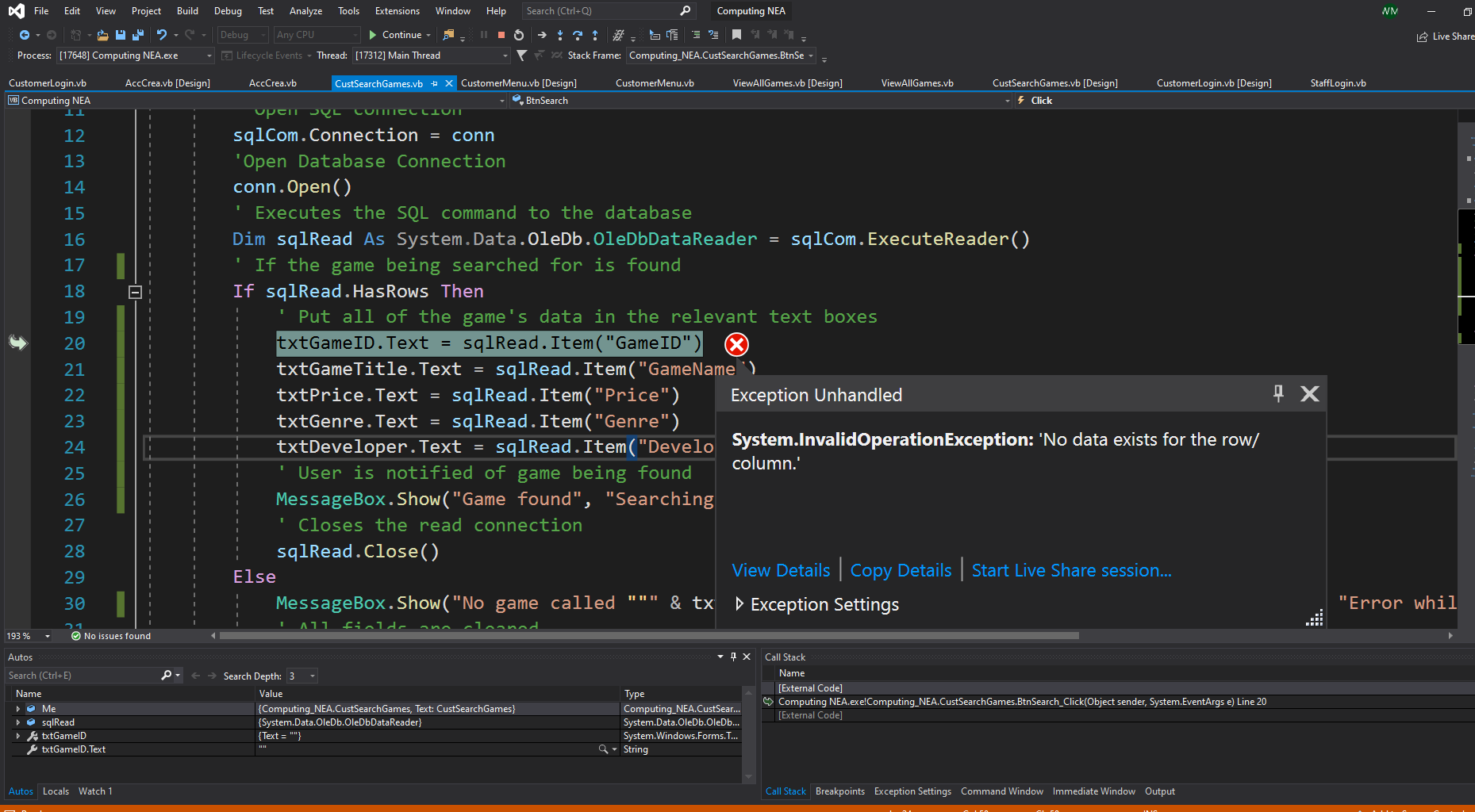
Developmental Testing

Introduction: In this part of the project I will be showing the testing that I have had to do while developing the program for Mongoose Games. I will include screenshots of the problems that I have encountered and the steps that I took to resolve them.

No data exists for the row/column

Whilst developing the part of my system which allows the customer to search for a game by their title, and then if it’s found, its details should be used to populate the relevant text boxes. However, I came across an error shown in the screenshot below. I

I encountered this error when I was attempting to develop a part of the system which produces all of a game’s details when it’s searched by its title, and then these details are populated into the relevant text boxes. This error stated that no data existed for the record that I was trying to fetch.

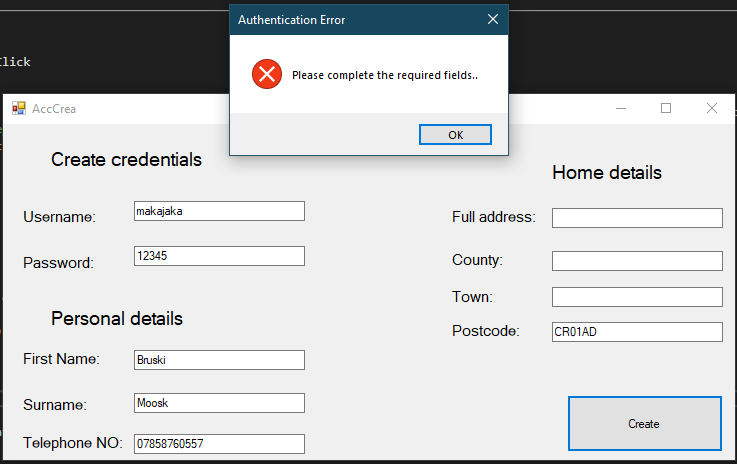


I then corrected this error by realising I needed to add a while loop after the IF clause utilising the sqlRead, to allow each field to be read through and copied into the text boxes until the entire table was searched.

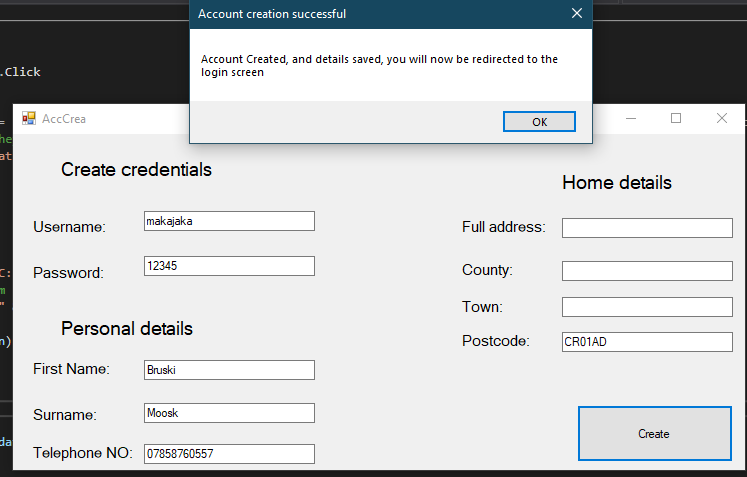
Customer account creation

While testing the customer account creation part of the system, I identified a logic error which was allowing customers to create accounts with missing parts of information.

If a customer would leave out one or more fields of information, an error is then produced which tells them that they are to complete all of the required fields



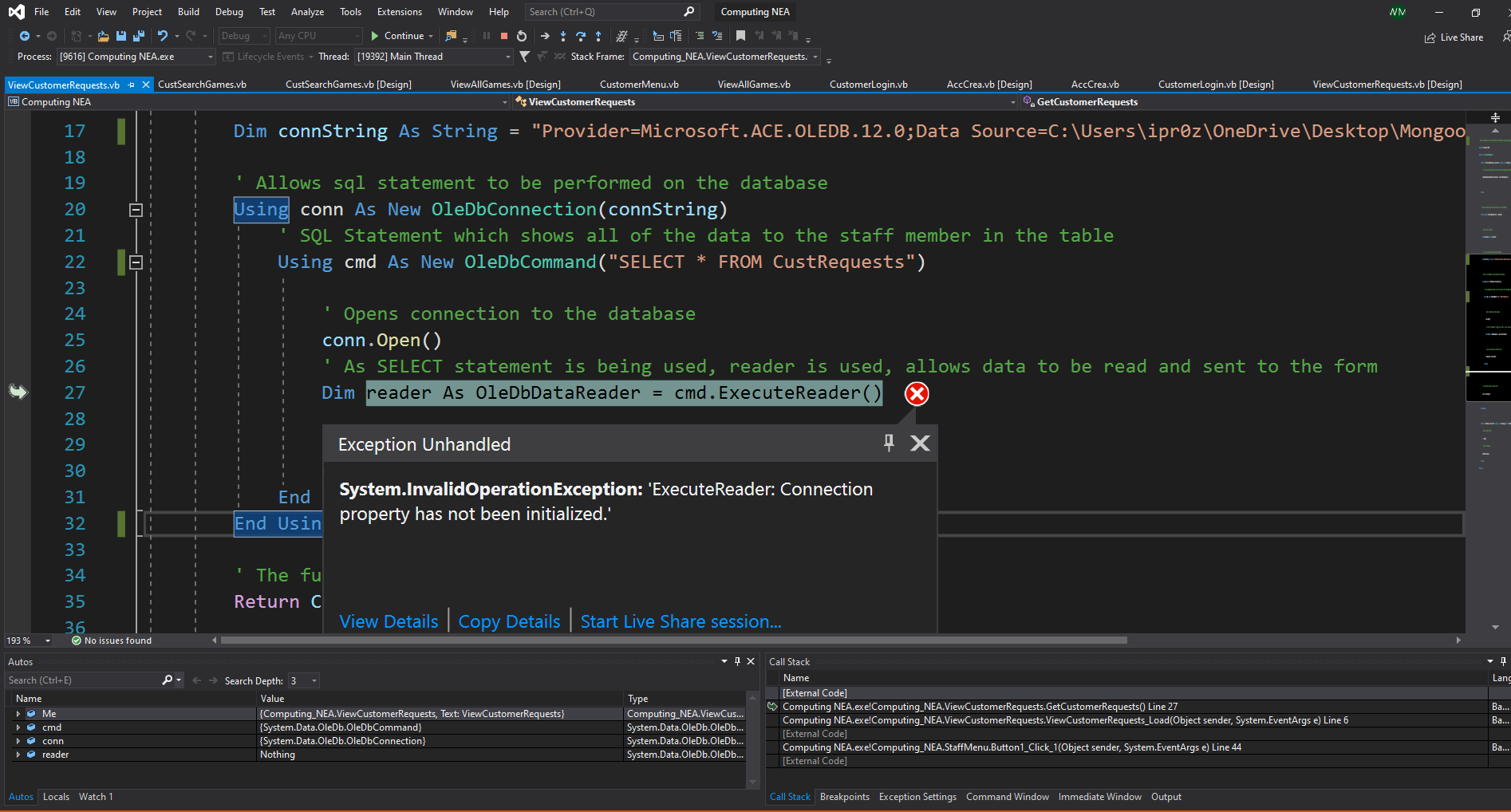
However, the information is still saved to the database.



In the light of this logic error, I then realised I had to use an ELSE statement, so that if they don’t receive an error, their details are saved, but if they do, they cannot create their account until no errors remain.

Connection property has not been initialized

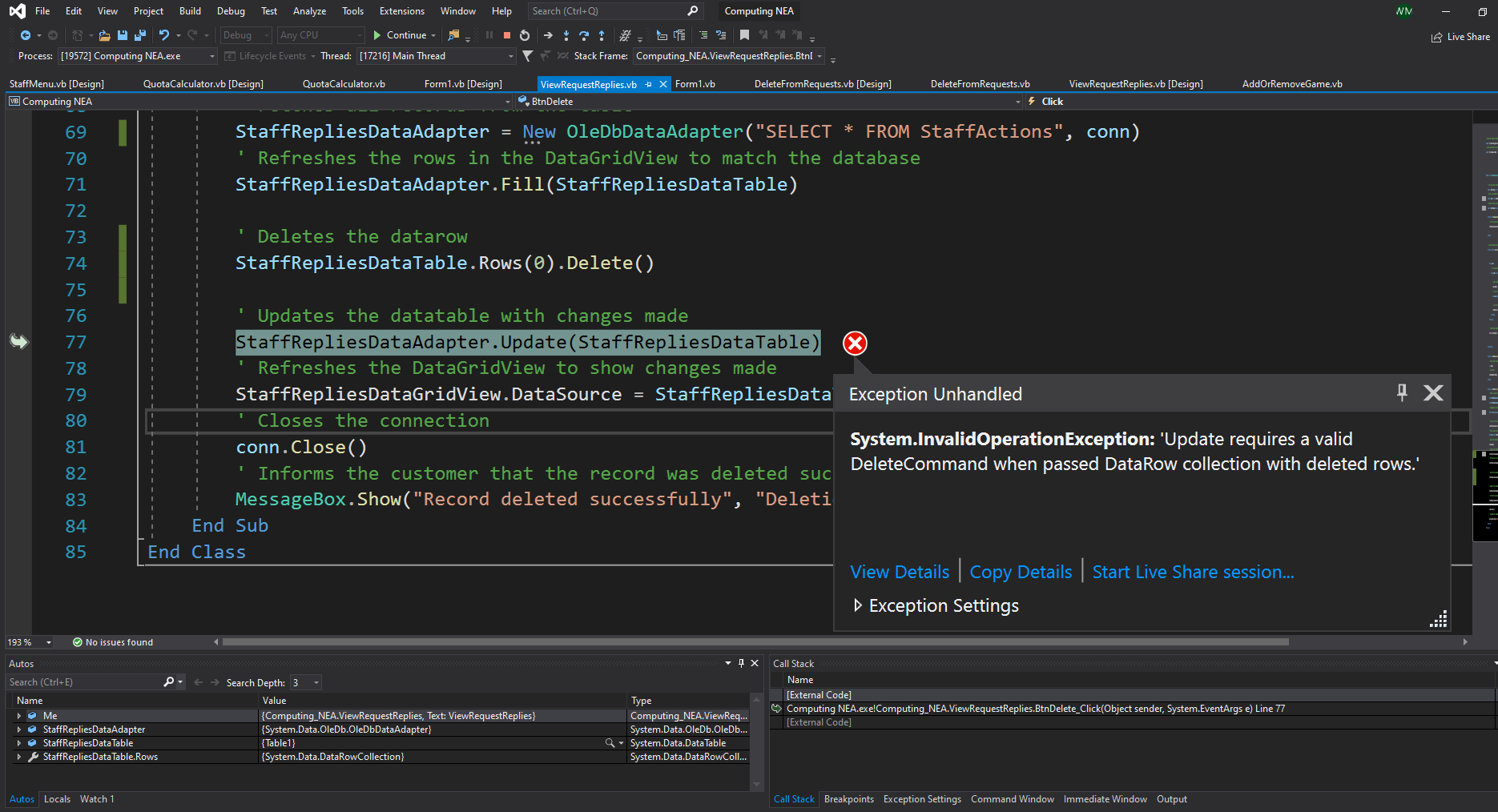
I encountered this error while trying to import a table into a DataGridView, allowing the staff member to view the customer requests from a form. This error stated that the connection property had not been initialized, even though it didn’t highlight the error to the connection string.



I then realised that I had not accompanied the SQL statement with the connection string, to enable the statement to be executed against the defined data source. After doing this, it stopped the error from happening and carried out the function as intended.

Requires a valid delete command

I encountered this error while trying to implement a feature for a customer to be able to delete the oldest reply from a staff member about a request of theirs, via a DataGridView on the form, which would subsequently delete that record from the database. This error stated that, while trying to update the DataGridView after a deletion, it required a valid DeleteCommand.



I corrected this error by realising that I needed to define a commandbuilder, and assign it to the data adapter, to be able to reconcile the changes that had been made to the database. After doing this, the error was no longer produced and allowed the intended function to be carried out.

Testing and Development for user inputs when creating a new account

Currently, there is no validation applied to the data entry of the customers when they create a new account. In order to ensure that only accounts with valid information can be created, their inputs will have to be validated so that no invalid data can enter the system, as well as checking if any fields are left blank.

---------------------------------------------------------------------------------------------------------------------

Imports System.Text.RegularExpressions

Public Class AccCrea

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

' Check if any field is empty and if any of them are empty

If txtPassword.Text = "" Or txtUsername.Text = "" Or txtFirstName.Text = "" Or txtSurname.Text = "" Or txtTelephone.Text = "" Or txtAddress.Text = "" Or txtCounty.Text = "" Or txtTown.Text = "" Or txtPostcode.Text = "" Then

' If so, then the user is notified of this and they are to then fill them in

MessageBox.Show("Please complete all required fields...", "Authentication Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

' Checks if the password is the same as the username

ElseIf txtUsername.Text = txtPassword.Text Then

' If so, user is then to ensure that their username differs from their password

MessageBox.Show("Please ensure that your username is different from your password", "Security warning", MessageBoxButtons.OK, MessageBoxIcon.Error)

' Clears the text boxes to faciliate re-entering of data

txtUsername.Clear()

txtPassword.Clear()

' Checks if the length of the customer's username or password is less than 5

ElseIf txtUsername.Text.Length < 5 Or txtPassword.Text.Length < 5 Then

' If so, they are prompted to make their username/password longer than 5 characters so that their credentials are more sophisticated

MessageBox.Show("Please ensure that your username/password is longer than 5 characters", "Security warning", MessageBoxButtons.OK, MessageBoxIcon.Error)

' Clears the text boxes to faciliate re-entering of data

txtUsername.Clear()

txtPassword.Clear()

' Checks if the phone number entered is less/more than 11 digits long

ElseIf txtTelephone.Text.Length <> 11 Or IsNumeric(txtTelephone.Text) = False Then

' If so, they are informed that they need to ensure their phone number exactly 11 characters long

MessageBox.Show("Please ensure that your phone number is 11 digits long", "Validation error", MessageBoxButtons.OK, MessageBoxIcon.Error)

' Clears the text box to faciliate re-entering of data

txtTelephone.Clear()

' Checks if the postcode entered is 7 or 8 characters long and if it contains a space

ElseIf txtPostcode.Text.Length < 7 Or txtPostcode.Text.Length > 8 Or txtPostcode.Text.Contains(" ") = False Then

' If not, they are informed that they need to enter a valid postcode

MessageBox.Show("Please ensure that your postcode is either 7 or 8 characters long, including a space in between the two parts ", "Validation error", MessageBoxButtons.OK, MessageBoxIcon.Error)

' Checks if any field which shouldn't be composed of digits e.g. firstname contains any digits

ElseIf Regex.Match(txtFirstName.Text, "\d").Success Or Regex.Match(txtSurname.Text, "\d").Success Or Regex.Match(txtCounty.Text, "\d").Success Or Regex.Match(txtTown.Text, "\d").Success Then

' If so, they are prompted to reevaluate this

MessageBox.Show("Please ensure that there are no numbers in fields which don't require them", "Validation error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Else

---------------------------------------------------------------------------------------------------------------------

Shown above is all of the code that I have created for the validation for all of the text boxes in the Create account form. I used a large if statement to validate all of the different text boxes in the form. The first condition of the code consists of checking if any of the text boxes have been left blank, and if so, they are made aware of this and prompted to fill out all of the fields.

The next condition checks if the username is the same as the password, and if so, the user is prompted to change this. I did this in order to ensure that the credentials created by the user are more sophisticated, making them harder to guess, improving the security of their account.

The following condition then checks if the username or password is longer than 5 characters, and if not, the user is prompted to change this. This is to ensure that the credentials created are advanced enough to withstand simple brute force attacks.

The following condition checks if the phone number entered by the user is 11 characters long. It also checks if the phone number entered consists of only digits, as it should. If any of these conditions are met, the user is made aware of this and informed to enter a valid phone number. This ensures data integrity.

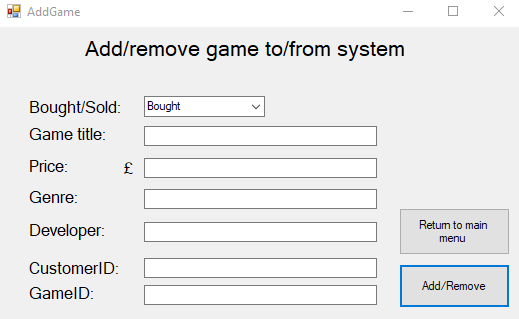
The next condition then checks if the postcode is not 7 or 8 characters long, or if it doesn’t contain a space. If the entered postcode meets any of these conditions, they are notified that they are required to enter a valid postcode, one that has a space in the middle, and comprised of 7 or 8 characters.

Finally, all of the text boxes which should only receive string inputs (Firstname, surname, county, town) are checked to see if the user has inputted any digits in the text boxes. If so, the user is informed of their error and that they need to ensure that text boxes which only warrant a string input, are only comprised of letters.

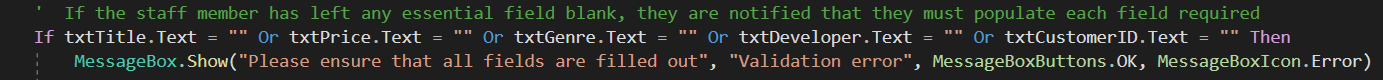
Testing and Development for adding games into the system

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a game is selected as being bought from a customer, but with all other fields left blank, the form just remains blank.



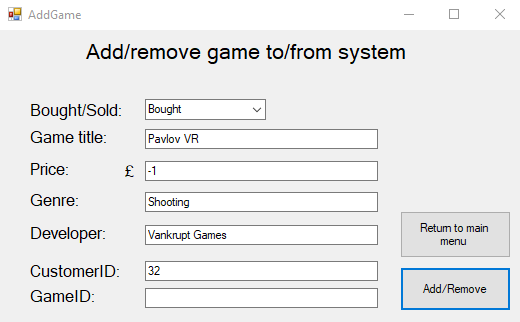
To fix this, I then had to add this code below which ensures that if any essential field is left blank, the game won’t be saved to the database

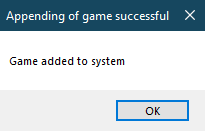


This detects if any fields have been left blank and if so they receive an error message informing them to populate all required text boxes.

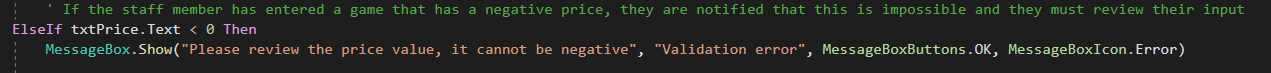
This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a game is selected as being bought from a customer, and the price field is entered erroneously (negative number) it still allows the game to be added to the database with a negative price.





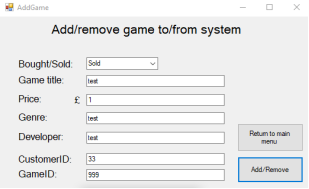
To fix this error, I then had to add this code below which ensures that any price value less than 0 cannot be accepted

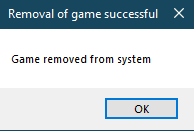


Testing and Development for removing games from the system

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a user chooses to sell a game to a customer which doesn’t exist in the system, an output is received saying that the game was deleted successfully from the system even though it wasn’t nor did it even exist in the first place.





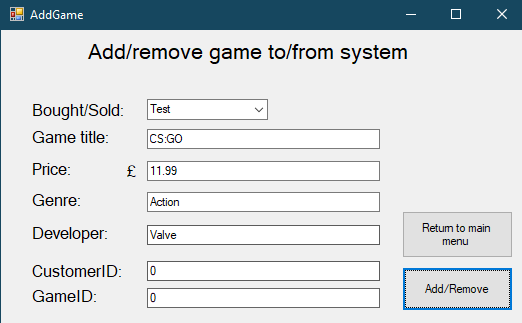
To fix this, I then had to add a condition to the current IF statement that ensured that the game that the user was entering, exists in the database.

...

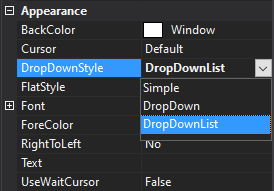
This checks if all of the games information inputted corresponds to an actual game in the system. If so, the process goes ahead and deletes the game from the system.

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a user edits the Bought/Sold dropdown menu, and clicks the Add/Remove button, no error message is displayed, the form just remains blank.



To fix this, I had to ensure that the drop down box values can only be what I have allowed, Bought or Sold. I did this by changing the properties of the ComboBox and changing the DropDownStyle to DropDownList

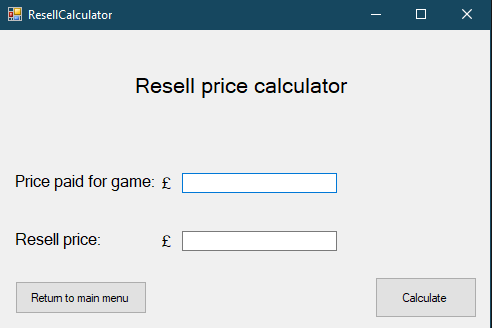


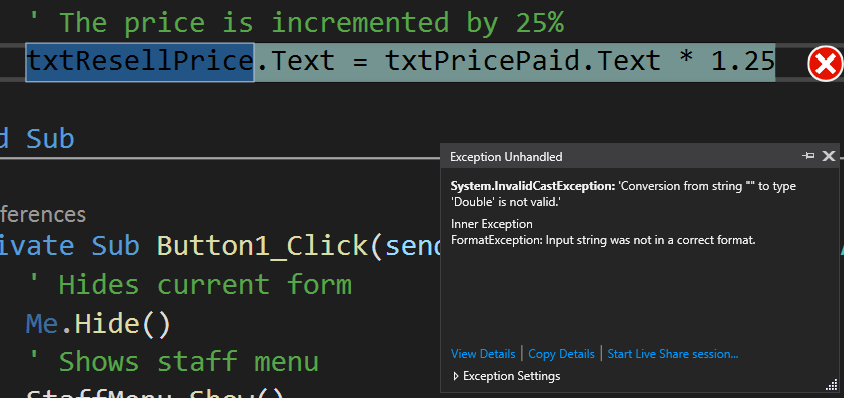
This now ensures that the dropdown box cannot be edited, and so this issue will not reoccur.

Testing and Development for resell price calculator

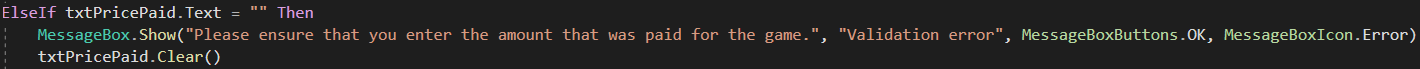
This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a user leaves the “Price paid for game” field blank, and clicks the “Calculate” button, the system crashes as it tries to manipulate the string “” mathematically





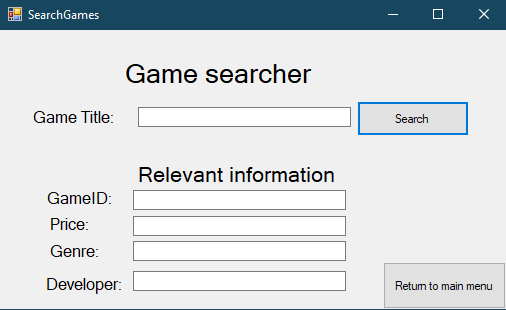
To fix this error, I had to introduce some validation which ensured that the text box holding the price paid value is populated before trying to manipulate the input mathematically.

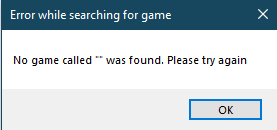


Testing and Development for game search

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently when a user attempts to search for a game, but leaves the Game title field blank, the output is an error message that says no game called “” was found instead of informing the user that the Game Title field cannot be left blank.





To fix this error, I had to introduce some validation which ensured that the text box that receives the game title is populated before trying to search for it.

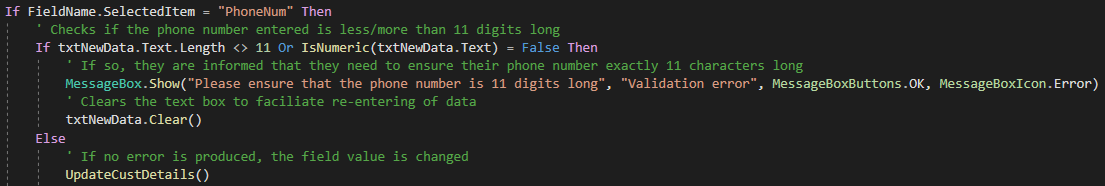


Testing and Development for customer data updater

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

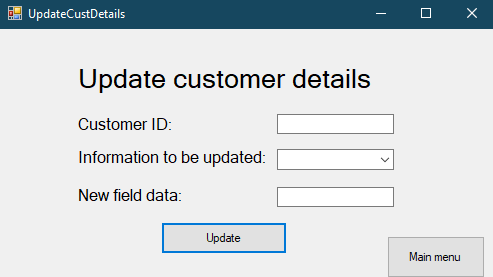
Currently, when a user wants to change a customer’s phone number to an invalid one, they are allowed to.

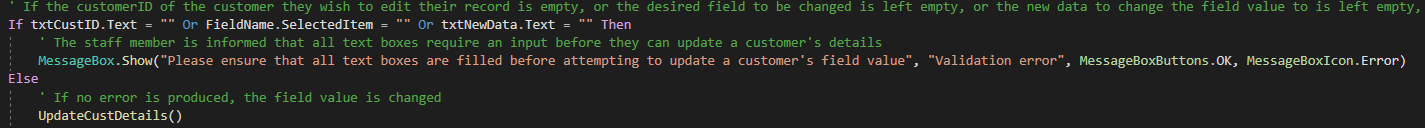
To fix this, I had to include the same validation from the user account creation section, ensuring the phone number is composed of digits and 11 digits long.



This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently, when a user leaves all fields blank, and then presses the update button, nothing happens and the form remains blank.

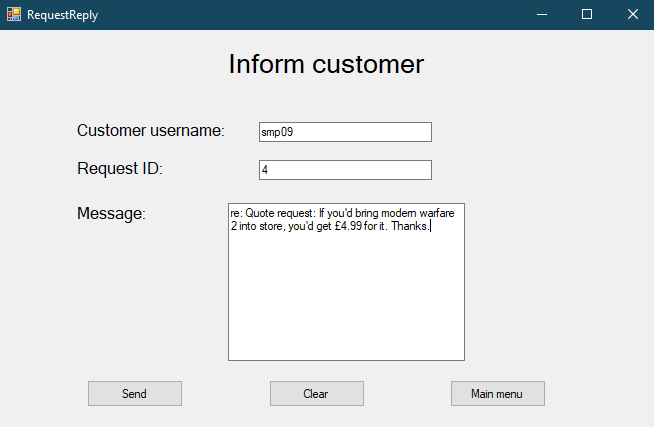


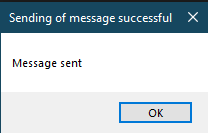
To fix this, I had to include validation that would inform the user if they had left any field blank. Moreover, I made the drop-down box read only, to allow the user to understand they have to choose a field to update.

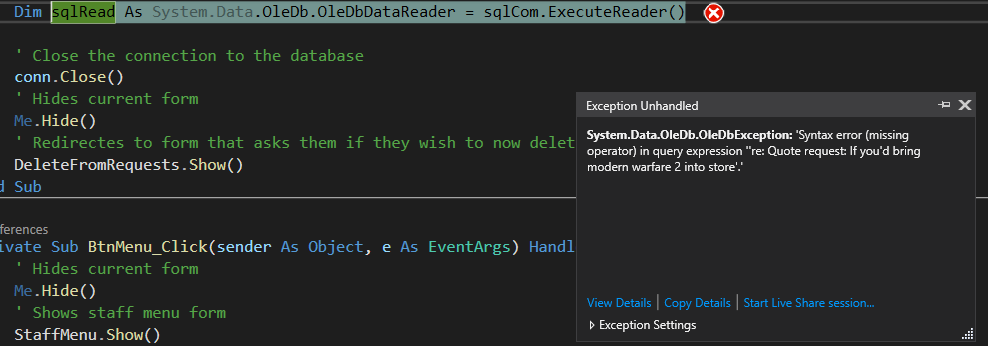
Testing and Development for sending a customer request

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently, when a user wants to reply to a customer request, and the “send” button is pressed, it will tell the user the message has been sent, and then promptly crash. I later found out this has to do with the fact that the message contains an apostrophe.







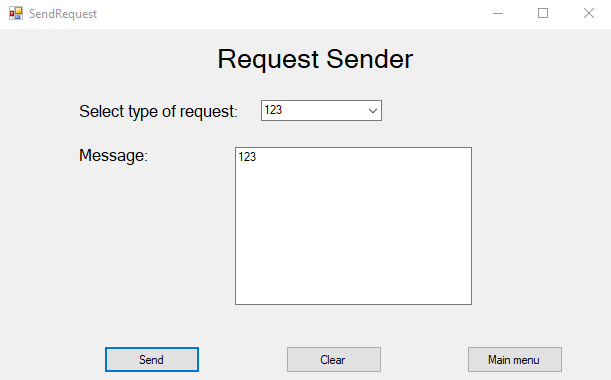
To fix this error, I had to change the SQL statement to be using double quotes “” instead of single quotes ‘’ surrounding the area referring to the data entered into the Message textbox

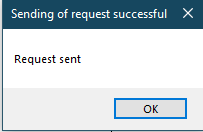


Testing and Development for sending a request to a staff member

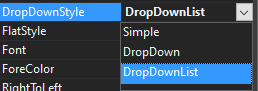
This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

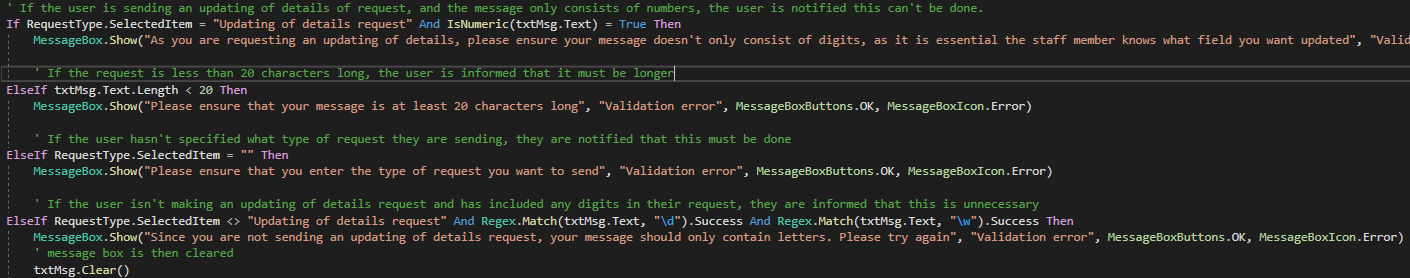
Currently, when a user makes an invalid request to a staff member, the request is still sent.





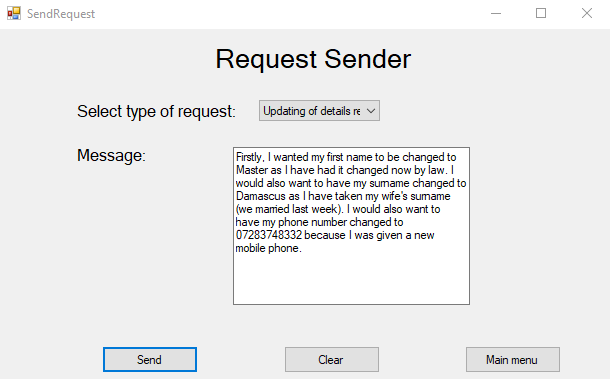
To fix this, I first had to ensure that the type of request dropdown box wouldn’t be able to be edited. I did this by changing the type of the dropdown to dropdownlist. I then had to add validation to ensure that the message entered is more than 20 characters long and that the Type of request was specified using the dropdown menu. Moreover, I added validation to check if the request isn’t an updating of details request, that the message does not need to contain digits, and if it does, the message won’t be sent an if the request is an updating of details of request, that the message must not only be composed of digits.

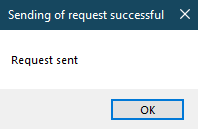


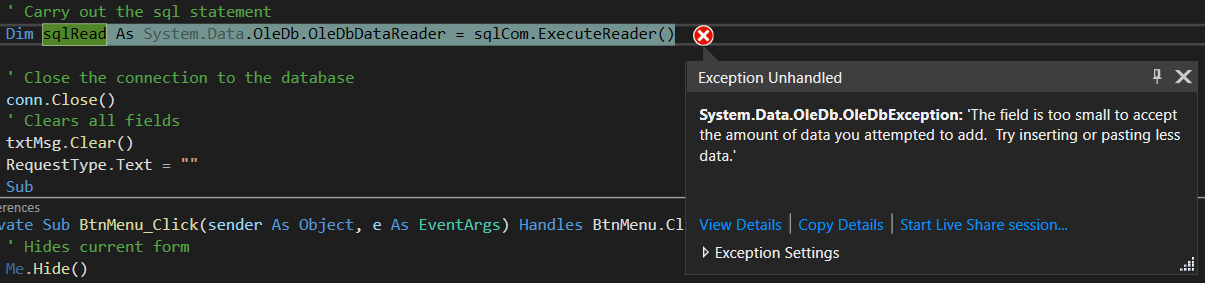


This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

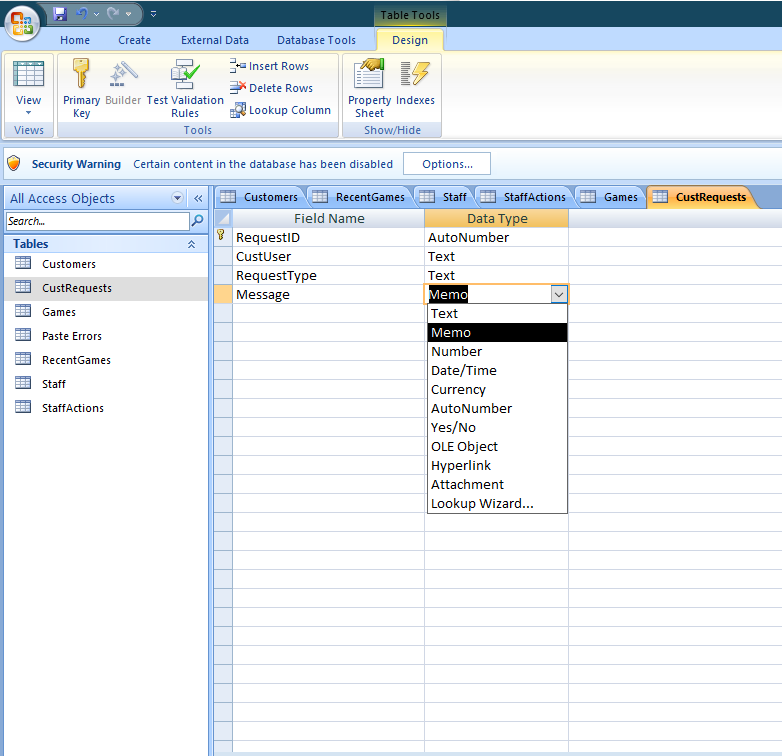
Currently, when a user writes an abnormally long request, the program produces an output wrongly saying that the request has been sent, the program then crashes.







To fix this, I had to change the data type of the field “Message” in the Microsoft Access database. This is the field that holds the requests that are sent by customers. I changed its data type from Text to Memo. I did this because as a string, the maximum length is 255 characters, and as a Memo, the maximum length is 65,536 which sufficiently covers any request that would be sent from a customer.



Testing and Development for change credentials form

This test shown below has been documented in my Test Plan Section of the project. I have identified that the program should have created a different output to the one that has occurred.

Currently, when a user tries to change their password to a value of 2 characters, no error is produced and their password is subsequently changed on the database.

To fix this, I had to add validation to the form to ensure that their new password/username needs to be at least 5 characters before it can be changed. This complies with the validation rules regarding usernames/passwords throughout the entire system.