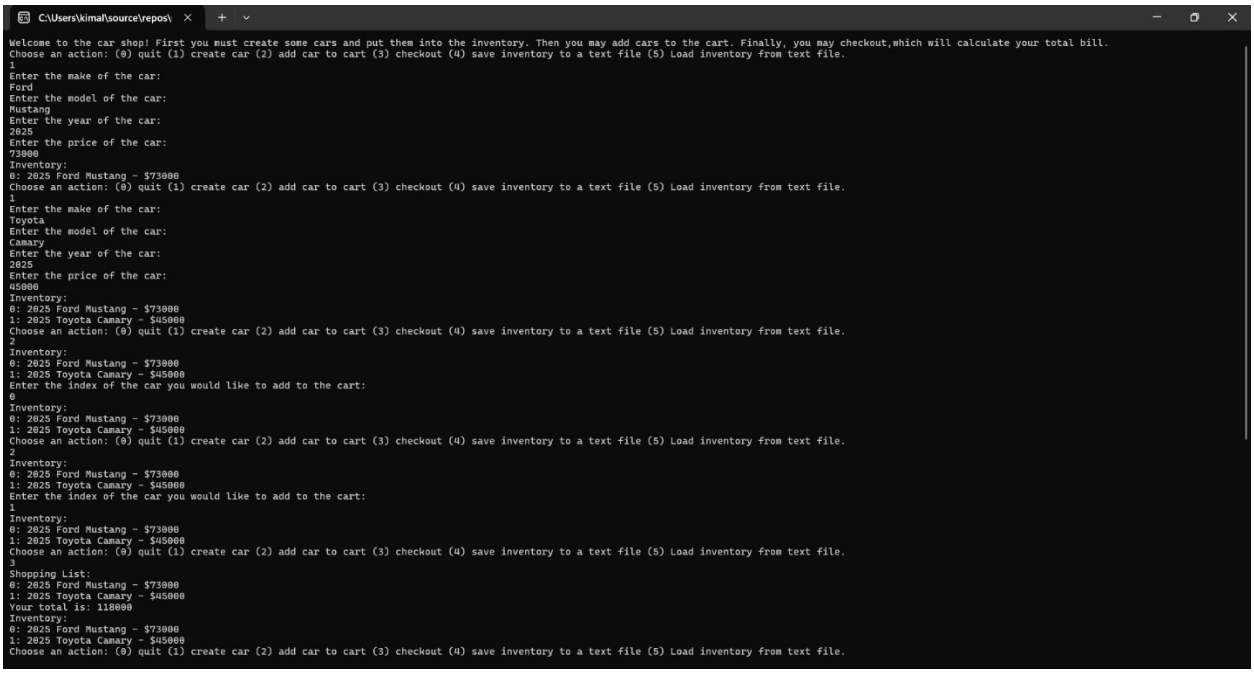
CST-250 Activity 1

Kimberly Alvarez

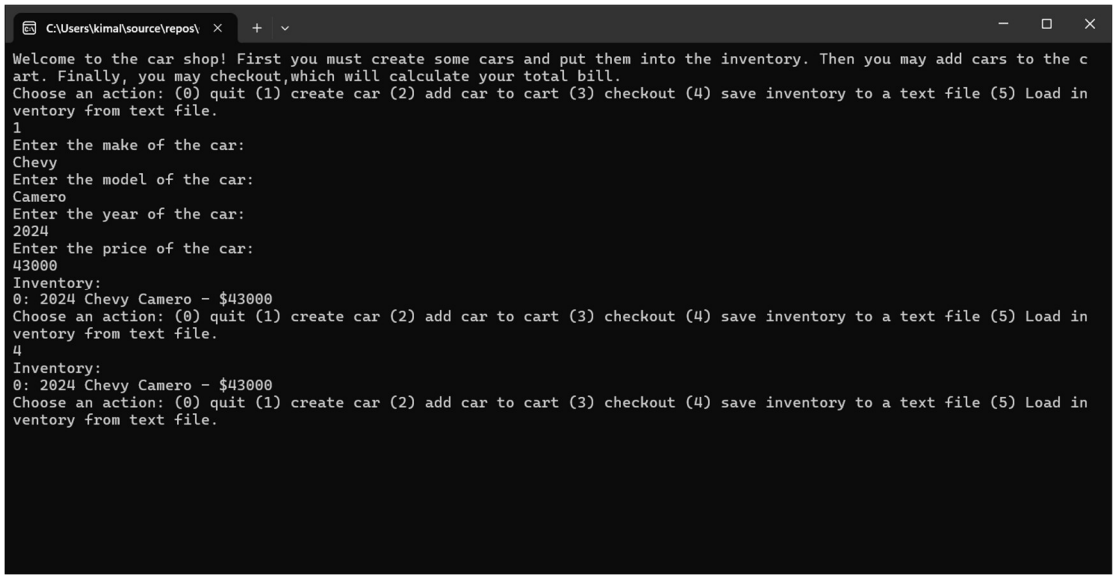
3/19/2025

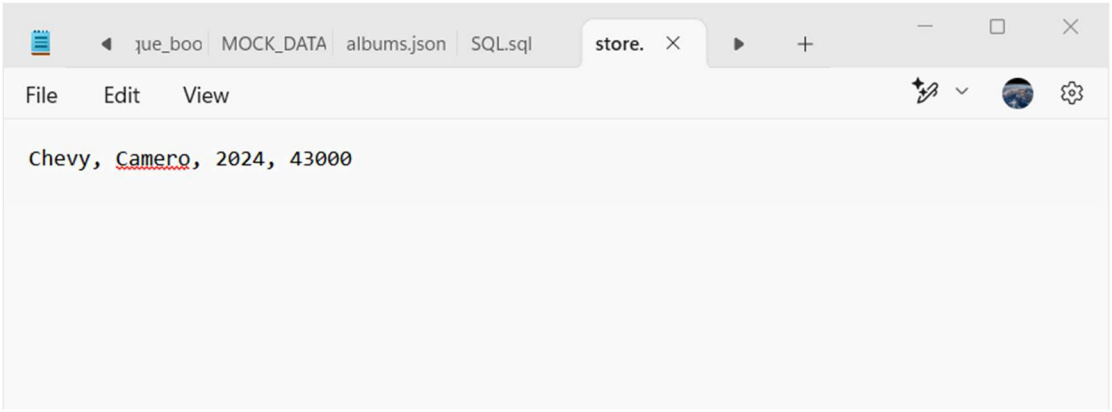
Part 2 – Console Application



In the image above we have creates a console application. The application begins by prompting the user to decide what action they wish to being in the program. In this image we can see that we began by choosing action 1 which is the creation of a car. After building the car the inventory prints out the car information in the system so far. In the example set above we created two cars. The next thing that we did was add these cars to our cart by typing action number 2 and adding each car by their index number provided in the inventory list. Then we are able to see what the total cost for both vehicles would be. While completing this exercise I did have some trouble as I kept getting CarLibrary.Car in my inventory rather than the cars' information to the cars I was building. I had to do some research on how to fix this issue and I used Microsoft Copilot to help me determine what could be causing the issue. It turned out that I needed to add an override string in my Car class to ensure that my data was readable.

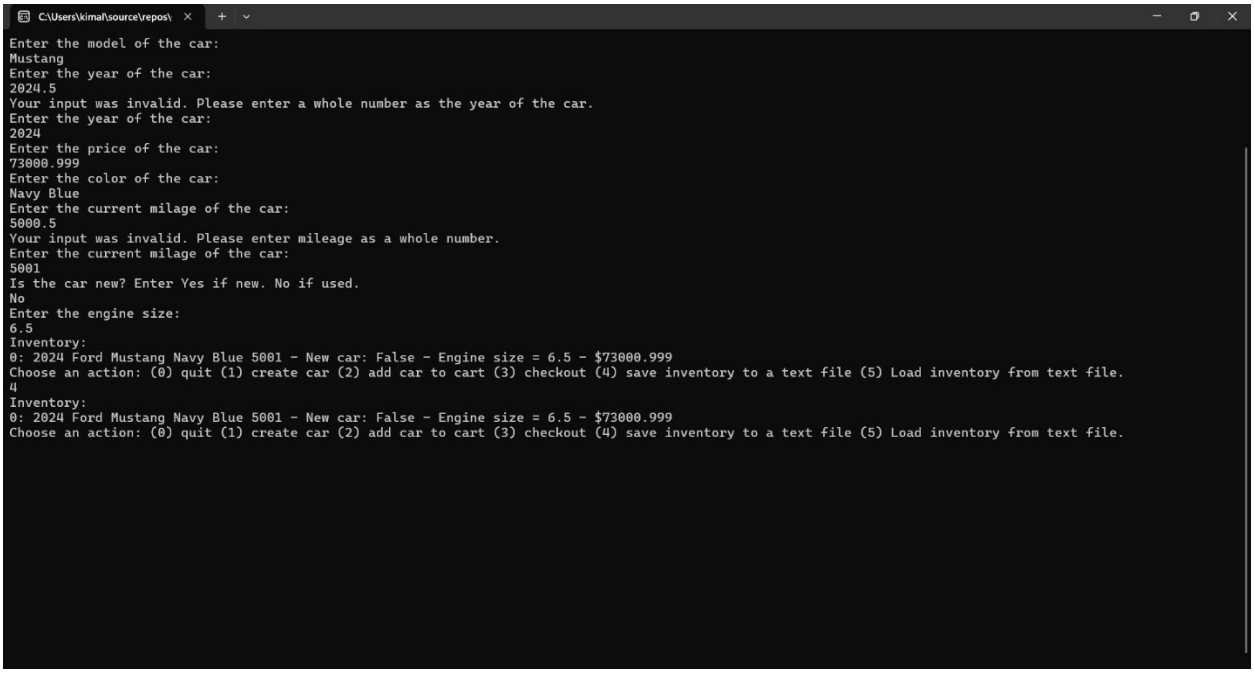
Data Storage

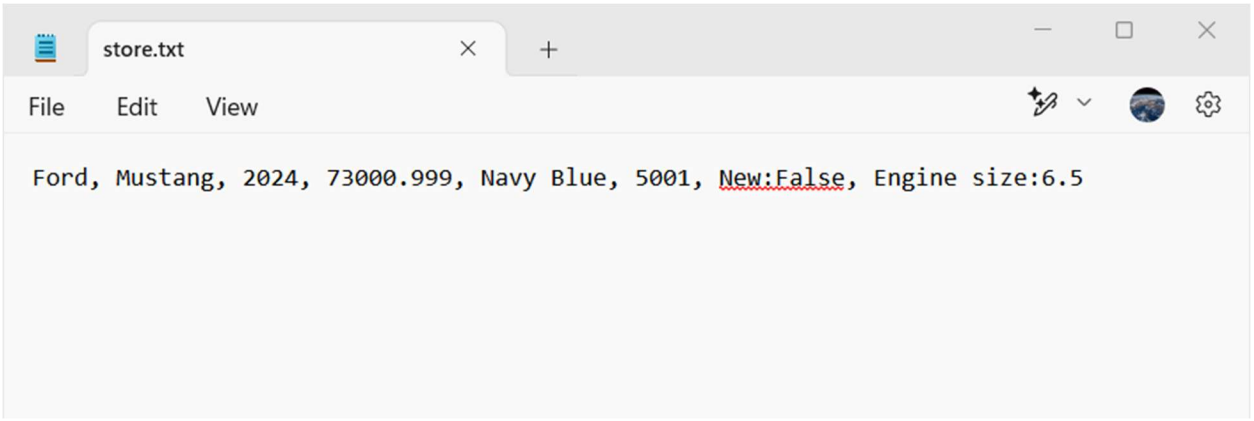




In the first image we are creating a car and saving it to our inventory text file. In the second image the we can see the inventory text file and the created car has been saved inside of it.

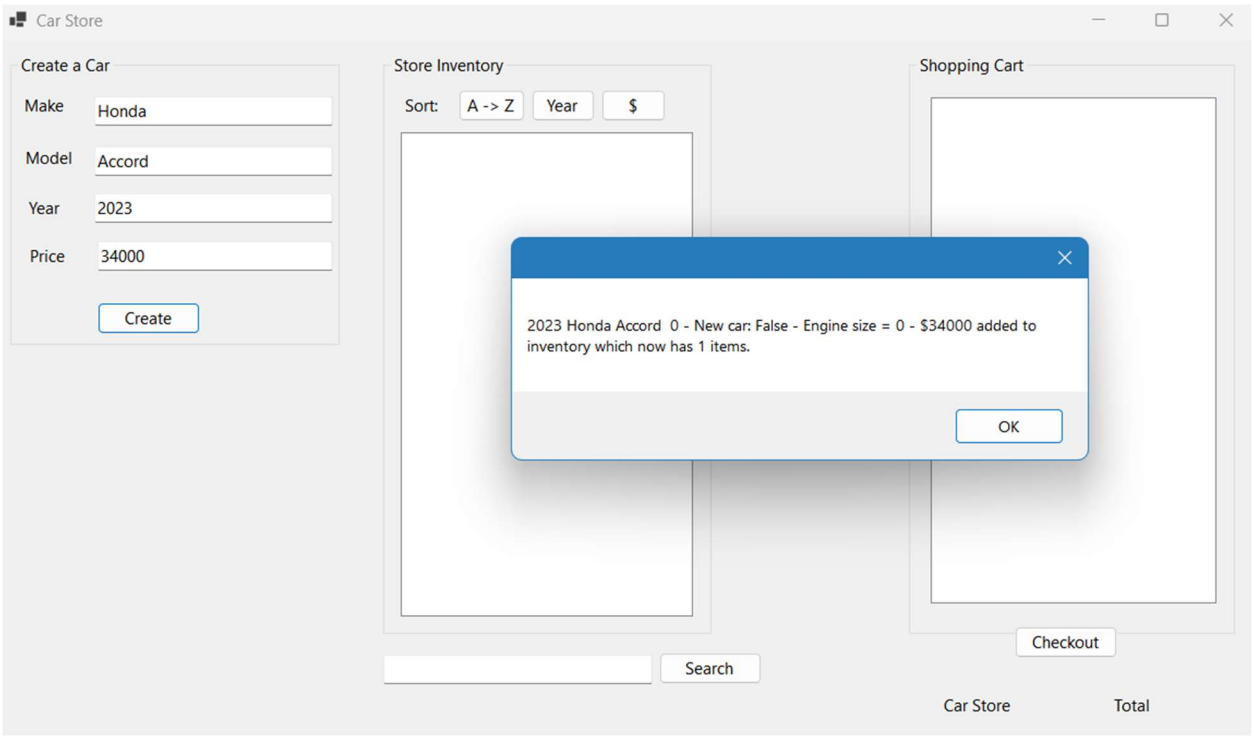
Console App Coding Challenge:



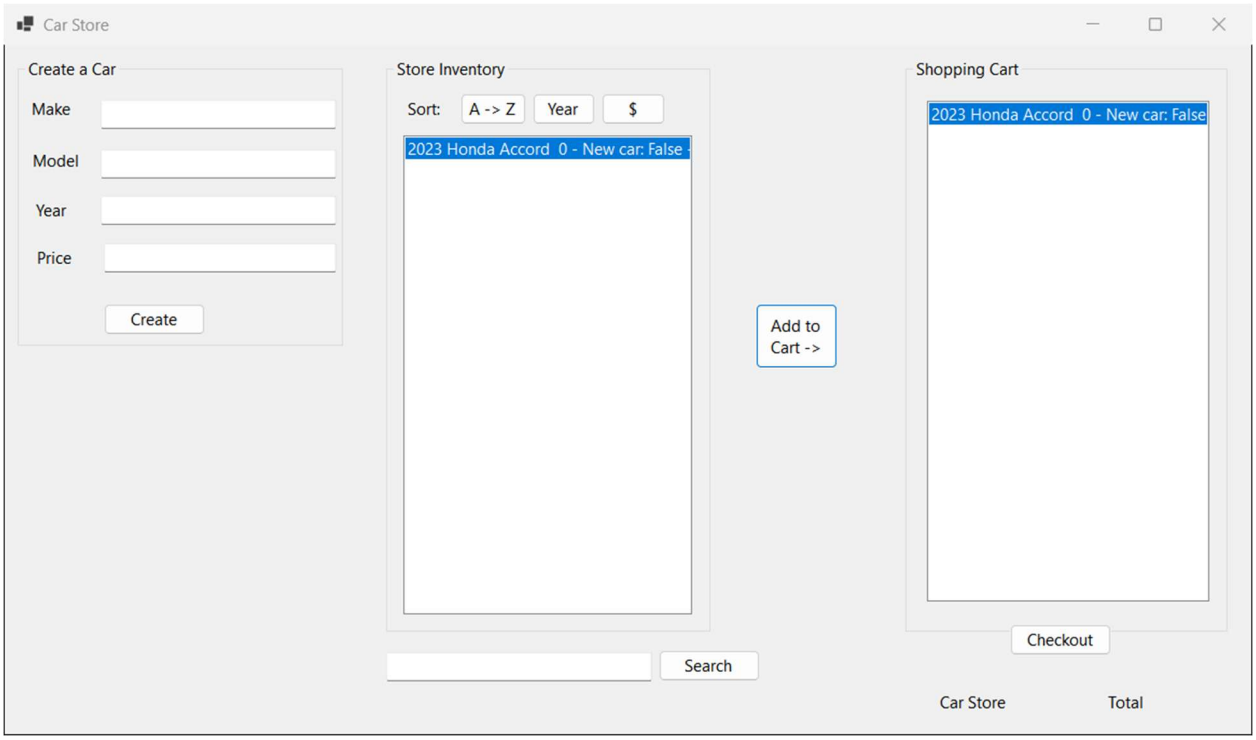


In the image above we were asked to add try and catch to our actions just in case the user inputs an invalid value. The program could handle the issue without crashing or stopping the program's progress. As we can see in the image above, I input a decimal where only integers are valid values, this caused a prompt for the user to try inputting a new valid value. The program was able to proceed after a valid input was entered. We can also see that the values were all saved in the store.txt file of inventory. For this challenge I was able to go back to previous projects I have built previously and the provided example to model how to develop the try and catch while loops.

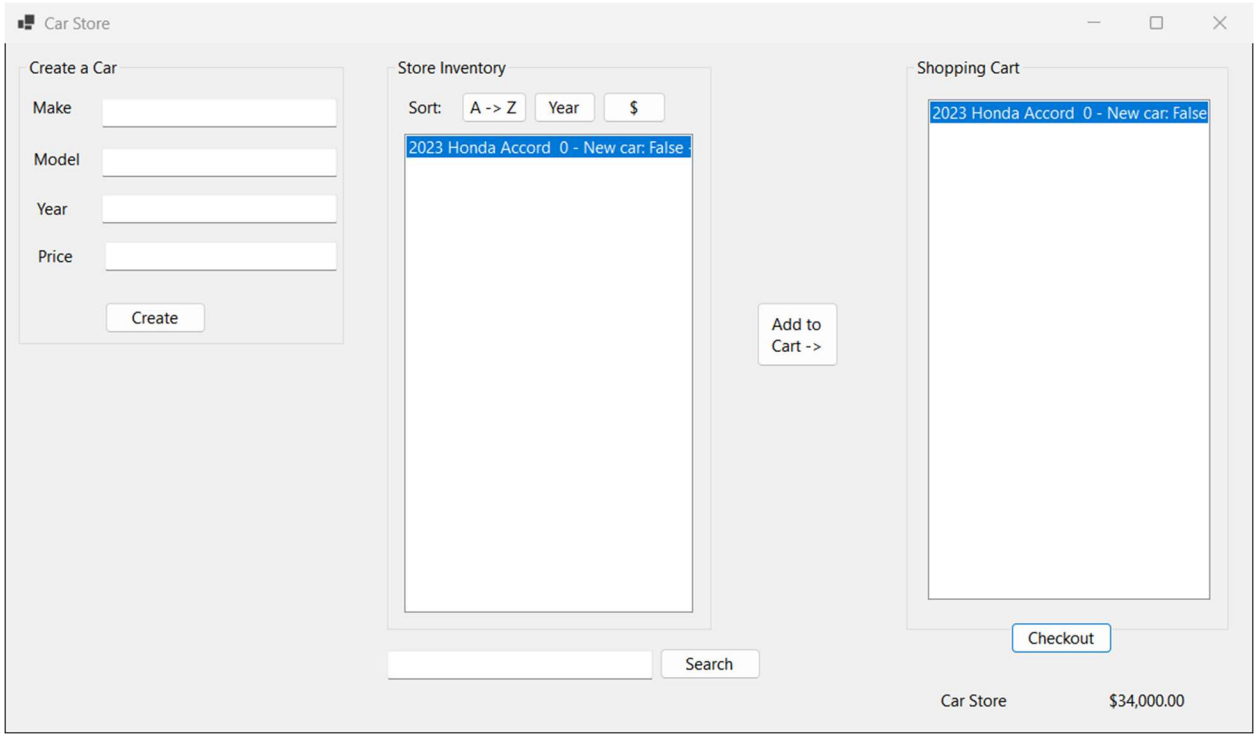
Part 3 – GUI



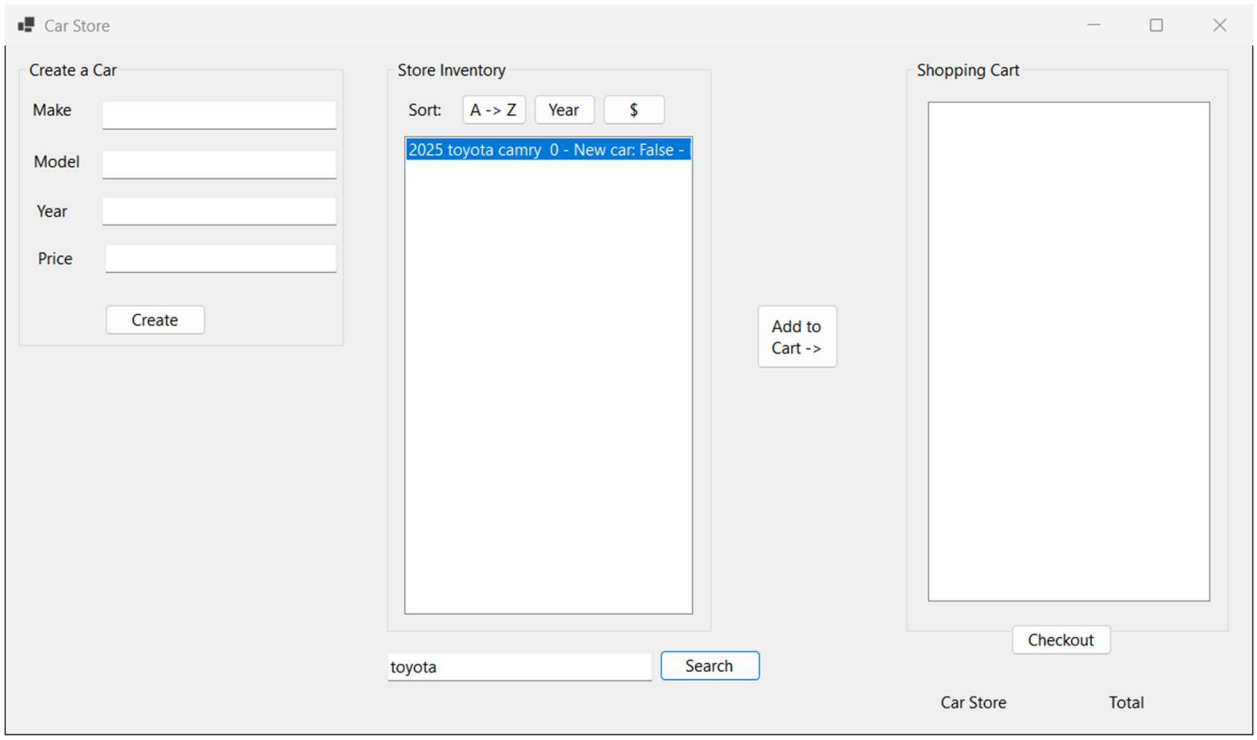
This image above demonstrates the application being implemented in a window format. The user is able to input the information for a car and once they press the create button. A secondary window opens and it lets the user know that the car has been added.



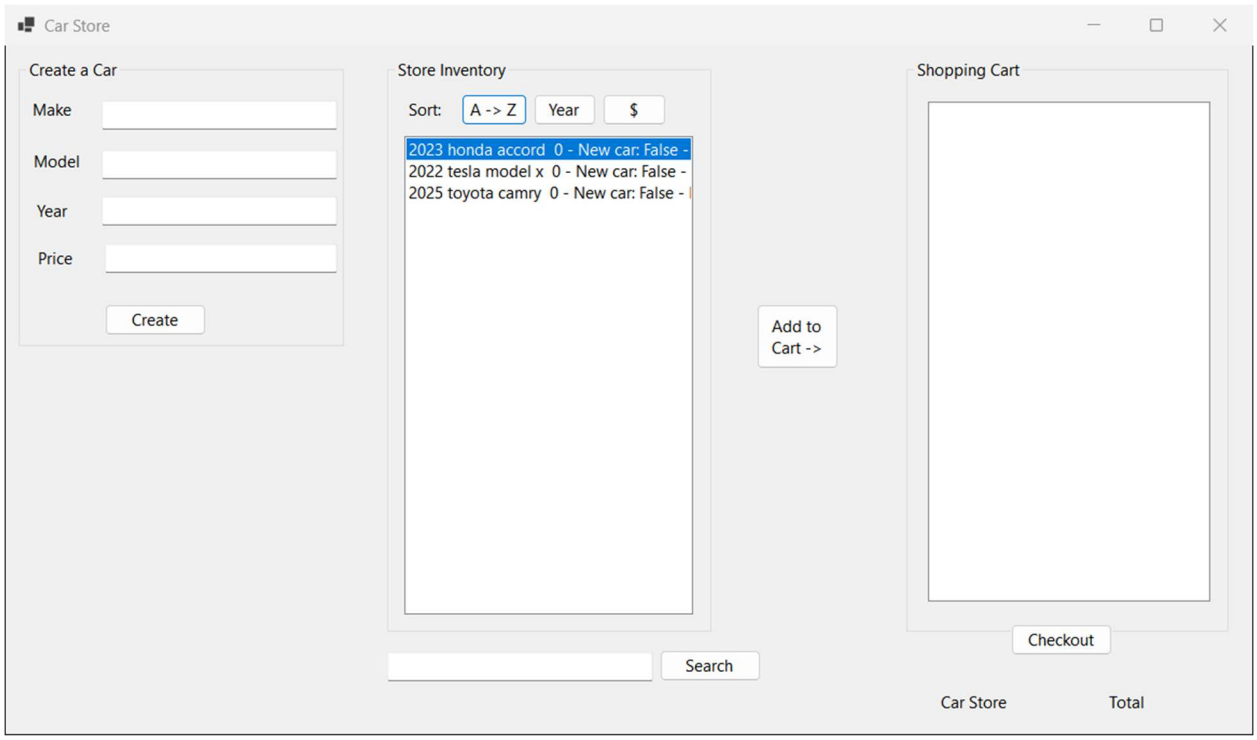
In the image above we have added the created car into the store inventory after creation/input. Then the car is being added to cart through the use of the add to cart button. Finally, we can see that the car has been added to the shopping cart list.



In the image above we have activated the checkout button to display the total cost of the shopping cart. The total is displayed where the label with the text that had said total or label6 was displayed.



In the image above the user has been able to input cars and completed a search in the cars input to find a specific car. Originally I had inputted two cars, but after completing the search only one car remained. Now I did have some difficulty with this one as I was trying to follow an old assignment, but the issue was that in my previous assignment we searched a datagridview and it was throwing me off for this assignment. However, the auto generated fillers in VS helped where I was lost.



In the image above I have sorted out the inventory list by make. I activated the sort by clicking the A -> Z sort button. This button can only put the names for the makes of the cars into alphabetical order. I was able to complete this by creating a Boolean that would check to see if the list was already in alphabetical order. I also had to add if and else statements to sort how the list needed to be sorted depending on the Boolean results.