

## **Milestone 4**

Ian M. McConihay

College of Science, Engineering and Technology, Grand Canyon University

CST-150: C# Programming I

Mark Smithers

September 15, 2024

**Video Link:**

<https://www.loom.com/share/b30b36a5f32243f08c60e727f3965a94?sid=d0cb39cc-28f9-49ff-aff6-132b54fbbf61>

**Github:** <https://github.com/Ian-McConihay/CST-150>

What was challenging?

My Code for the most part was already outside of the main method so there were not to many challenges.

What did you learn?

Binding textbox to method inputs.

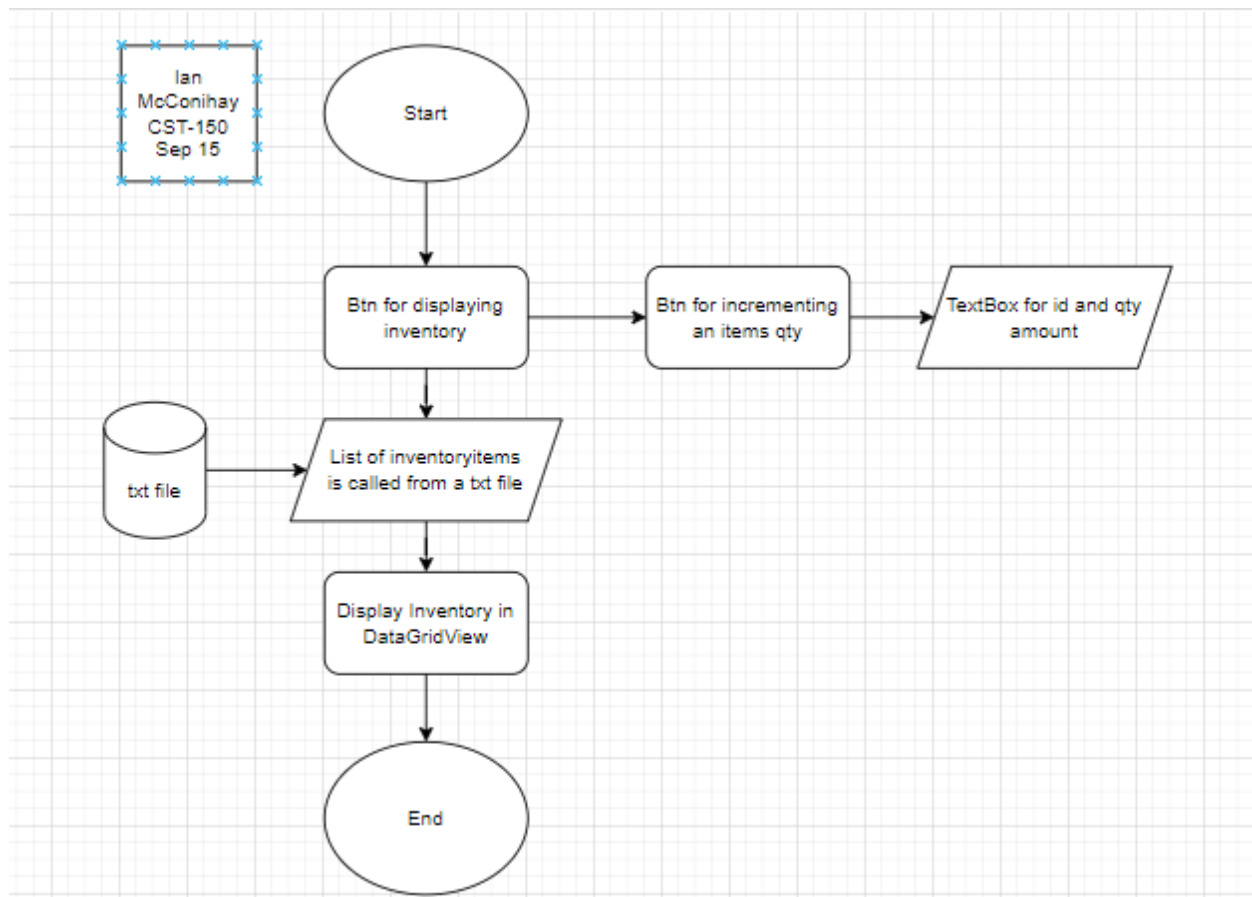
How would you improve on the project?

I would create a system to update all of the item's data.

How can you use what you learned on the job?

Clean code makes it easier for other developers to understand what's going on.

Figure 1: FlowChart



At the start of this application will open to a button for the user to click and persist a grid view of inventory items from a text file. Another button will allow the user to increment one of the items quantity. The user will be able to enter the id and the amount they want to increment an item. The datagridview provides table functions for the user to view the inventory.

Figure 2: UML InventoryItem

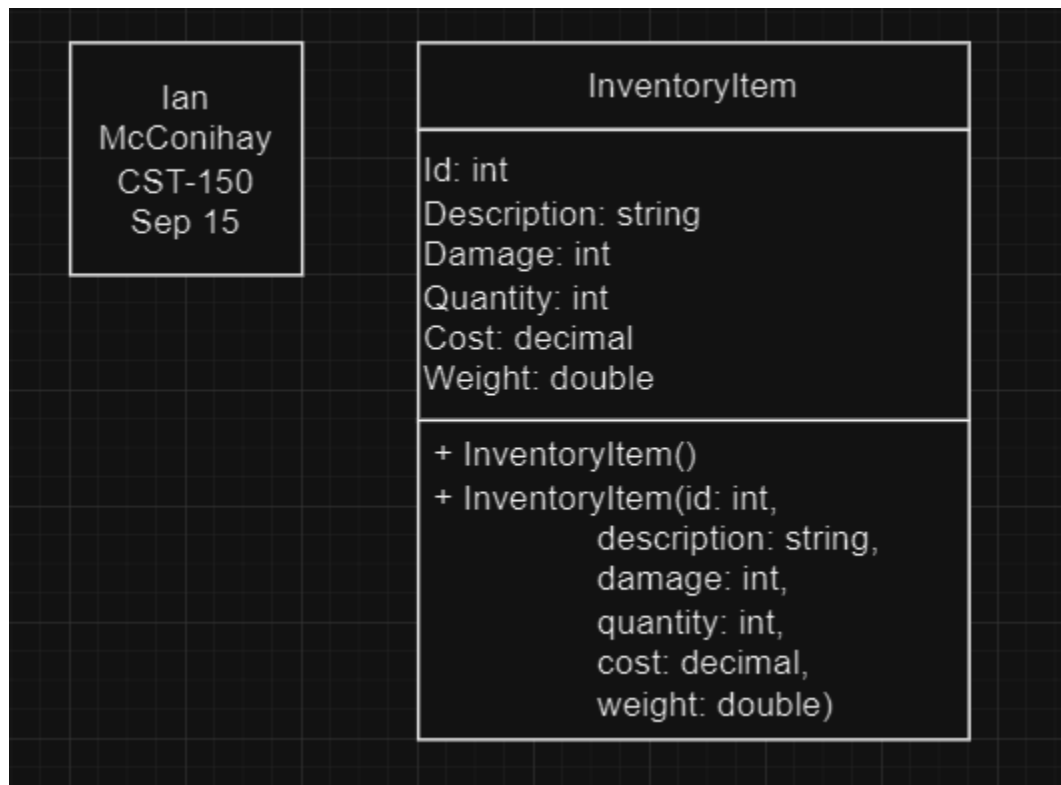
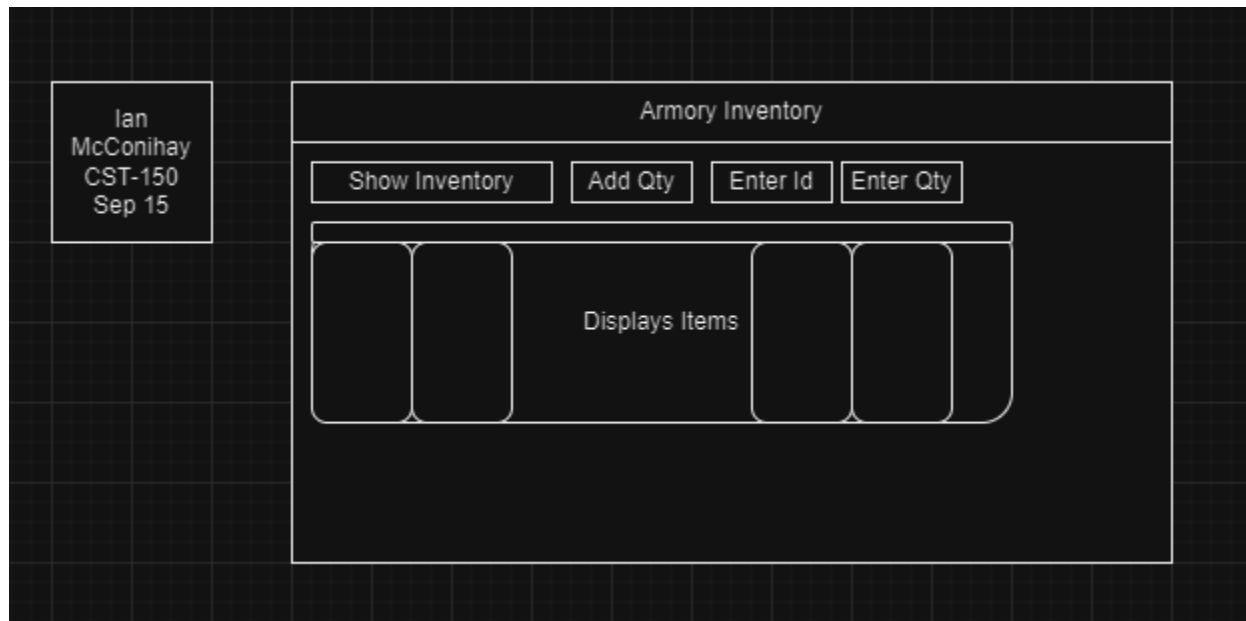


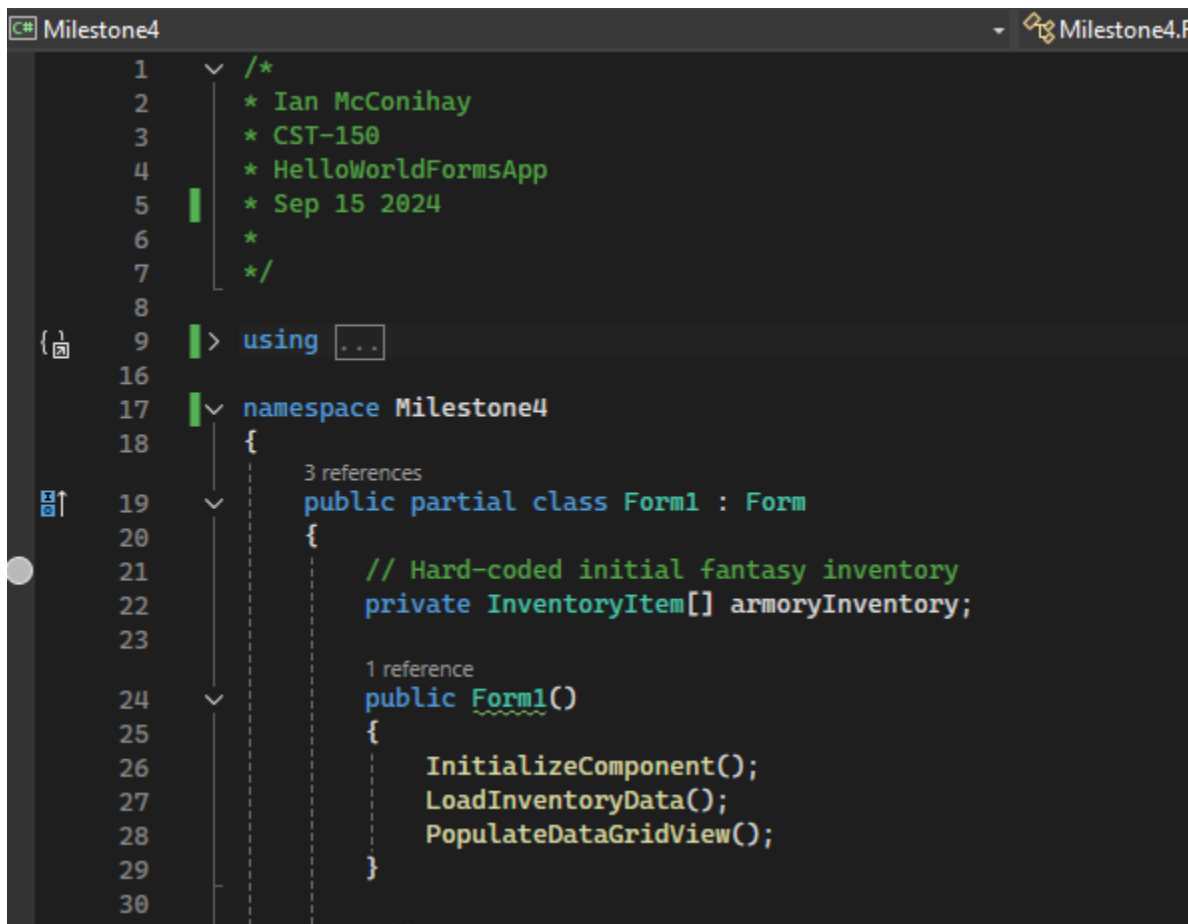
Figure 3: Wireframe



The updated wire frame has a handful of changes. The form has a name when displayed for the user. The button shows inventory is located above where the list will be displayed using a data grid view. The user will have columns to display the information. Add Qty button for incrementing an items quantity now has two textboxes for the user to enter the id and qty amount.

## Application Screenshots

Figure 4: Code



```
1  /*
2   * Ian McConihay
3   * CST-150
4   * HelloWorldFormsApp
5   * Sep 15 2024
6   *
7   */
8
9  > using ...
10
11
12
13
14
15
16
17  namespace Milestone4
18  {
19      3 references
20      public partial class Form1 : Form
21      {
22          // Hard-coded initial fantasy inventory
23          private InventoryItem[] armoryInventory;
24
25          1 reference
26          public Form1()
27          {
28              InitializeComponent();
29              LoadInventoryData();
30              PopulateDataGridView();
31          }
32      }
33  }
```

For figure 4 we start off with the citation at the top. As we can see there is no business logic located in the main method/ Form1() method. This seemed to be the main point for milestone 4.

Figure 5: Code

```

46 private void btnIncrement_Click(object sender, EventArgs e)
47 {
48     IncrementInventory(int.Parse(textBoxIncQty.Text), int.Parse(textBoxQty.Text));
49 }
50
51 1 reference
52 private void LoadInventoryData()
53 {
54     string filePath = "C:\\Users\\nmcco\\Desktop\\CST-150\\Milestone3\\Milestone3\\bin\\Debug\\net8.0-windows\\Data\\Inventory.txt";
55
56     try
57     {
58         // Read all lines from the file
59         string[] lines = File.ReadAllLines(filePath, Encoding.UTF8);
60         if (lines.Length == 0)
61         {
62             MessageBox.Show("The inventory file is empty.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);
63             return;
64         }
65         // Initialize the array with the correct size
66         armoryInventory = new InventoryItem[lines.Length];
67
68         for (int i = 0; i < lines.Length; i++)
69         {
70             string line = lines[i];
71             string[] parts = line.Split(',');
72
73             if (parts.Length == 6)
74             {
75                 int id = int.Parse(parts[0]);
76                 string description = parts[1];
77                 int damage = int.Parse(parts[2]);
78                 int quantity = int.Parse(parts[3]);
79                 decimal cost = decimal.Parse(parts[4]);
80                 double weight = double.Parse(parts[5]);
81
82                 armoryInventory[i] = new InventoryItem(id, description, damage, quantity, cost, weight);
83             }
84         }
85     } catch (Exception ex)

```

Figure 5 has the LoadInventoryData method that calls to the filePath leading to the txt file. Using a try catch we read all the lines in the file and break them up using a for loop. The btnIncrement\_Click now has been adjusted to take arguments from textBoxes instead of a set value.

Figure 6: Code

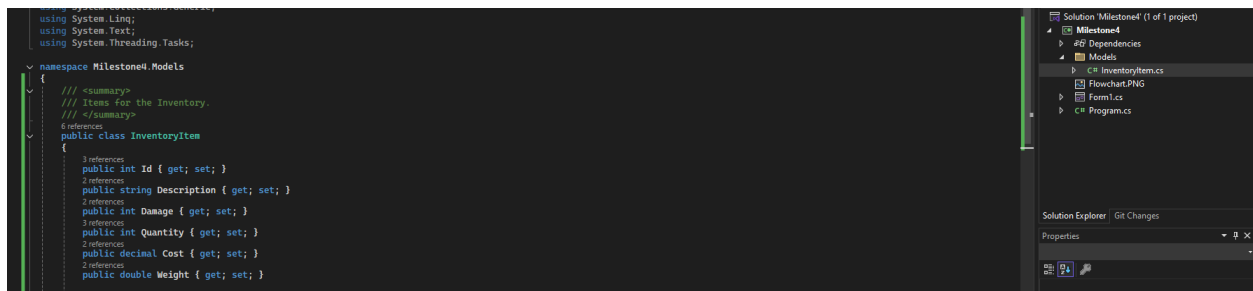


Figure 6 shows that I have created a Models folder. In the model folder I now will store my `InventoryItem` and any other class objects to be added.

Figure 7: Code

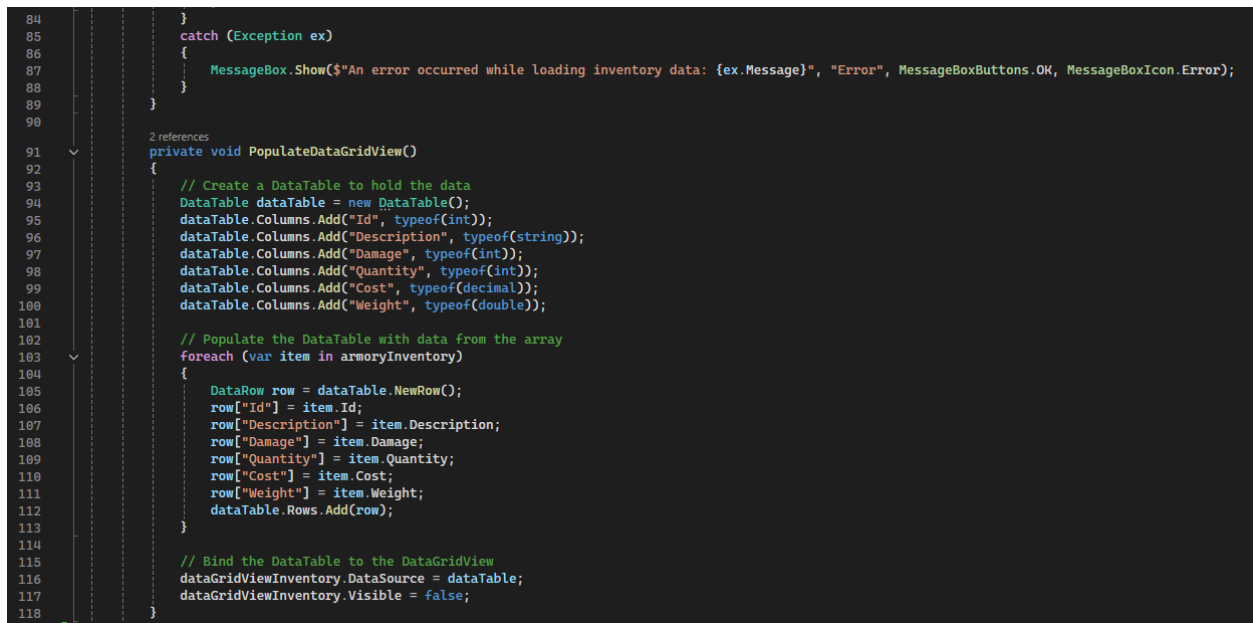
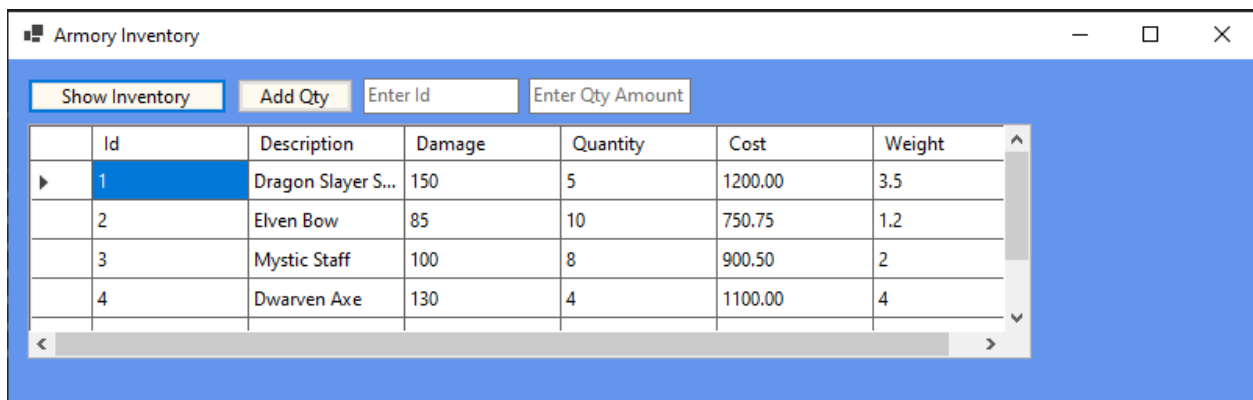


Figure 7 is to show more inventory inside of a separate method from the main. The logic is unchanged for populating the view.



Figure 8: Application Start



The start of the application displays the show inventory button and add qty button. From here I can enter in an Id and qty amount. I freshened up the display using the color scheme tool.

Figure 9: Application Add Inventory

The screenshot shows a window titled "Armory Inventory" with a blue background. At the top, there are two buttons: "Show Inventory" and "Add Qty". To the right of the "Add Qty" button are two input fields containing the numbers "1" and "2". Below these controls is a table with the following data:

	Id	Description	Damage	Quantity	Cost	Weight
▶	1	Dragon Slayer S...	150	7	1200.00	3.5
	2	Elven Bow	85	10	750.75	1.2
	3	Mystic Staff	100	8	900.50	2
	4	Dwarven Axe	130	4	1100.00	4

Entering a 1 in and 2 the item with an Id of 1 was incremented positively by 2. This give the user a lot more freedom to make adjustments to the items in stock. Adjusting the rest of the items would be the next step.

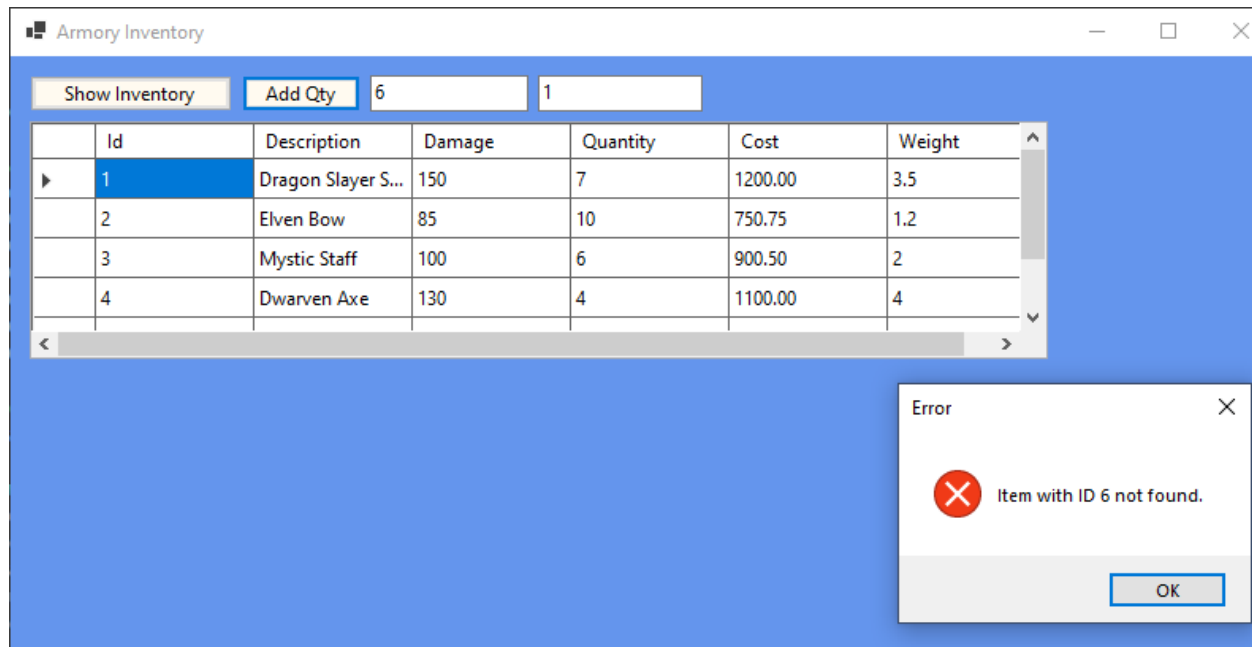
Figure 10: Application subtract Inventory item

The screenshot shows the same "Armory Inventory" window. The "Add Qty" button is still visible, but the input fields now contain "3" and "-2". The table below shows the updated quantities:

	Id	Description	Damage	Quantity	Cost	Weight
▶	1	Dragon Slayer S...	150	7	1200.00	3.5
	2	Elven Bow	85	10	750.75	1.2
	3	Mystic Staff	100	6	900.50	2
	4	Dwarven Axe	130	4	1100.00	4

Figure 10 is post pushing the Add Qty button with a negative number. As we can see the Quantity entered was -2 so the stock item went from 8 to 6 for the id 3 item. This is not a good practice but it does add loads of functionality.

Figure 11: Error handling.



Here we have tried to update an id not listed. A message box appears for the user to be notified the item is not found. Next the user can click okay and try an id that is available.

## Bug Reports

Bug Report

Class name

Method name :

Steps to reproduce the bug:

Expected results

Actual results

details: N/A for milestone 1

Solution

1. List your computer specs (type of computer, OS, memory, etc)

Device name DESKTOP-IAQ5CCD

Processor Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM 8.00 GB (7.88 GB usable)

Device ID A0AC8D02-4885-4491-B27B-B40F0A0D2E35

Product ID 00356-02139-31547-AAOEM

System type 64-bit operating system, x64-based processor

Pen and touch Touch support with 10 touch points

2. Create 3 test cases

Valid File with Proper Data: The method should correctly populate the `armoryInventory` array with `InventoryItem` objects based on properly formatted data in the file.

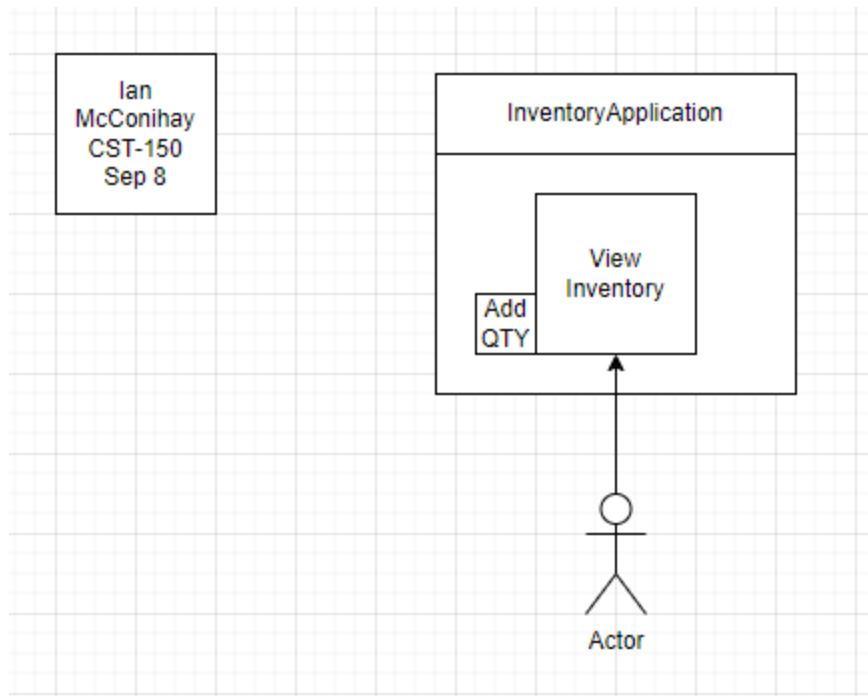
File is Empty: The method should display a warning message indicating the file is empty and leave the `armoryInventory` array uninitialized.

File with Incorrect Data Format: The method should show an error message indicating an issue with loading data and only initialize valid `InventoryItem` objects in the `armoryInventory` array.

3. List 3 Programming conventions that will be used all milestones

Naming, Format, and Documentation Conventions

4. Create Use case diagram



System Boundary: Representing the WinForms application.

Use Case: "View Inventory" indicating the functionality provided by the application.

Actor: "User" who interacts with the system to view the inventory.

Monday

Start: 900pm End: 9:30pm Activity: Read announcements

Start: 930pm End: 1030 Activity: DQ1 and DQ 2

Start: 1030pm End: 1100pm Activity: Read Book

Tuesday

Start: 900pm End: 9:30pm Activity: Participation post

Start: 930pm End: 1030 Activity: Activity 4

Start: 1030pm End: 1100pm Activity: Read Book

Wednesday

Start: End: Activity: N/A

Start: End: Activity: N/A

Start: End: Activity: N/A

Thursday

Start: 900pm End: 9:30pm Activity: Participation post

Start: 930pm End: 1030 Activity: Activity 4

Start: 1030pm End: 1100pm Activity: Read Book

Friday

Start: 900pm End: 9:30pm Activity: Participation post

Start: 930pm End: 1030 Activity: Milestone

Start: 1030pm End: 1100pm Activity: Read Book

Saturday

Start: 900pm End: 9:30pm Activity: Milestone

Start: 930pm End: 1030 Activity: Milestone

Start: 1030pm End: 1100pm Activity: Read Book

Sunday

Start: 900pm End: 9:30pm Activity: Activity 4

Start: 930pm End: 1030 Activity: Milestone

Start: 1030pm End: 1100pm Activity: Read Book