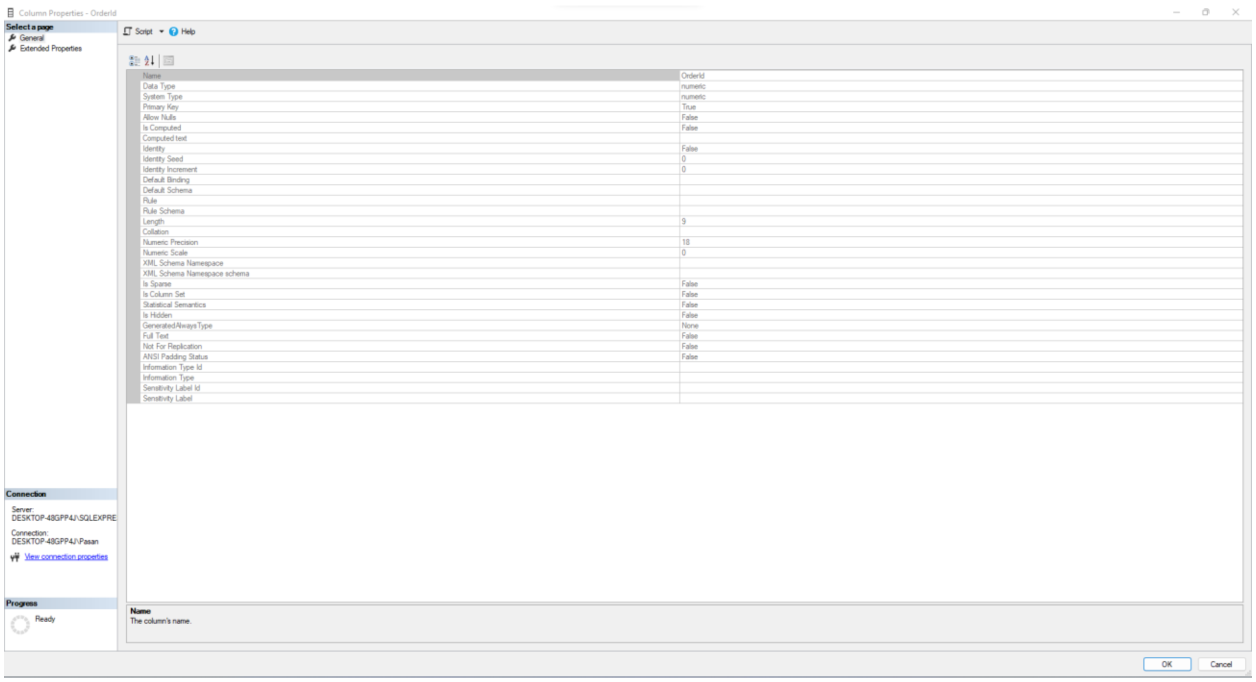
```
internal void InsertOrder(string orderIDs, int customerIDs, string orderDate, string LastUpdate, List<Item> itemslist)
{
    using (IDbConnection connection = new System.Data.SqlClient.SqlConnection(Helper.CnnVal("GotoGro")))
    {
        List<Order> order = new List<Order>();
        List<Item> items = new List<Item>(itemslist);
        order.Add(new Order {orderId = orderIDs, customerId = customerIDs, orderDate = orderDate, LastUpdate = LastUpdate});
        connection.Execute("dbo.AddOrders @OrderId, @CustomerId, @OrderDate, @LastUpdate", order);
        foreach (Item item in items)
        {
            connection.Execute($"dbo.AddOrderList '{orderIDs}', '{item.itemId}'");
        }

    }
}
```

[figure 3.1: Code for method of adding sales record (aka insert order)]

This can be seen in Figure [3.2] where datatype for order id is numeric (as we are assuming only numbers will be entered there and not alphabets) which in the database software used (SSMS - SQL Server Management Studio) can accommodate values from “-10^38 +1 to 10^38-1”, which is more than enough for order id. In figure [3.3], the datatype for customer id is int (as we are assuming only numbers will be entered there and not alphabets) which in the database software used (SSMS - SQL Server Management Studio) can accommodate values from “-2^31 to 2^31-1”, which is more than enough for customer id. In figure [3.4,3.5], the data type for orderdate and lastupdate is nchar (as here we are inputting in dates) which in the database software used (SSMS - SQL Server Management Studio) can accommodate “0 to 4000 characters”, which is more than enough for both dates.

[Note: For all the default field sizes, I have looked it up on “https://www.sqlshack.com/an-overview-of-sql-server-data-types/” and “SSMS documentation” for each data type and thus can assure its validity]



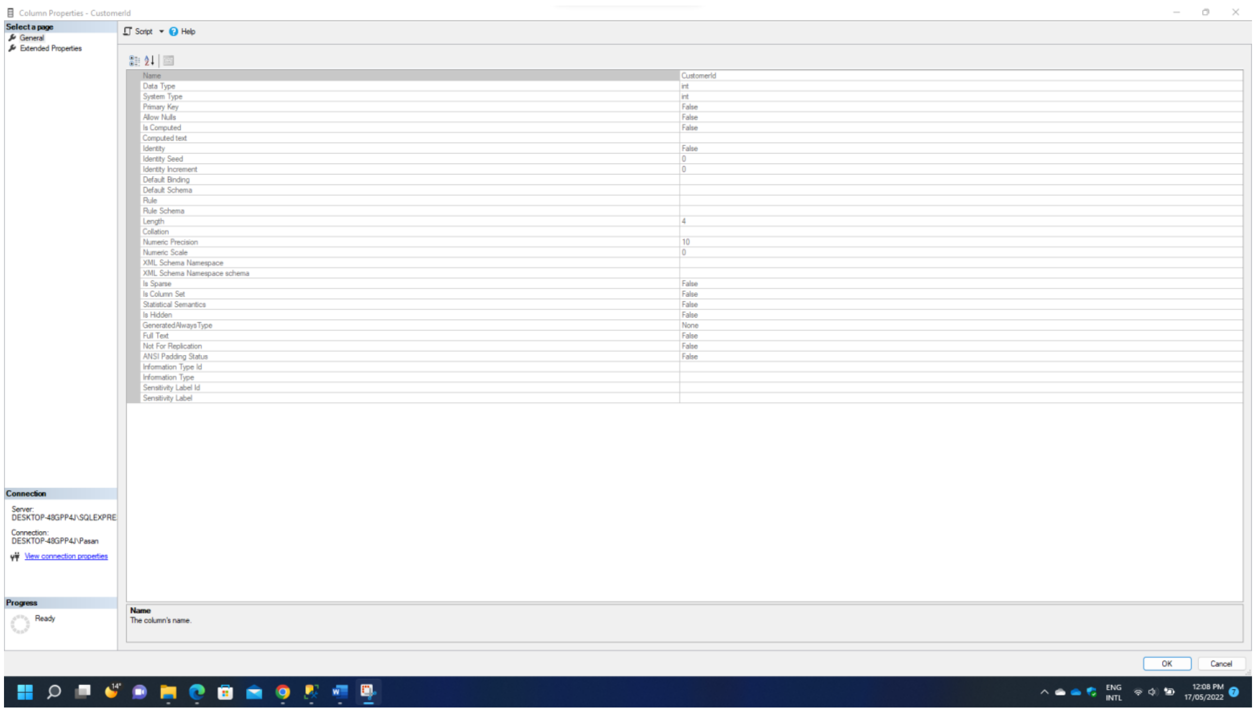
Name	OrderId
Data Type	numeric
System Type	numeric
Primary Key	True
Allow Nulls	False
Is Computed	False
Computed text	
Identity	False
Identity Seed	0
Identity Increment	0
Default Binding	
Default Schema	
Rule	
Rule Schema	
Length	9
Collation	
Numeric Precision	18
Numeric Scale	0
XML Schema Namespace	
XML Schema Namespace schema	
Is Sparse	False
Is Column Set	False
Statistical Semantics	False
Is Hidden	False
GeneratedAlwaysType	None
Full Text	False
Not For Replication	False
ANSI Padding Status	False
Information Type Id	
Information Type	
Sensitivity Label Id	
Sensitivity Label	

Connection
Server:
DESKTOP-48GPP4J\SQLEXPRE
Connection:
DESKTOP-48GPP4J\Pasan
View connection properties

Progress
Ready

Name
The column's name.

[figure [3.2]: Order id properties]



Name	CustomerId
Data Type	int
System Type	int
Primary Key	False
Allow Nulls	False
Is Computed	False
Computed text	
Identity	False
Identity Seed	0
Identity Increment	0
Default Binding	
Default Schema	
Rule	
Rule Schema	
Length	4
Collation	
Numeric Precision	10
Numeric Scale	0
XML Schema Namespace	
XML Schema Namespace schema	
Is Sparse	False
Is Column Set	False
Statistical Semantics	False
Is Hidden	False
GeneratedAlwaysType	None
Full Text	False
Not For Replication	False
ANSI Padding Status	False
Information Type Id	
Information Type	
Sensitivity Label Id	
Sensitivity Label	

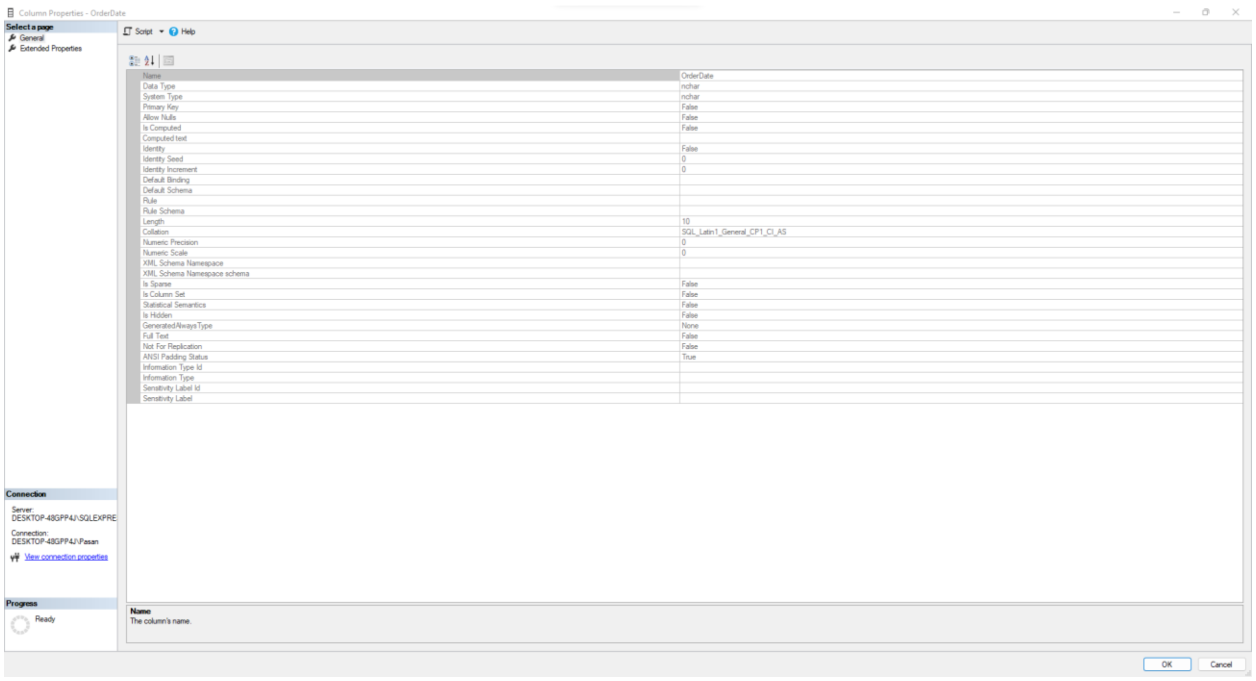
Connection
Server: DESKTOP-48GPP4J\SQLEXPRE
Connection: DESKTOP-48GPP4J\Pasan
View connection properties

Progress
Ready

Name
The column's name.

ENG INTL 12:08 PM 17/05/2022

[figure [3.3]: customer id properties]



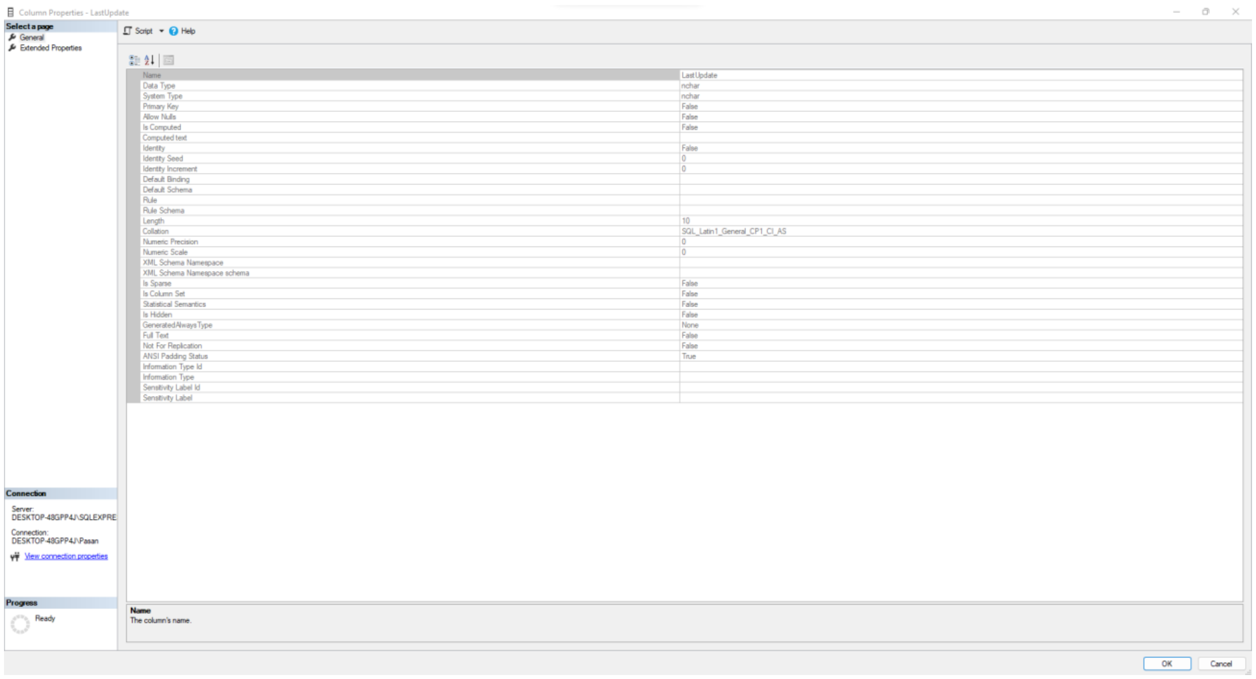
Name	OrderDate
Data Type	nchar
System Type	nchar
Primary Key	False
Allow Nulls	False
Is Computed	False
Computed text	
Identity	False
Identity Seed	0
Identity Increment	0
Default Binding	
Default Schema	
Rule	
Rule Schema	
Length	10
Collation	SQL_Latin1_General_CP1_CI_AS
Numeric Precision	0
Numeric Scale	0
XML Schema Namespace	
XML Schema Namespace schema	
Is Sparse	False
Is Column Set	False
Statistical Semantics	False
Is Hidden	False
GeneratedAlwaysType	None
Full Text	False
Not For Replication	False
ANSI Padding Status	True
Information Type Id	
Information Type	
Sensitivity Label Id	
Sensitivity Label	

Connection
Server: DESKTOP-48GPP4J\SQLEXPRE
Connection: DESKTOP-48GPP4J\Pasan
View connection properties

Progress
Ready

Name
The column's name.

[figure[3.4]: Orderdate field properties]


Column Properties - LastUpdate

Name	LastUpdate
Data Type	nchar
System Type	nchar
Primary Key	False
Allow Nulls	False
Is Computed	False
Computed text	
Identity	False
Identity Seed	0
Identity Increment	0
Default Binding	
Default Schema	
Rule	
Rule Schema	
Length	10
Collation	SQL_Latin1_General_CP1_CI_AS
Numeric Precision	0
Numeric Scale	0
XML Schema Namespace	
XML Schema Namespace schema	
Is Sparse	False
Is Column Set	False
Statistical Semantics	False
Is Hidden	False
GeneratedAlwaysType	None
Full Text	False
Not For Replication	False
ANSI Padding Status	True
Information Type Id	
Information Type	
Sensitivity Label Id	
Sensitivity Label	

Connection

Server:
DESKTOP-48GPP4J\SQLEXPRE

Connection:
DESKTOP-48GPP4J\Paaan

View connection properties

Progress

Ready

Name
The column's name.

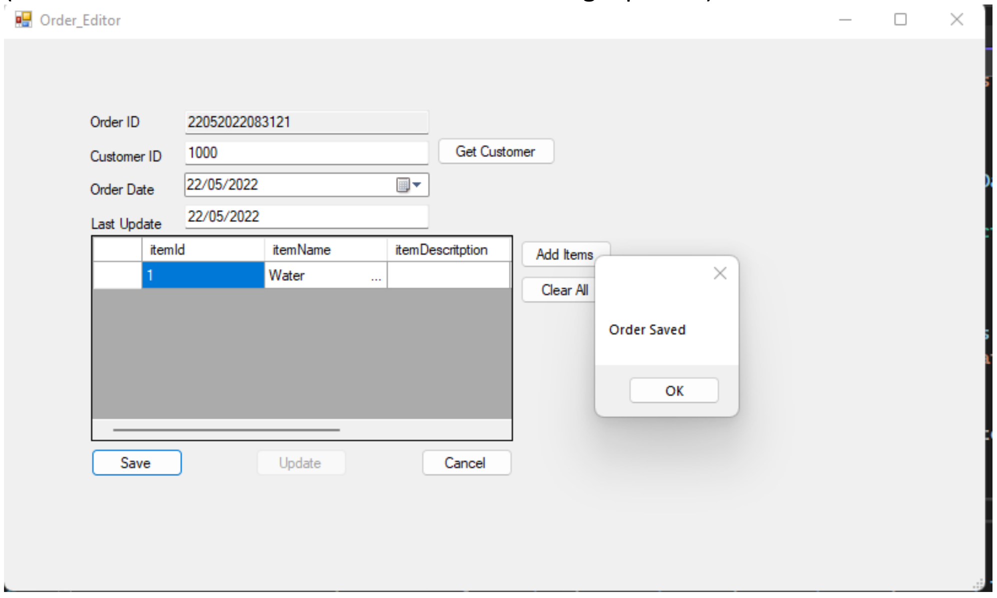
[figure[3.5]: LastUpdate field properties]

Point 5-6: Checking to see if system is providing confirmation message when sales form is added and cancel message when form is cancelled

We checked this in both the GUI and also the backend code parts and ensured that relevant messages popped up when "save" and "cancel" buttons were pressed on the GUI. This is extremely useful as it assures the User that their Sales order has been either:

A) Submitted to the system and hence added to the database table once they click "save" [as seen in figure [4.1]]
B) Not submitted to the system and thus not added to the database table once they click "cancel" [as seen in figure [4.2]]

Hence Users will feel relieved when submitting (when all data is accurate and correct) or cancelling (when a mistake has been made in the data being inputted)

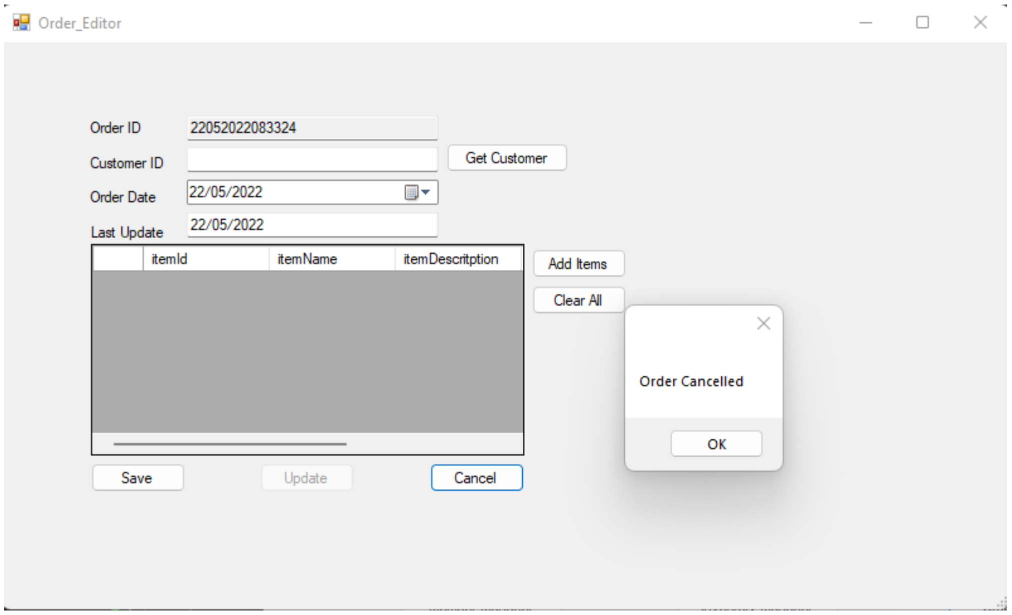


Order ID 22052022083121
Customer ID 1000
Order Date 22/05/2022
Last Update 22/05/2022

	itemId	itemName	itemDescritption
	1	Water	

Order Saved

[figure[4.1]: order saving dialogue box appearing when order is saved]



Order ID 22052022083324
Customer ID
Order Date 22/05/2022
Last Update 22/05/2022

	itemId	itemName	itemDescritption

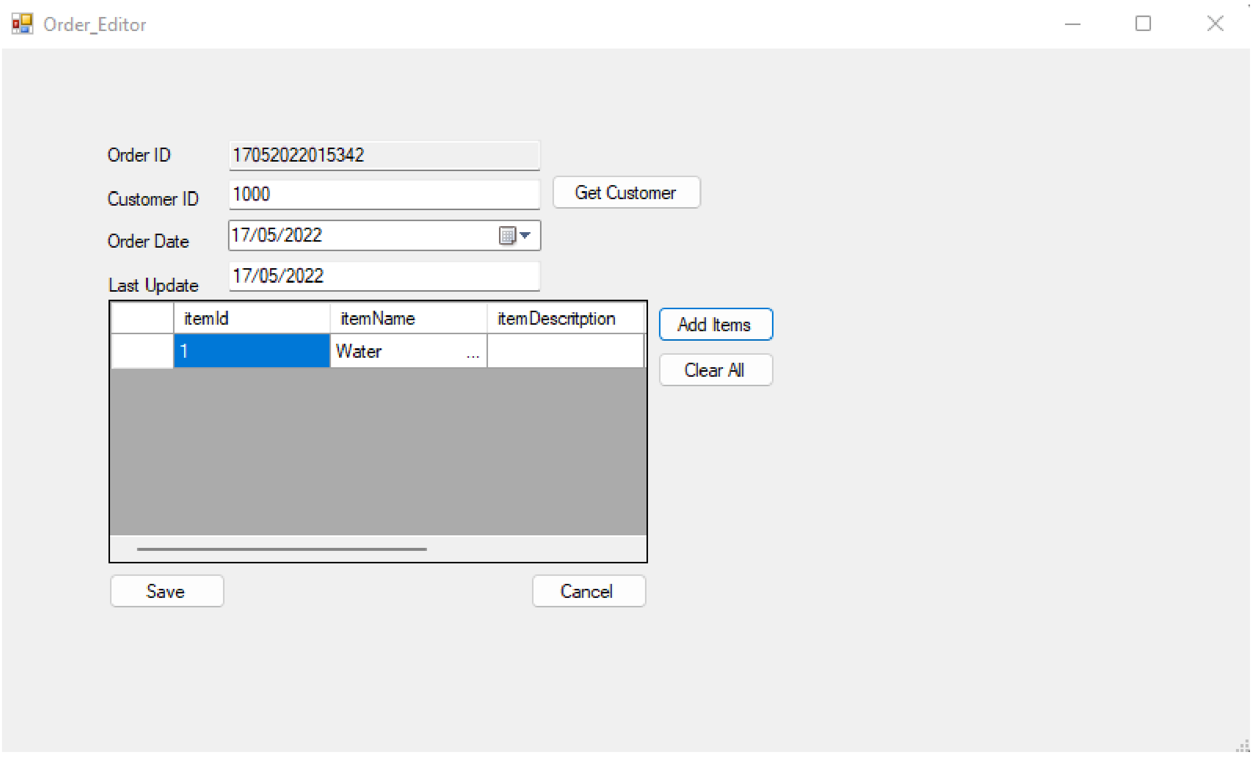
Order Cancelled

[figure[4.1]: order cancelled dialogue box appearing when order is cancelled]

Point 7: Checking to ensure special characters cannot be entered as data inputs with relevant error message

We checked this in both the GUI and also the backend code parts and ensured that special characters cannot be entered by changing the data entry method itself. Now the user doesn't need to type in any values. Instead he just needs to click the relevant buttons beside the fields to select whichever options we wants to enter the data, as you can see in the figure [5].

Thus, sanitization of the data input is no longer needed as user would instead click buttons to add in the data instead of manually inputting the values and thus SQL injection can still be prevented.



Order ID 17052022015342
Customer ID 1000
Order Date 17/05/2022
Last Update 17/05/2022

itemId	itemName	itemDescritption
1	Water	

[Figure [5]: GUI for adding sales records]

Point 8: Checking to ensure only one sales record can be entered at a time

We checked this in both the GUI and also the backend code parts and ensured that only one sales record can be entered at a time from the software. This can be seen in figure [5] where we can only enter a single sales record at a time. This ensures that users don't get overwhelmed and make mistakes in entering the relevant data as they will only be doing it one sales record at a time.

Review and comment on quality:

Based on the results, it can be seen that all 8 testing checkpoints have been fulfilled and the software has passed all of them. Thus currently, total number of all failed user operation for this functionality is 0%. This is less than the 5% that had been taken as the threshold criteria that the software needed to maintain *(see 71D)* and so I would accept that the backlog item is fulfilling its “User Error Protection” quality under ISO25010’s “Usability” category.

What I would do differently to assure the quality requirements are adhered to in the future sprint?

In the current software, we had bypassed the sql injection problem by using a click input method instead of using data input (ie inputting in values via typing). Although this is an interesting and more user friendly solution, it would be better to just use sanitization methods and allow user to input data both ways. That way user will both have their freedom of choice in entering data and also the system can be protected from SQL injection.

Furthermore, in field size allocation, currently we have been using the default sizes allocated by the database and not putting any restriction on size in order to prevent truncation error. Instead of doing this, it would be better to put data size restriction on input from GUI side, on size accepted by method and finally on the database sides. Also, keeping more than 5 characters size for order and customer id and more than 8 characters for date and lastupdate (“dd/mm/yy” format) is not necessary as most regular grocery shops will not have that much transaction or customer records to deal with it.

Moreover, it would be also be better to allow alphanumeric characters for entry for order and customer ids as that is the general practice for most shops and perform more tests that ensure that only those types of data is inputted. Thus, it would be better to keep these in mind while setting up quality requirement tests in future sprints.