Project | Programming 2

Term 2 2020

Table of Contents

[Introduction 1](#_Toc74834545)

[Explanation of the program 1](#_Toc74834546)

[Waterfall 1](#_Toc74834547)

[Figure 1 – Waterfall diagram 1](#_Toc74834548)

[Analysis 1](#_Toc74834549)

[What data needs to be inputted into the program 1](#_Toc74834550)

[What processes need to be performed while the program is running 1](#_Toc74834551)

[What output is generated when the program closes? 2](#_Toc74834552)

[Project Plan 2](#_Toc74834553)

[List the tasks that needed to be performed to create this project 2](#_Toc74834554)

[List the job title for each task, when and in what order (UI designer, programmer, etc) 2](#_Toc74834555)

[List the physical resources required to complete the project (PC, software, etc) 2](#_Toc74834556)

[Algorithm Design 3](#_Toc74834557)

[Drone ADD button 3](#_Toc74834558)

[Drone load binary file 4](#_Toc74834559)

[Drone save binary file 5](#_Toc74834560)

[Sort algorithm 6](#_Toc74834561)

[Test Data and Evidence 6](#_Toc74834562)

[Describe the test data and activities that will be used to test the design and program functionality 6](#_Toc74834563)

[Test table 7](#_Toc74834564)

[Run a series of tests using the test data previously stated. Record your tests and collect screenshots of the program functionality 8](#_Toc74834565)

[Figure 1 8](#_Toc74834566)

[Figure 2 8](#_Toc74834567)

[Figure 3 9](#_Toc74834568)

[Figure 4 9](#_Toc74834569)

[Figure 5 10](#_Toc74834570)

[Figure 6 10](#_Toc74834571)

[Figure 7 11](#_Toc74834572)

[Figure 8 11](#_Toc74834573)

[Figure 9 12](#_Toc74834574)

[Figure 10 12](#_Toc74834575)

[Figure 11 13](#_Toc74834576)

[Figure 12 13](#_Toc74834577)

[Figure 13 14](#_Toc74834578)

[Figure 14 14](#_Toc74834579)

[Figure 15 15](#_Toc74834580)

[Recommendations and Improvements 15](#_Toc74834581)

[Describe all concerns and improvements that can be added to you program 15](#_Toc74834582)

[Explain the limitations, error trapping, data filters and usability/portability of your application 15](#_Toc74834583)

# Introduction

## Explanation of the program

The program is designed to store and display data in relation to Ace Drones customers, transactions and drones. The user should be able to individually add customers or drones but using both of those are required to create a transaction.

The user must be able to search the customer array with a right click function and be able to clear the input boxes of each object when either of the top input boxes are double clicked. Data should be saved when closing the program and loaded plus displayed when opening the program.

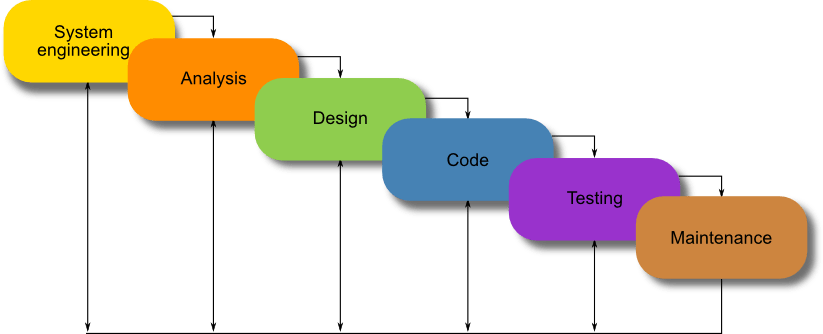
## Waterfall

The waterfall model was used to develop this application, it is a linear model that separates each stage of the SDLC process to be completed and then used as input for the following stage.

For example, the requirements gathering and analysis stage were fully explored through the documentation of the assessment and as the developer having built similar applications it made sense to build and test each block of code at a time without needing to make adjustments once complete.

Stages do not overlap in this model meaning development was not complicated. Prototyping or the spiral SDLC models would have also suited developing the application however there is no client to have ongoing discussions.

### Figure 1 – Waterfall diagram



# Analysis

## What data needs to be inputted into the program

The program takes data as strings which are assigned to the values of the matching objects.

## What processes need to be performed while the program is running

Whilst the program is running a number of processes take place;

* The three arrays are loaded and displayed
* Each of the arrays can be increased up to the max size of 20 with the drones array being bubble sorted following each successful addition.
* Arrays are checked for data being present in all the correct input boxes and the type prior to allowing a new addition, error messages are displayed if required. The customer array does have the option however if some data is missing the user can create a default customer.
* Double clicking either the serial number, customer ID or transaction ID boxes clears the matching object input boxes.
* A single click on an item within either the drone or customer list boxes should populate the matching object input boxes and the matching transaction input boxes. A single click within the transaction list box should populate the transaction input boxes as well as the matching customer and drone input boxes.
* Entering a customer ID to the input box then right clicking and selecting search should either fill the matching customer input boxes or display an error message to the user.
* When the user hovers over major functions tool tips appear with hints.

## What output is generated when the program closes?

When the program is closed a save process occurs on three separate binary files each representing one of the three arrays.

# Project Plan

## List the tasks that needed to be performed to create this project

* Consult with Ace Drones to gather requirements
* Design the user interface
* Code both the events and methods required
* Test each event or method as they are built
* Write documentation
* Deployment
* Maintenance

## List the job title for each task, when and in what order (UI designer, programmer, etc)

* Senior programmer, consult with Ace Drones, contribute to coding, testing and handle deployment
* UI designer, consult with Ace Drones, design user interface, contribute to coding and testing
* Programmer, contribute to coding, testing and maintenance
* Document writer, consult with Ace Drones, write documentation and assist deployment

## List the physical resources required to complete the project (PC, software, etc)

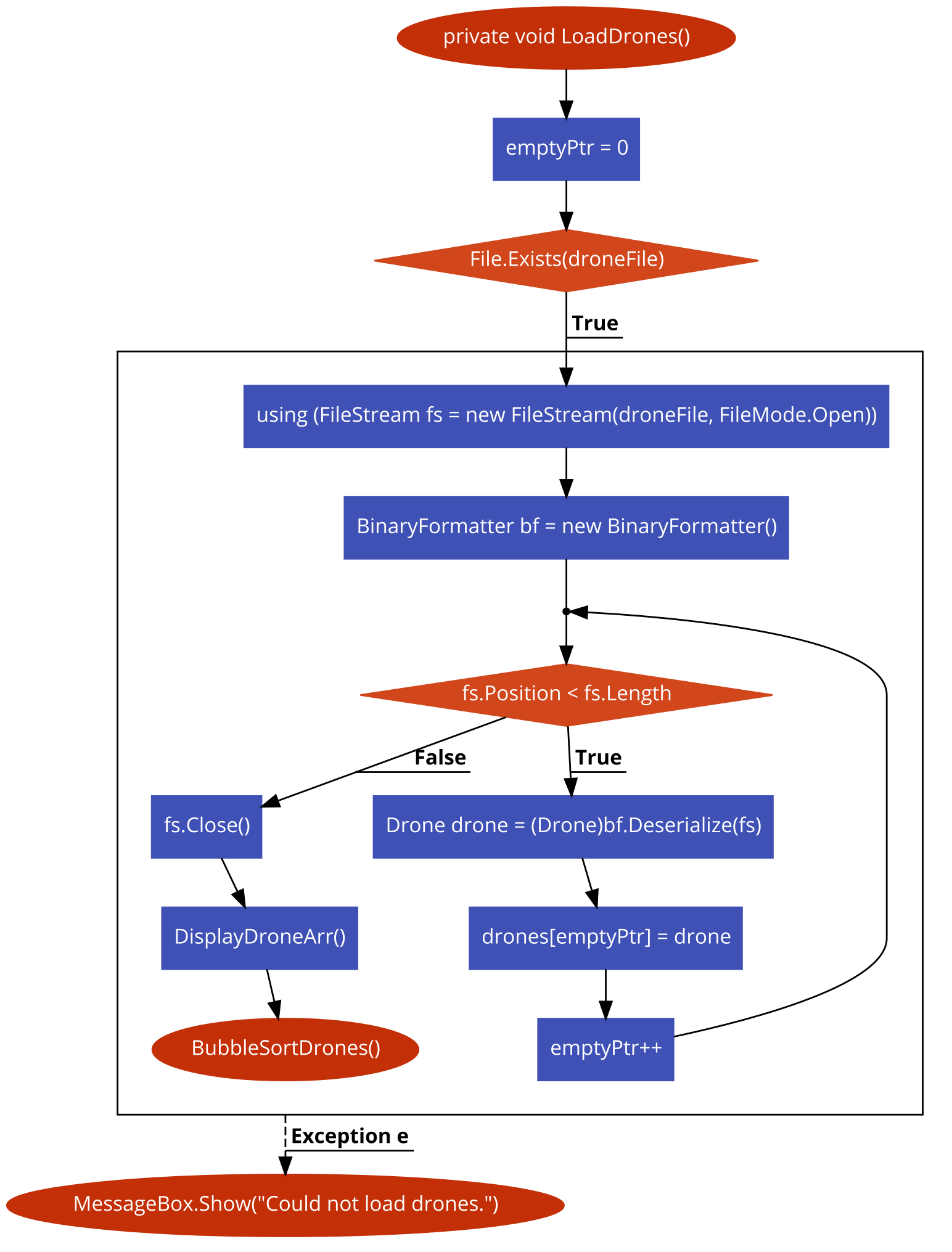
* PC
* Visual Studio (IDE software)
* Office space for four people
* Listed Requirements

# Algorithm Design

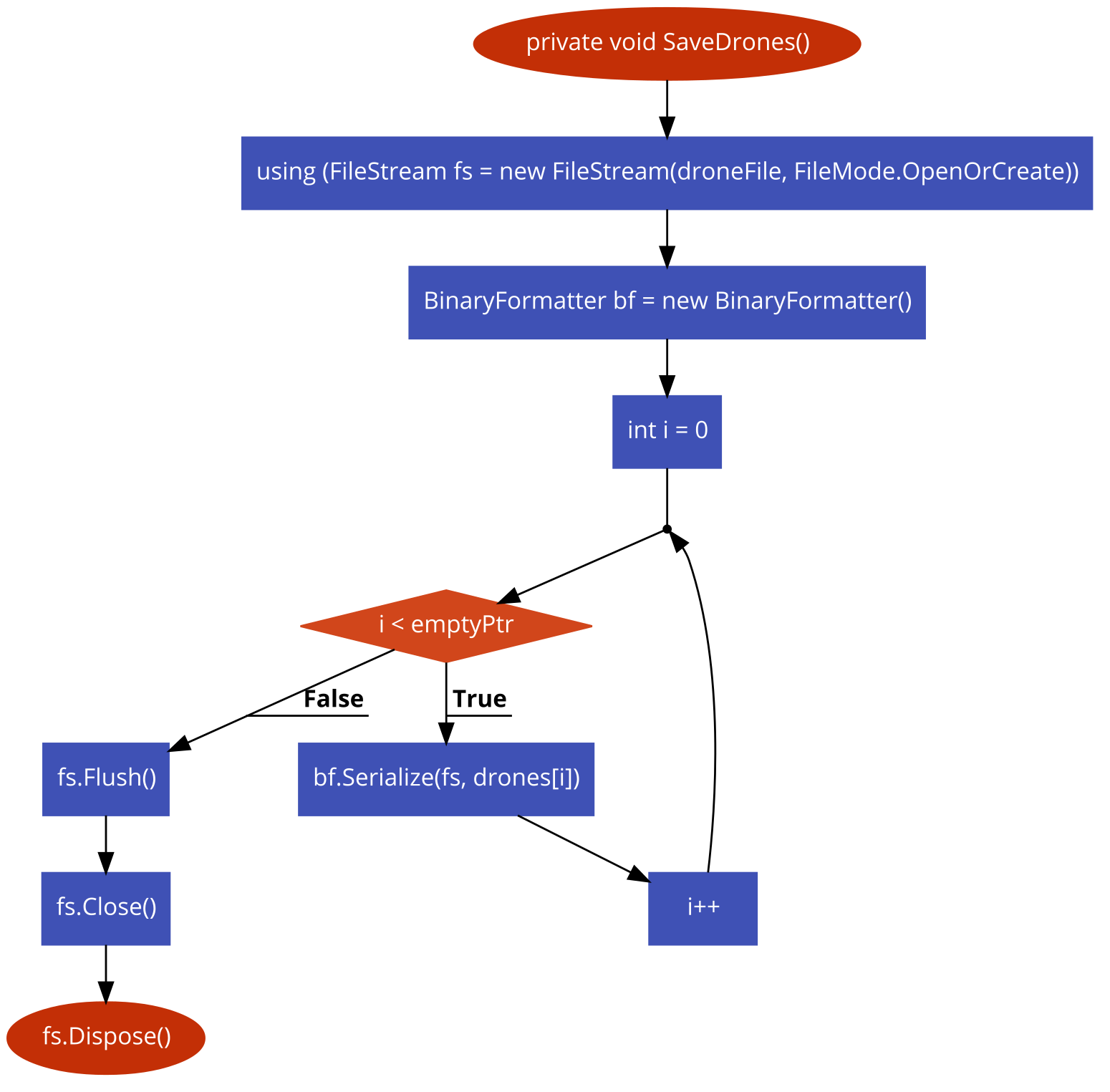
## Drone ADD button



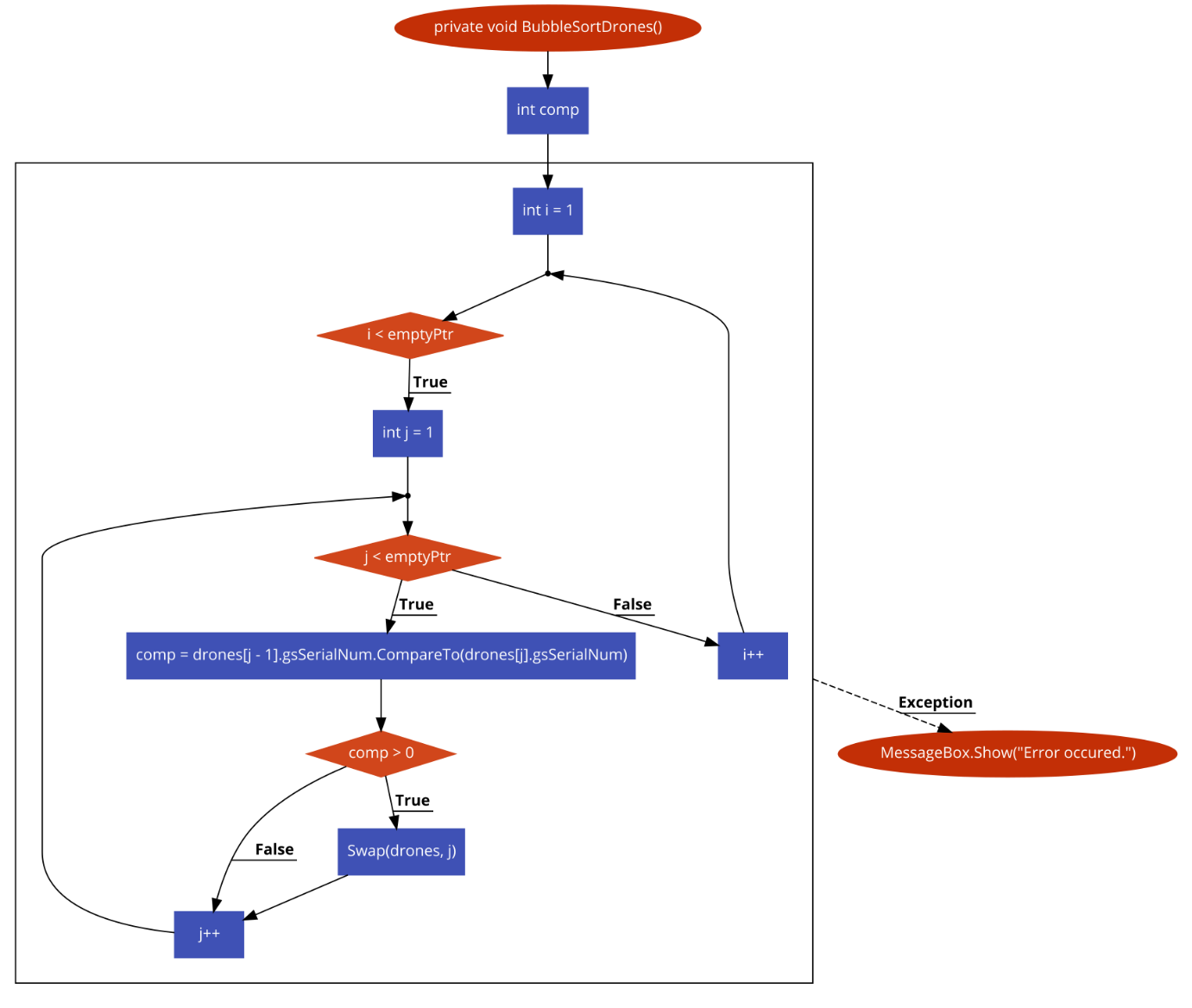
## Drone load binary file



## Drone save binary file



## Sort algorithm



# Test Data and Evidence

## Describe the test data and activities that will be used to test the design and program functionality

* Incorrect data type used to try create an object
* Attempting to create an object with missing input fields
* Searching for a customer ID that will be found
* Searching for customer ID that won’t be found
* Hovering over a major function to check for a tool tip
* Double clicking an appropriate input field to check all appropriate input fields are cleared
* Double clicking an inappropriate input field to check all appropriate input fields are not cleared
* Single clicking an item from the customer list box to check the appropriate input boxes are populated
* Single clicking an item from the transactions list box to check the appropriate input boxes are populated including the matching drone and customer input boxes

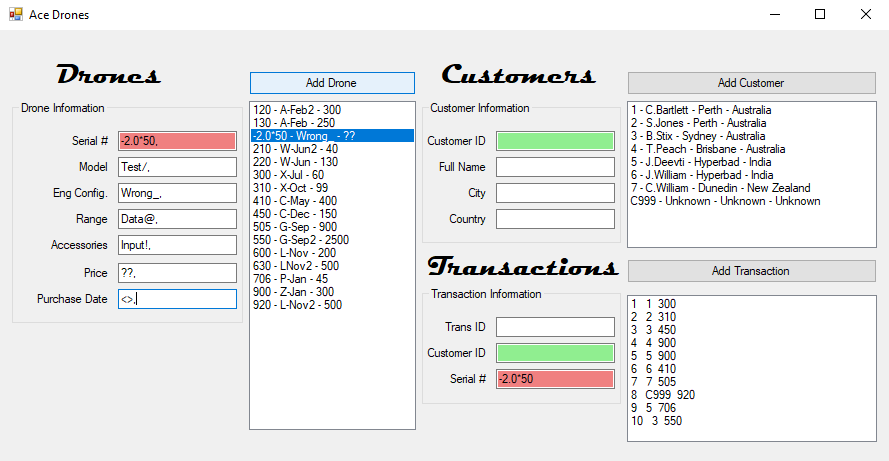
## Test table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Expected Result | Actual Result | Comments | Verdict |
| Incorrect data type used to try create an object  Attempting to create an object with missing input fields | All fields take in string data type and no error will occur. | No error occurred. | See figures 1 – 2 | Pass |
| Attempting to create an object with missing input fields | Error message displayed when trying to create drone. | Error message displayed. | See figure 3 – 4 | Pass |
| Searching for a customer ID that will be found | Remaining customer input boxes populated with correct data. | Remaining customer input boxes populated with correct data. | See figures 5 – 6 | Pass |
| Searching for customer ID that won’t be found | Error message displayed. | Error message displayed. | See figures 7 – 8 | Pass |
| Hovering over a major function to check for a tool tip | Tool tip displayed. | Tool tip displayed. | See figure 9 | Pass |
| Double clicking an appropriate input field to check all appropriate input fields are cleared | Appropriate input boxes are cleared. | Appropriate input boxes are cleared. | See figures 10 – 11 | Pass |
| Double clicking an inappropriate input field to check all appropriate input fields are not cleared | The text within the input box will be highlighted. | The text within the input box was be highlighted. | See figures 12 – 13 | Pass |
| Single clicking an item from the customer list box to check the appropriate input boxes are populated | Appropriate input boxes are populated. | Appropriate input boxes are populated. | See figure 14 | Pass |
| Single clicking an item from the transactions list box to check the appropriate input boxes are populated including the matching drone and customer input boxes | Appropriate input boxes are populated including the matching details for the drone and customer input boxes. | Appropriate input boxes are populated including the matching details for the drone and customer input boxes. | See figure 15 | Pass |

## Run a series of tests using the test data previously stated. Record your tests and collect screenshots of the program functionality

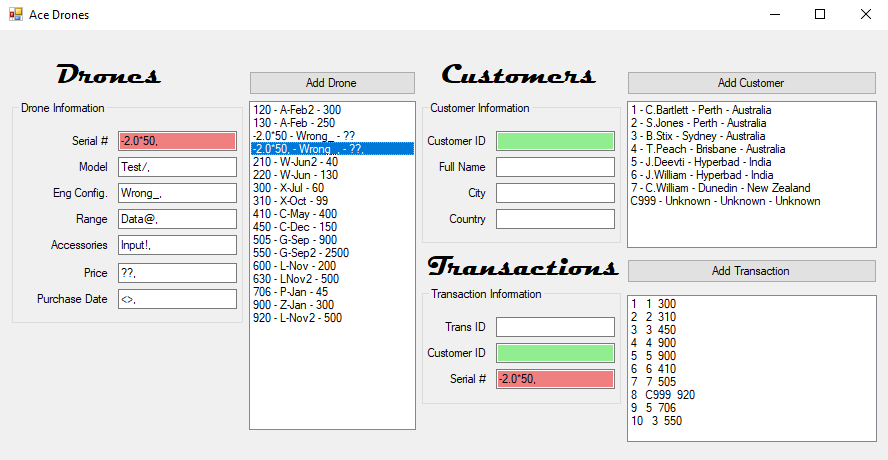
### Figure 1

User attempts to enter random data to create a drone, the program accepts strings as inputs as there was no requirements for specific data types meaning no error should or will occur.

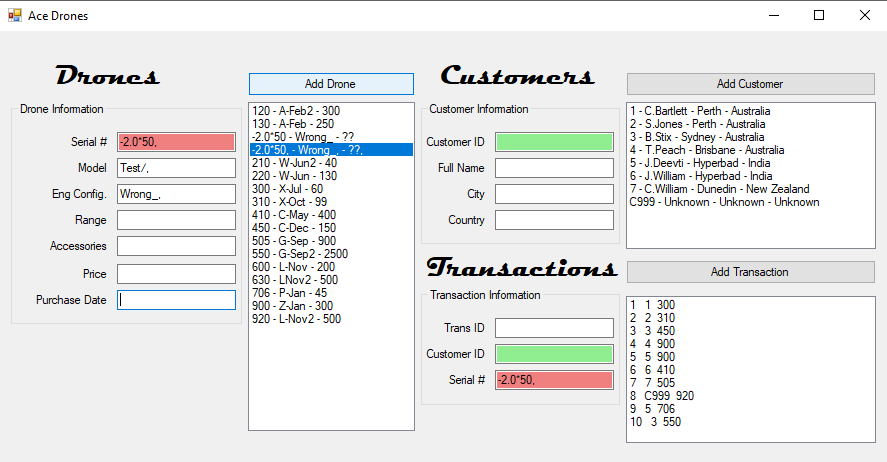


### Figure 2

As expected, the drone has been added to the list box.

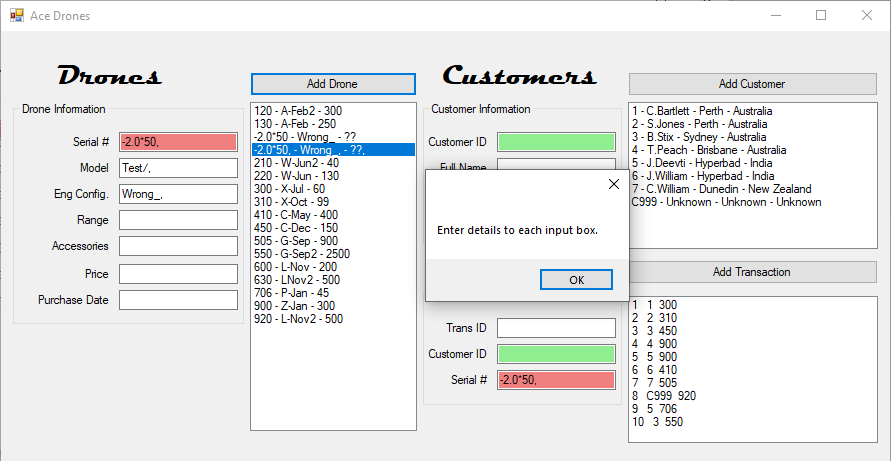


### Figure 3

The user attempts to add a new drone with data missing from some of the input boxes, an error message is expected to occur.

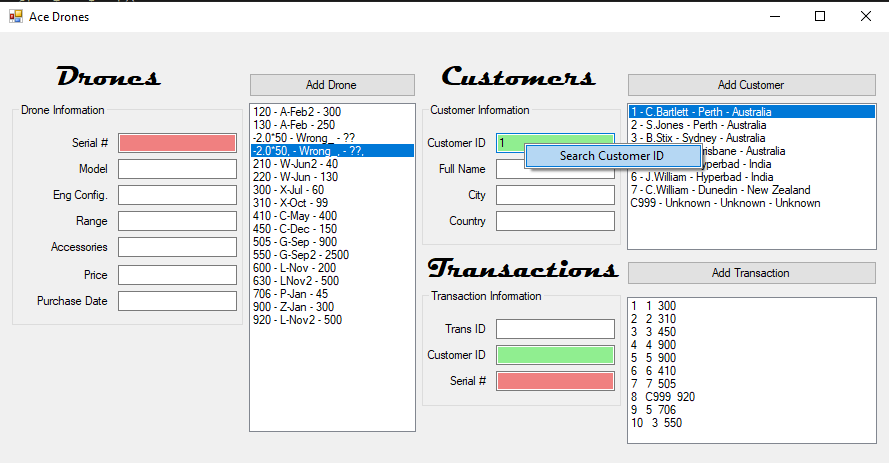
### Figure 4

As expected, the program has displayed an error message to the user advising what to do.



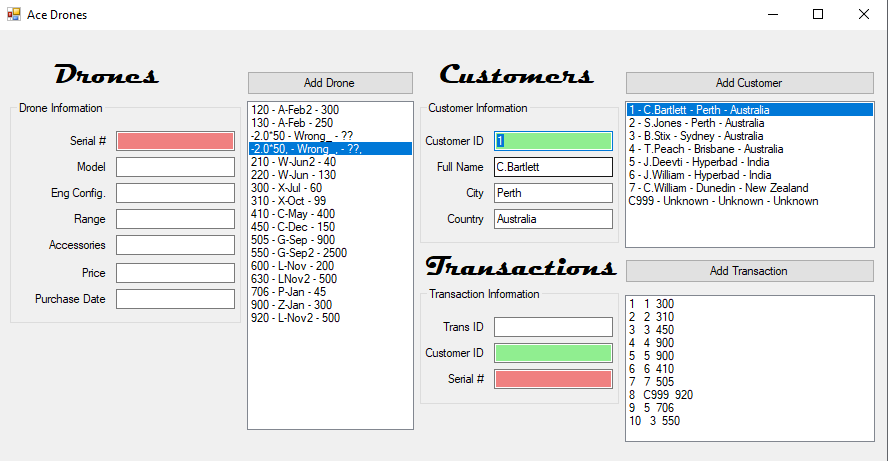
### Figure 5

The user attempts to search for a customer ID that will be found.



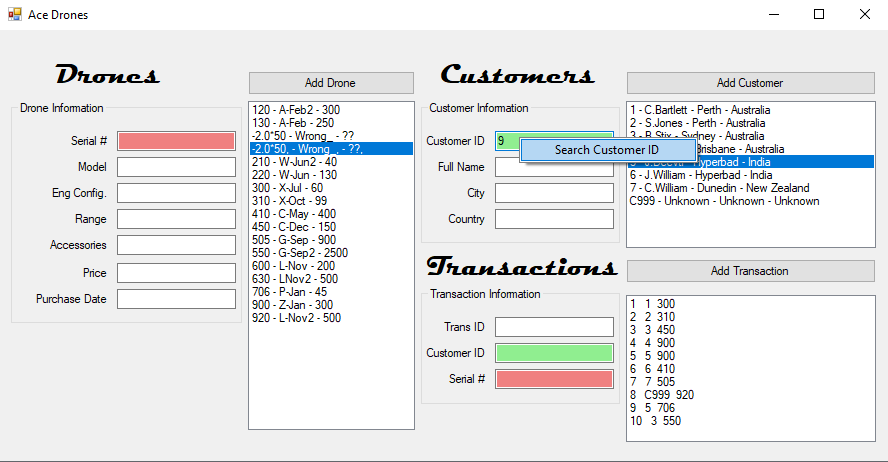
### Figure 6

As expected, the data matching customer ID 1 populates the remaining input boxes related to that customer ID.



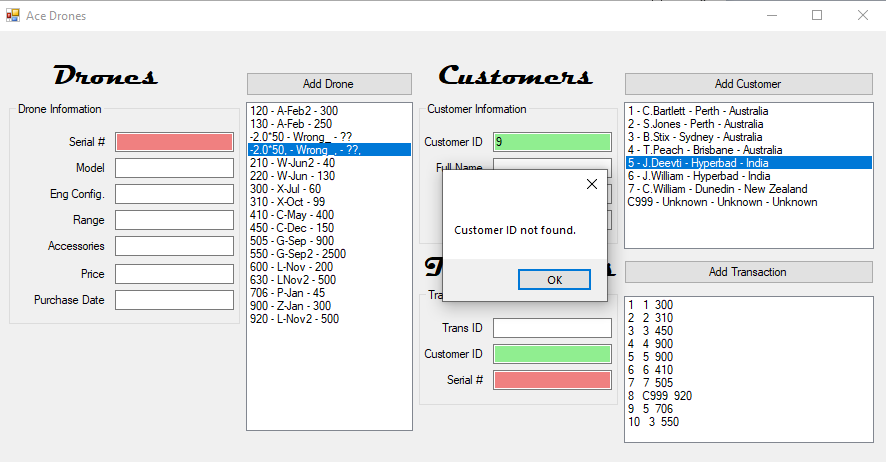
### Figure 7

The user attempts to search for a customer ID that won’t be found.



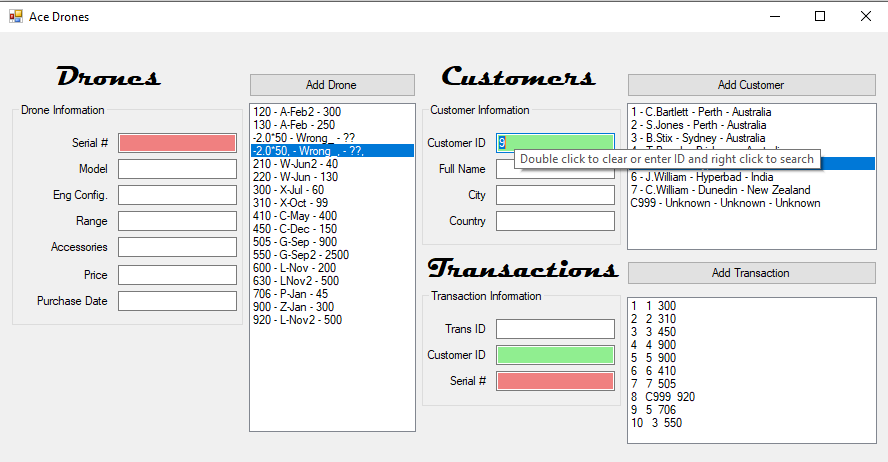
### Figure 8

As expected, an error message is displayed to the user given the customer ID that was searched isn’t in the array.



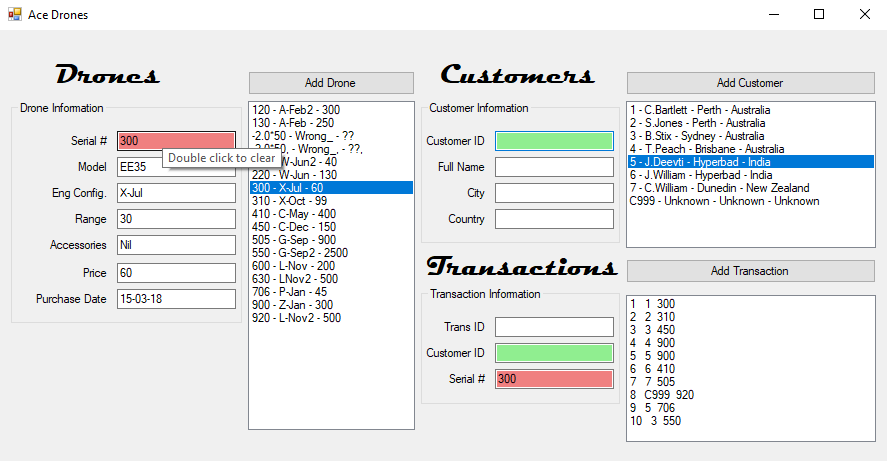
### Figure 9

The user hovers the mouse pointer over a major function and as expected a tool tip with a hint is displayed.



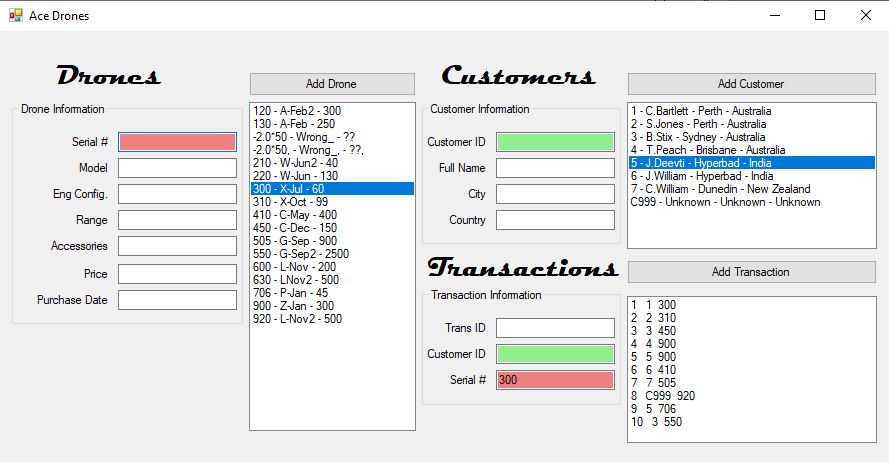
### Figure 10

The user double clicks the serial number input box under drones following a single click of an item within the drone list box.



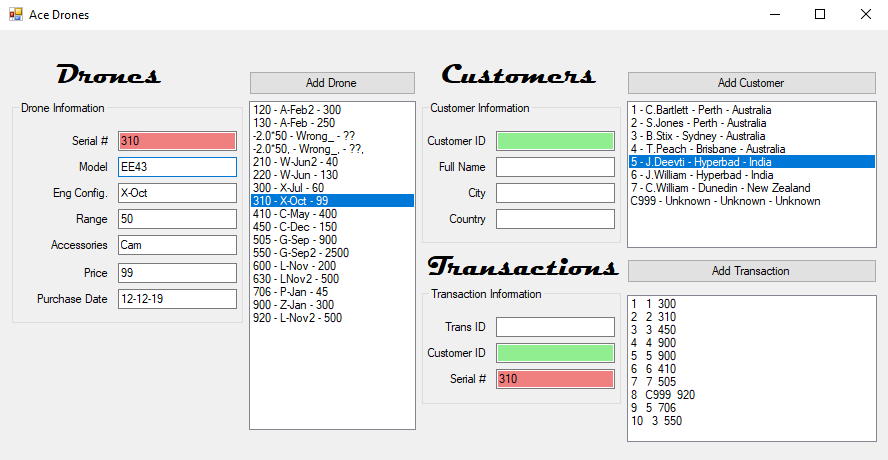
### Figure 11

As expected, following the user making the double click each of the drone input boxes has been cleared.



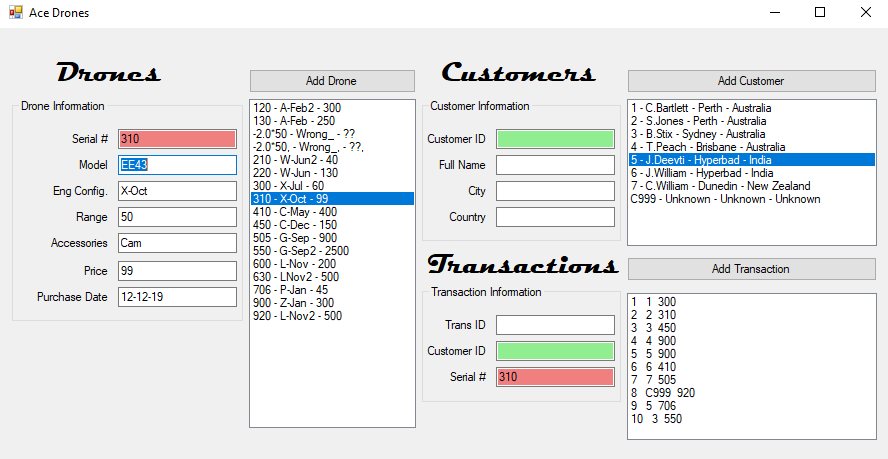
### Figure 12

The user double clicks the model input box under drones following a single click of a drone item within the drone list box.



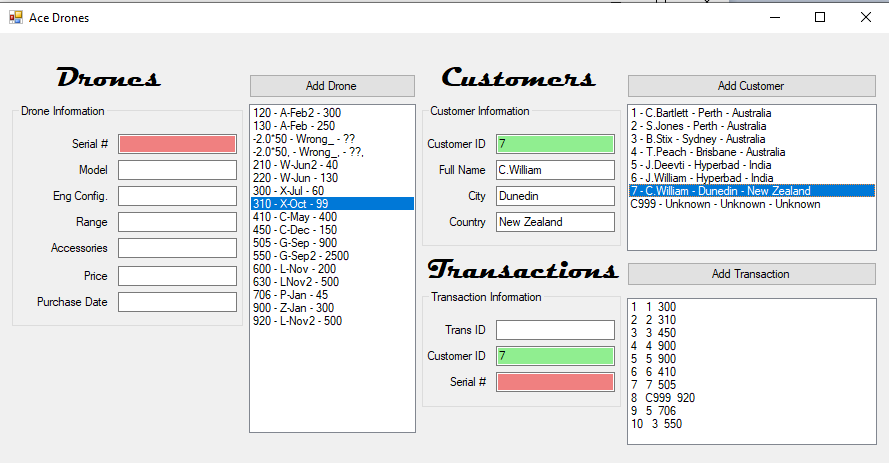
### Figure 13

As expected, the text within the model box under drones is highlighted and not cleared. The clear function is specifically meant for the serial number, customer ID and transaction ID boxes.



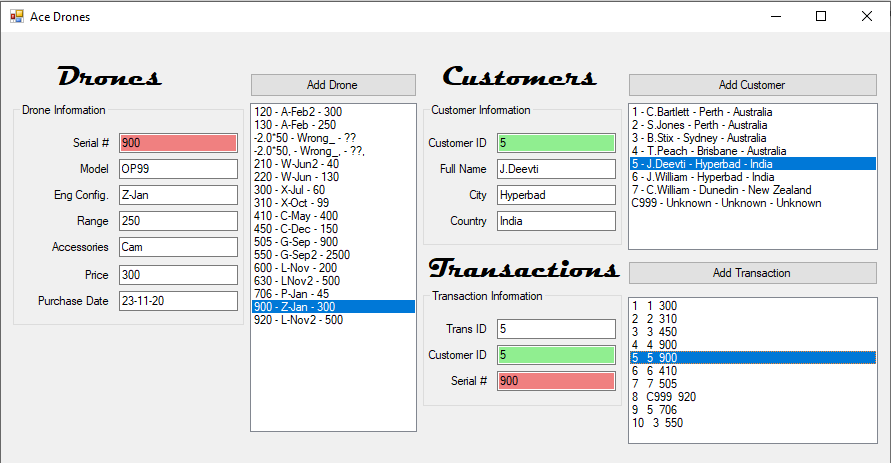
### Figure 14

The user makes a single click on an item in the customer list box and the appropriate customer input boxes are populated.



### Figure 15

The user makes a single click on an item in the transaction list box and the appropriate transaction input boxes are populated. The appropriate drone and customer input boxes are also populated.



# Recommendations and Improvements

### Describe all concerns and improvements that can be added to you program

The program could make use of a database rather than binary files and further error catching could have been used making the program more secure and unlikely to crash.

## Explain the limitations, error trapping, data filters and usability/portability of your application

Using try catch statement assist in preventing runtime crashes and currently the program can’t be deployed for multiple devices.