**Info Systems Development – Report**

**Introduction**

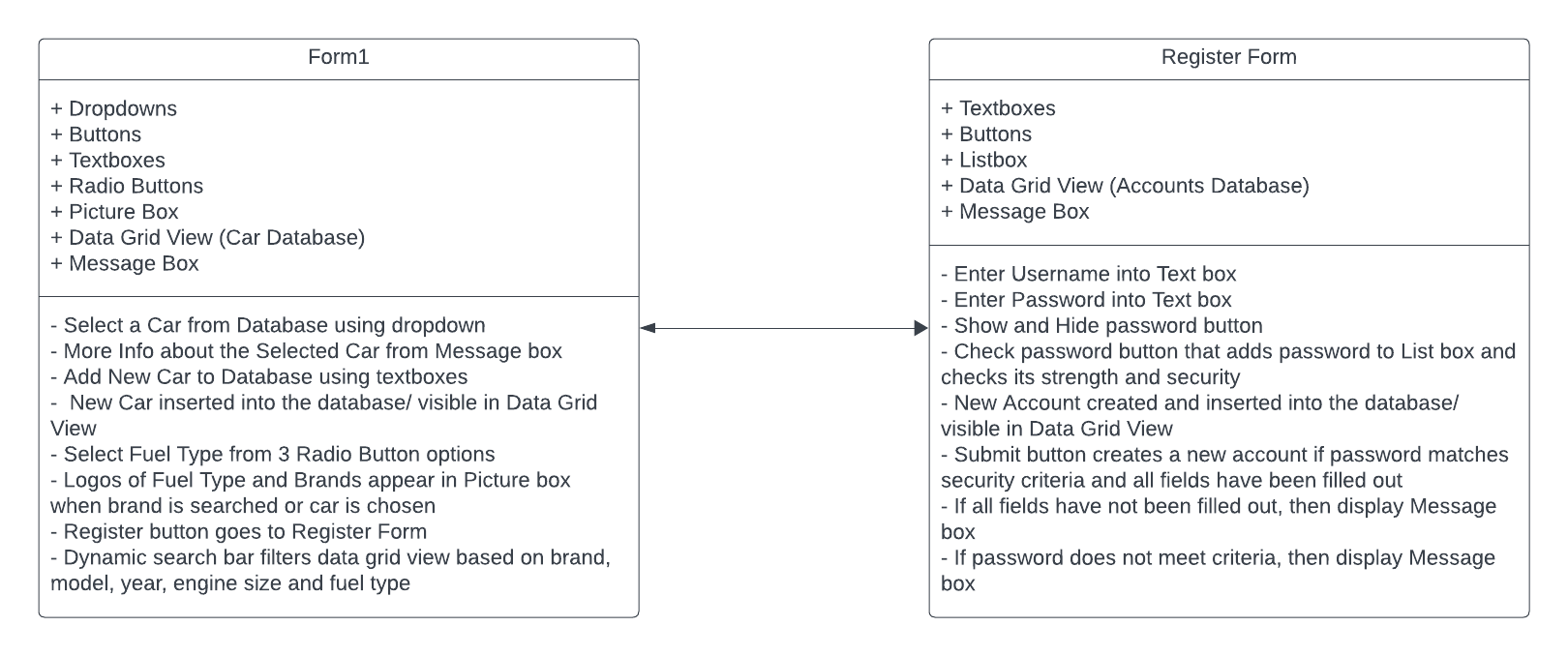
I have developed a Car Form in C# using Visual Studio with a backend SQL database containing data such as the Car ID, the brand, model, year, engine size and fuel type. The application is supposed to select and filter information about a car from the SQL database using user input from search bars and textboxes to provide them more detailed information about a car.

Some features that I have implemented into the application are being able to view the data grid view of the Car SQL Database inside the form with all the headings visible. From this data grid view they can filter each row of the table from high to low, low to high and alphabetical order simply by clicking on the table row on the grid view. I have changed the visual styling of the form to include a dark black/ grey background. I did this in Adobe Photoshop to fix the dimensions of the form so that it gives a much better background contrast. This was a much better option than having white background because it puts less visual strain on the eyes, makes it visually appealing and makes other content on the form much easier to read. I have also changed the font size to be a little bit larger, changed the colour to white to make it easier to read on a dark background. The font style has also been adjusted to a sans serif font (Rockwell) to ensure clarity of the information displayed on the software. The icon of the form has also been replaced from the Windows Form default to an icon of a car to suit the application better.

For interactive features on the car form application, I have developed a Select a Car with 2 dropdowns. When the user clicks on the first dropdown, a massive list of car brands from the SQL database appear, all arranged in alphabetical order. When the user clicks on any of the brands, they can use the second dropdown underneath that shows the list of models under that brand name that is selected. Now when the user clicks on the model of the car in the second dropdown, a message box appears displaying all the information about the car from the SQL database and shows an image of the brand logo on the form inside a picture box. The next feature that I have implemented is the ability to add a new car to the database from the form. This was done by creating an insert query for the ‘Add New Car’ button at the bottom of the text boxes. On the form the user needs to input the brand, model, year, engine size, fuel type of the car into each of the textboxes and onclick, the query will insert the data into the SQL table and then clear all the textboxes at the same time. Since there are three main fuel types, there was no need of having another textbox for so instead I added radio button functionality for either Petrol, Diesel or Electric cars. Another feature that I have implemented into the cars form application is the dynamic search bar that allows you to search by brand, model, year, engine size or fuel type. When you type in a brand into the text box, the data grid view will automatically update live to show the results from the search. Each category will filter you results live onscreen. When you search for a specific brand the brand logo will appear inside the picture box of the form. Same thing for the fuel type, when the user inputs either petrol, diesel or electric the logo for the fuel type will also appear inside the picture box. I have added a button that goes to the Register Form where you can enter your username and password into the textboxes. The password inside the textbox is displayed as a \* character but can be shown and hidden using the show button next to the textbox. I have used the lecture slides and videos for help with the password checking and adding strength to the password by making it more secure. There is also a check password button next to the show/hide button so that the user can check if their password is secure and if it meets the requirements. When you click on that button, the password is put into the list box and then feedback is given on what can be improved and gives you the strength of the password. The data grid view is also displayed on this page so you can check if the register details have been added to the accounts database. To improve functionality, I have also created an account class. On the bottom of the form, there is a submit button which uses this class to check if the password contains at least 10 characters, an uppercase and lowercase letter, a special character and a symbol. If your password does not meet the criteria, then a message box will appear asking the user to re-enter the password again to make the password more secure. If the user forgets to fill in both the password and the username text boxes, another message box will ask to fill out all the fields to continue. If all is correct, then the submit button will use the insert query into the SQL accounts table, user will get a message that registration has been successful and then the new details can be viewed inside the data grid view.

The features that I have implemented but are still not fully functional is the Add New Car insert query because when the user inserts a new car into the database with a brand name that already exists, it will create a second brand with the same name. When you search for that brand using the Select a Car feature on the form, inside the dropdown you will see 2 of the same brand names. The insert query is working however, if the brand already exists in the database, the model should be added into the same brand category so that the new car appears under the already existing brand. Even though that it is fully operational, it could further be improved and added more functionality so that many more cars can be inserted into the database in the future. Also, a feature that I wanted to implement is the ability to add a logo into the picture box if a new brand is inserted. However, due to the lack of resources and time, this would be a lot more difficult to implement because the picture box already has a lot of if and else if statements depending on which brand has been chosen, to display that logo of that brand. Meaning that if the user wants to add a new car into the database, the image that appears inside the picture box, when brand is searched in the dynamic search bar or by selecting the car in the dropdown will no longer work. Data inserted into both SQL tables has an issue where data is displayed and inserted into the SQL table, however when the form is closed any changes are not permanently saved in the database.

**Graphical Model of Application**



In developing both the Cars Form and Register Form for the application, I found it important to create a class diagram for the graphical model. The class diagram served as a blueprint, providing a clear visual representation of the structure and relationships between different components of the forms. For the Cars Form, the class diagram helped organize the various elements such as car details, input fields, and database interactions. Overall, the class diagram allowed for effective planning, and implementing both forms, resulting in well-structured and functional components within the application.

**Data Dictionary**

Car Table:

* Car ID – Autoincrementing car ID for each car inside the database.
* Brand – The brand of the car.
* Model – The model of the car brand.
* Year – The manufacturing year of the car.
* Engine Size – The engine size of the car.
* Fuel Type – The type of fuel the car uses (Petrol/ Diesel/ Electric)

User Interface Components:

* Select a Car
  + Dropdown1 – Car Brand.
  + Dropdown2 – Car Model.
* Add a New Car
  + TextBox1 – Enter Car Model to add to database.
  + TextBox2 - Enter Car Brand to add to database.
  + TextBox3 - Enter Car Year to add to database.
  + TextBox4 - Enter Car Engine Size to add to database.
  + Radio Buttons – Select Fuel Type out of 3 options (Petrol/ Diesel/ Electric).
  + Submit Button – Runs Insert Query to take all fields, insert into the database and clear all textboxes.
* Dynamic Search Bar
  + TextBox1 – Search/ Filter Car Details by brand, model, year, engine size and fuel type.
* Data Grid View - DataGridView for displaying the list of cars from database.
* Database Operations
  + Insert Query – Insert New Car into Car database.
* Images
  + Brand Logos – Images representing the car brand logos.
  + Fuel Type Logos – Images representing the fuel type logos used for cars.

Register Form:

* Enter Username and Password
  + TextBox1 - Username
  + TextBox2 – Password displayed in \* characters
* Show/ Hide Button – Password is originally hidden but when clicked will show the password in word format, text on button will change to hide and then password can be hidden again.
* Check Password Button – Adds password in textbox into list box showing the number of characters, the strength of the password and gives guidance on how the password can be made more secure. Checks if password needs more characters, needs an uppercase/ lowercase letter, a special character and a number.
* Password Requirements – Listed underneath the password box
* DataGridView – Accounts database listing your account when you register.
* Database Operations
  + Insert Query – Insert New Account into Accounts database

**Test Plan and Results**

1. Objective - The objective of this test plan is to ensure the functionality, usability, and reliability of the Cars Form and Register Form developed for the application.
2. Scope - This test plan covers testing of the Cars Form and Register Form, focusing on various features such as user input, database integration, visual representation and other user interface elements such as functionality.
3. Testing Approach - Testing will be conducted using both manual methods. Manual testing will involve testers using the software and checking if each feature is working and potential areas of improvement.
4. Testing Cases
   1. Car Form
      1. Verifying that all input fields (brand, model, year, engine size and fuel type) accept user input correctly.
      2. Testing the functionality of the “Add New Car” button to add a new car to the database.
      3. Validate the dynamic search bar feature by searching for different car attributes.
      4. Check if the data grid view displays car database entries correctly.
      5. Test filtering functionality by sorting data in the data grid view.
   2. Register Form
      1. Test username and password input fields to ensure they accept user input.
      2. Verify that the form allows submission only if password meets the security requirements (contains at least 10 characters, at least 1 uppercase/ lowercase, 1 digit and 1 special character).
      3. Validate database integration by checking if the new user accounts are successfully inserted into the database.
      4. Test form validation by attempting to submit the form with missing or invalid data/ data that doesn’t meet requirements.
5. Test Environment
   1. Operating System: Windows 11
   2. Development Application: Visual Studio 2022
   3. Database: SQL Server
   4. Project Framework: .NET Framework
6. Test Results
   1. Car Form
      1. Input fields accept user input correctly without any issues.
      2. “Add New Car” button successfully adds new cars to the database.
      3. Dynamic Search bar functionality works as expected, filtering data based on user input.
      4. Data grid view displays car database entries correctly.
      5. Filtering functionality sorts data in the data grid view correctly.
   2. Register Form
      1. Username and password input fields accept user input as intended.
      2. Form allows for competition only when the password meets the security requirements.
      3. New user accounts are inserted into the database upon successful completion.
      4. Form validation prevents submission with missing or invalid data, displaying appropriate error messages through message boxes.

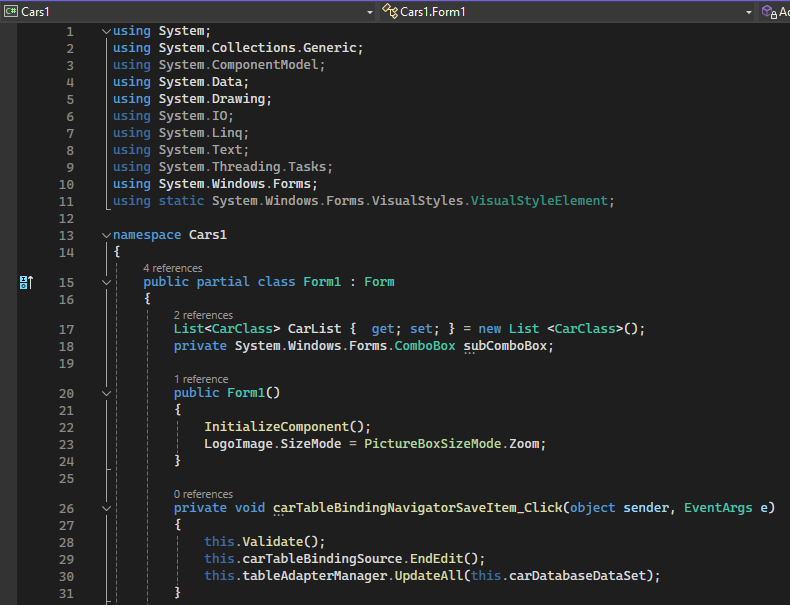
**Software Development Lifecycle**

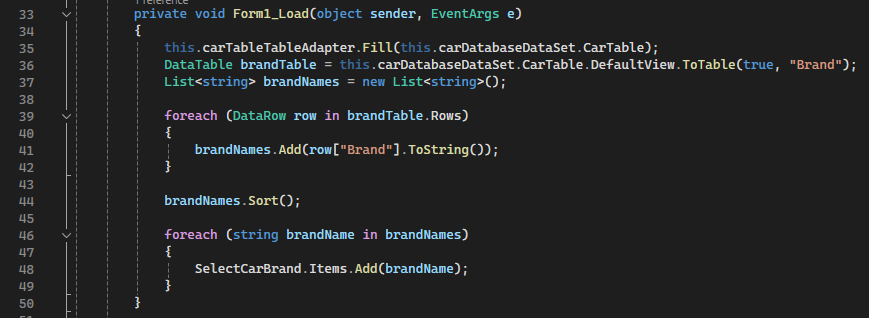
In the scrum development lifecycle, I have made sure to define project goals and requirements for the cars form project. Therefore, in the project initiation stage, I have analysed how finding a car in a database can be made more efficient with the use of a forms project. When planning the project, I was thinking about what features could be implemented into the application to make finding a car easier. I have scheduled a short time frame to develop each component and feature of the project so that all features can be fully functional without any issues. This way, after each feature has been implemented, it can then be improved upon after each scrum cycle. The scrum development lifecycle is suited better for completing small projects within a very short time frame. However, due to the lack of resources in this development lifecycle, certain features or bugs could be present as the development team may not have the adequate resources and personnel to fully complete the project. During each stage of the scrum methodology, I have scheduled regular meetings and testing to reflect and identify areas of development and improvement. Therefore, I can quickly deliver a product to the stakeholders through regular testing and meetings. By the end of the project the team will be able to document the successes, challenges faced in this project and areas of improvement to this project that could be made in the future, if more time and resources would be available to their disposal.

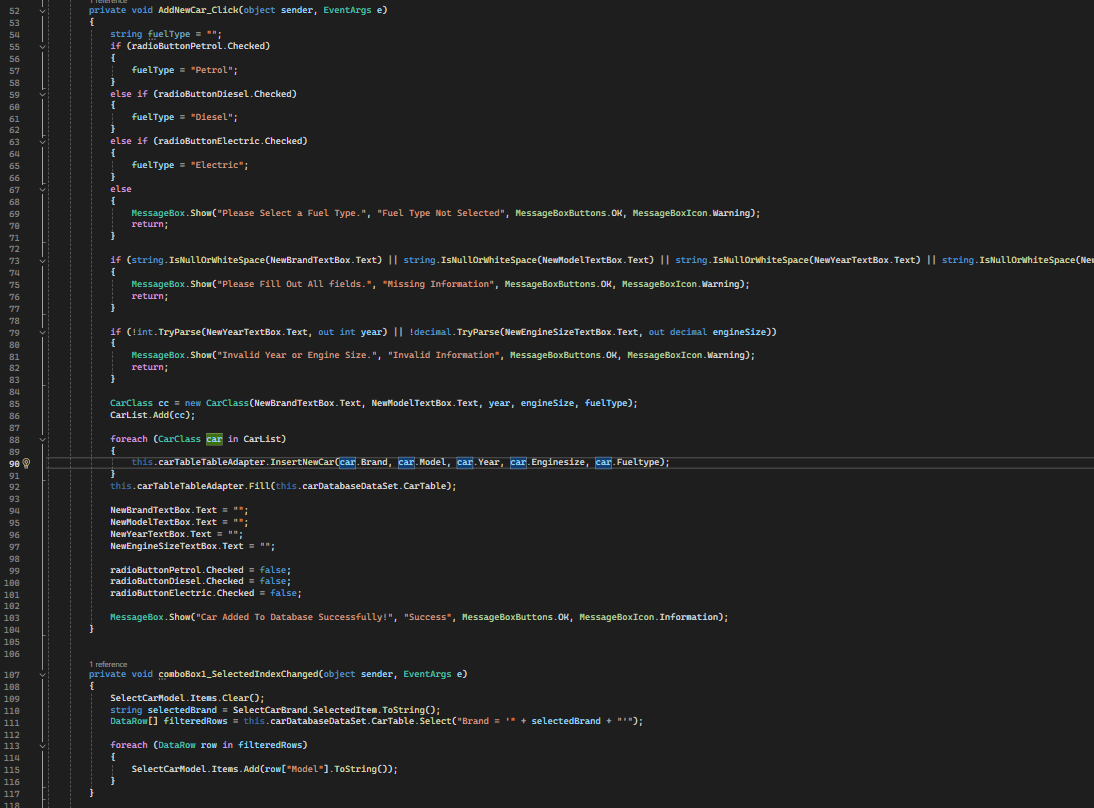
In the implementation stage of the scrum development lifecycle, first I have developed the Car Table database in the form. Then inserted all the cars into the fields to give the team more fields to work with and develop new features for the application. Then the data grid view was added to the form so that the user can view the entire database. After the next meeting, the Select a Car dropdowns have been implemented to be able to quickly search for a specific car brand and model. Then you are provided with more information about the car from the database, displayed in the message box with an image of the car logo also appearing in the picture box. Next the Add a New Car insert query was added in this stage so that the user can insert a new car into the database if they wish using the given fields and textboxes. Extra functionality was added with the dynamic search bar, allowing the user to manually search for any field. Then the data in the data grid view is filtered according to that search. After a bit of testing and discussion in the meeting, it was decided that to further add interactivity to this feature it would be good to add images of logos and fuel types depending on the user’s input in the dynamic search bar. During the implementation stage of the Scrum development lifecycle, a significant addition was the creation of a Register Form with password validation. At first, the focus remained on refining the Car Table database within the form. Once this was solid, attention turned to integrating the Register Form functionality. The Register Form allowed users to create new accounts, with a key feature being password validation. Passwords had to meet specific criteria: they needed to be at least 10 characters long, contain at least one lowercase letter, one uppercase letter, one digit, and one special character. When users submitted the Register Form, the system checked if the password met these criteria. If it did, the new account details were securely stored in the database. If not, an error message asked users to revise their password. This addition ensured better account security and guided users in creating stronger passwords. Overall, it was a crucial improvement in the usability and security of the Car Form application.

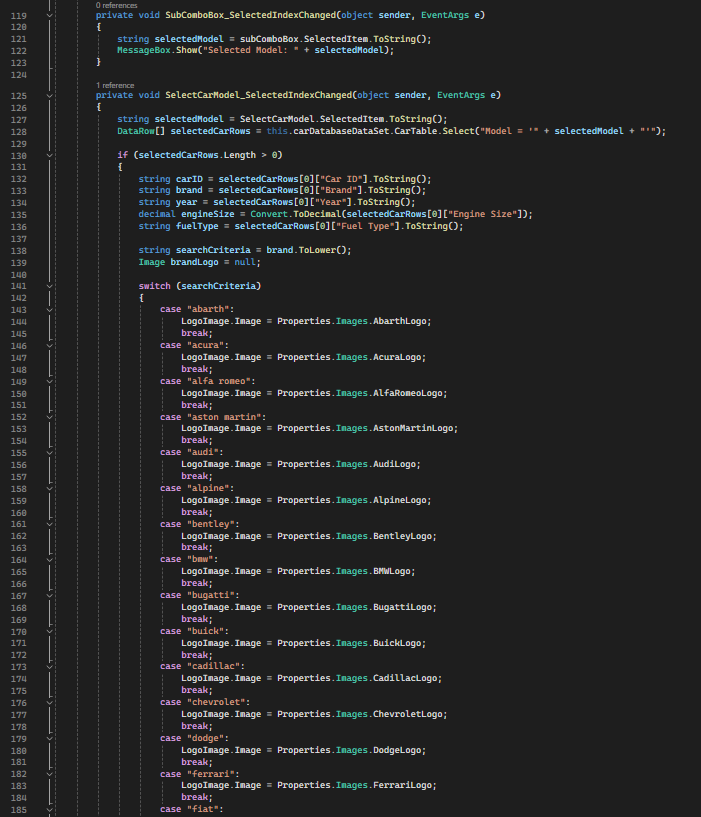
**Printout of Code**

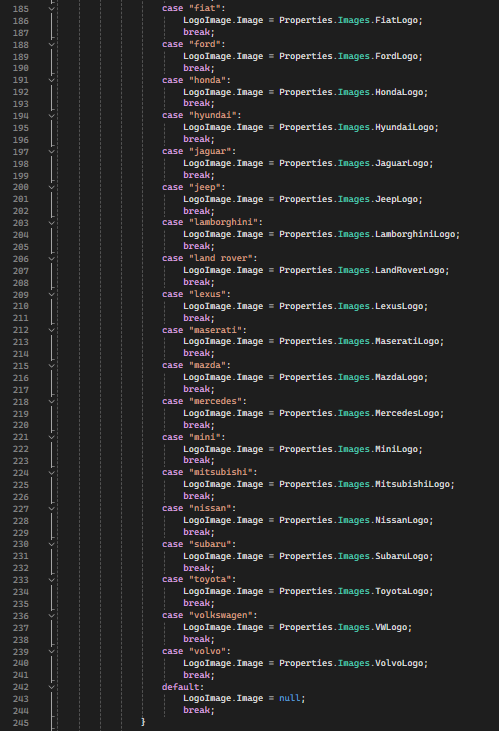
Form1.cs

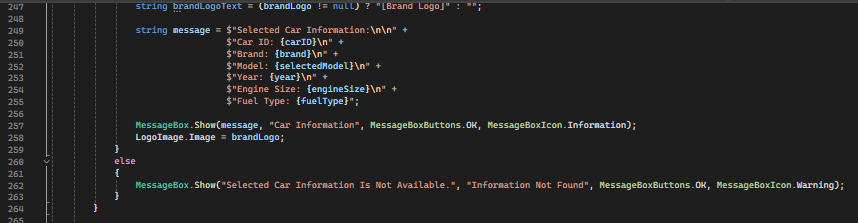


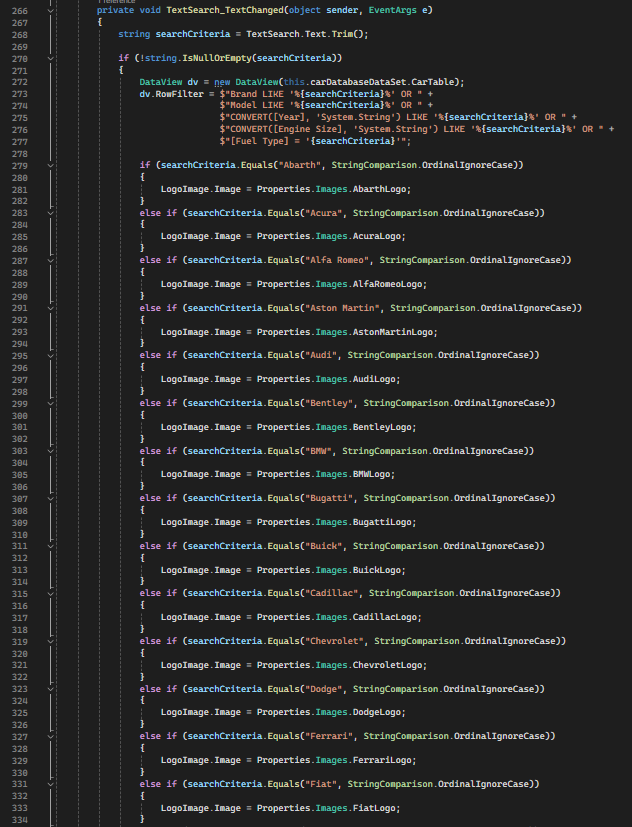


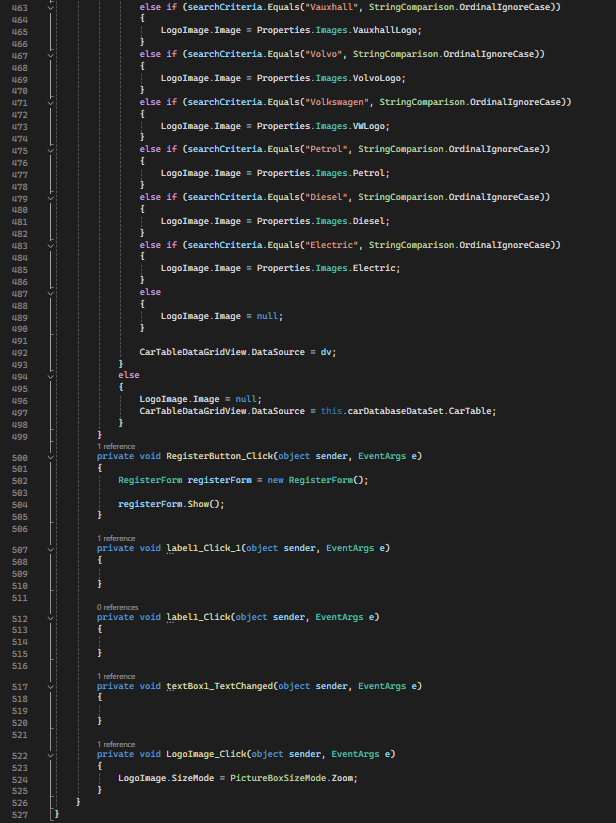




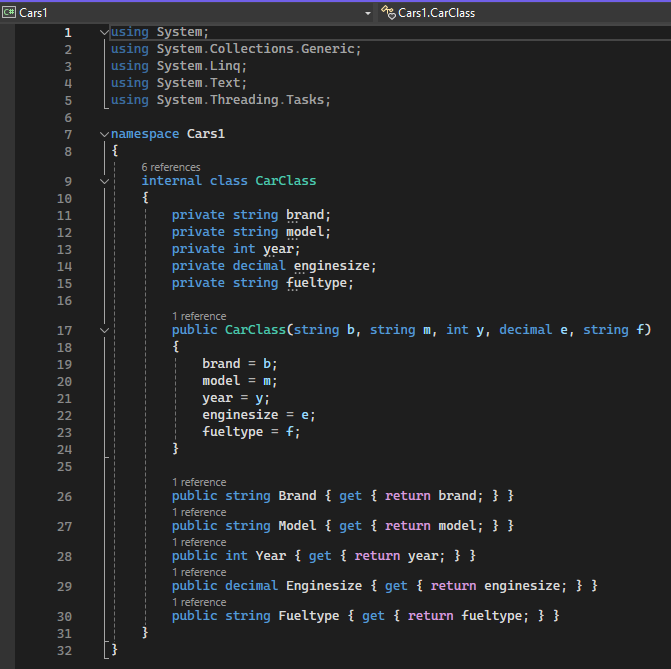




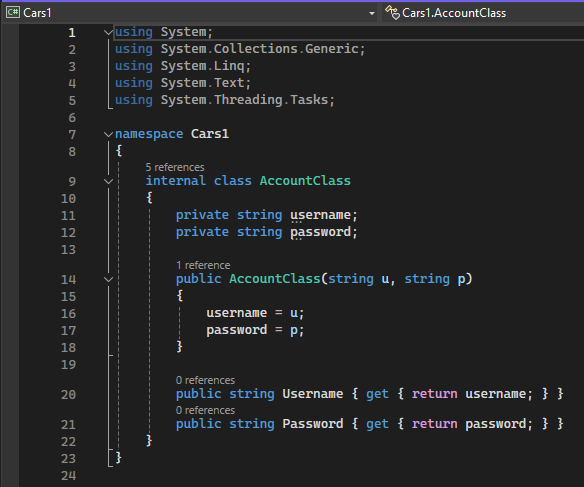




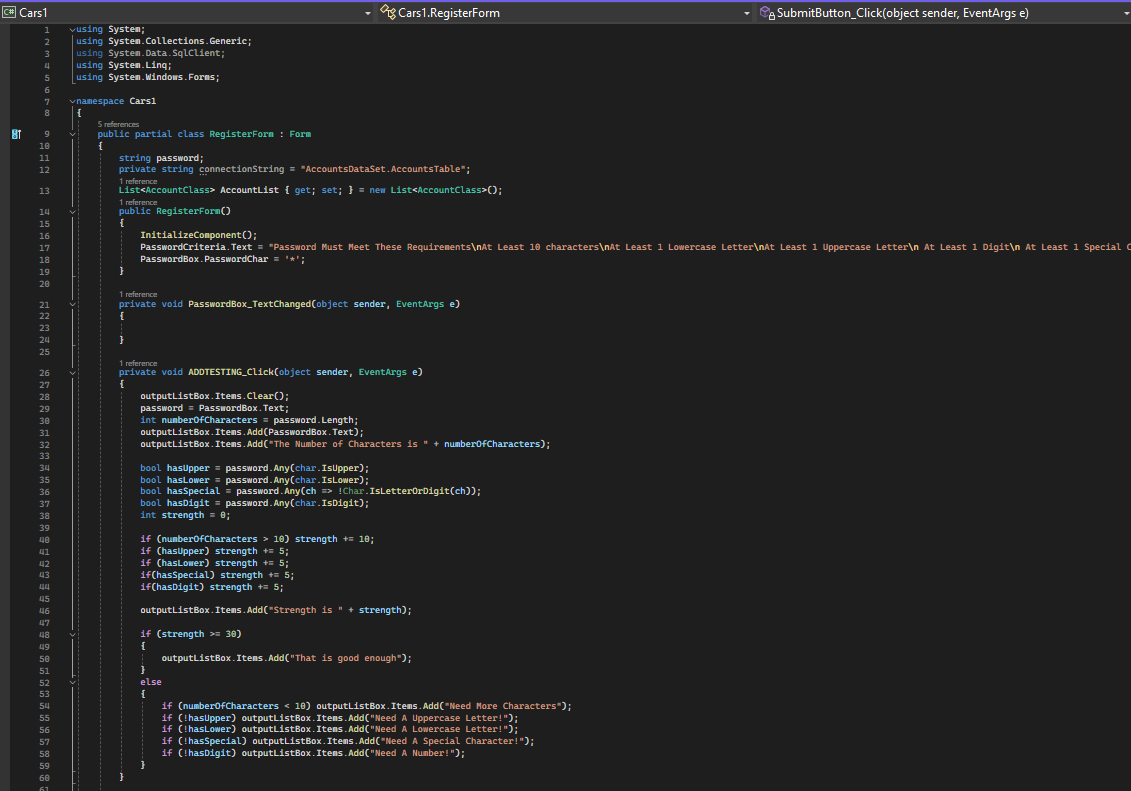
CarClass.cs

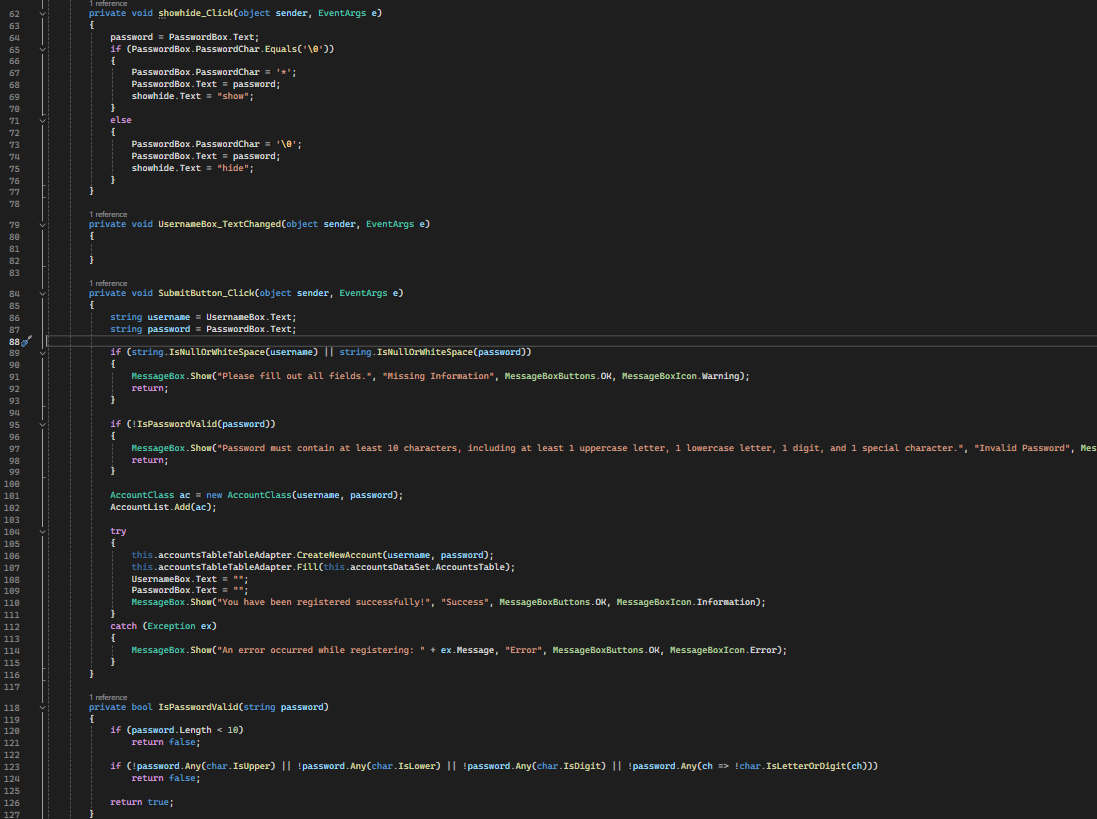


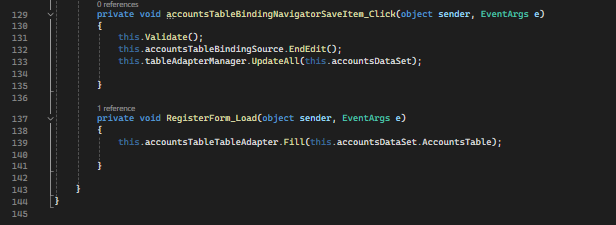
AccountClass.cs



RegisterForm.cs



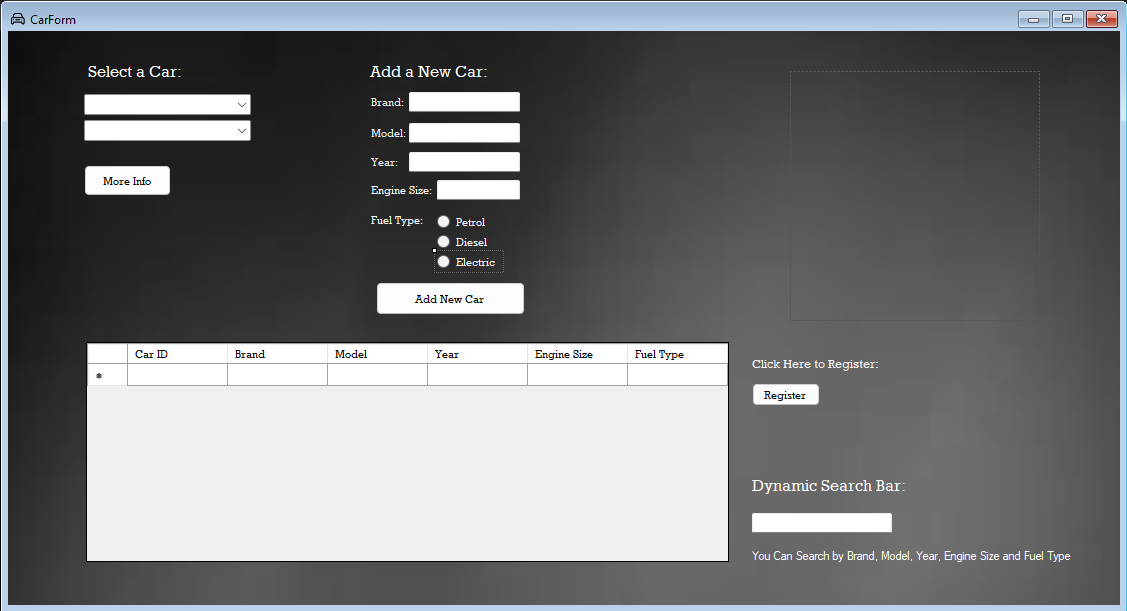




**Step-by-Step Guide**

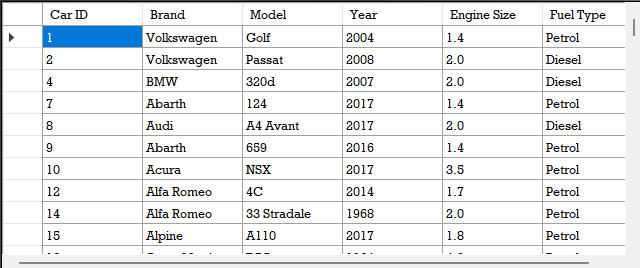
* Step Description
* Small Print Screen and Output of your software (up to 5 pages)

Opening the Application:



* Launch the Car Form Application by running the .exe file.
* When opening, you will notice the dark black background, that helps reduce visual strain and enhances readability.

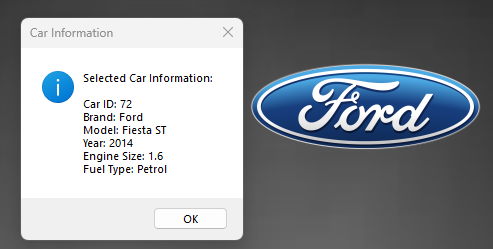
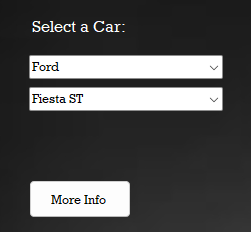
Viewing the Car Database:



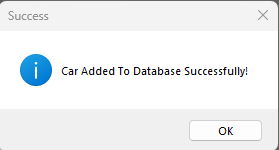
* On the form, you will see a data grid view displaying the Car SQL database with all the table rows visible.
* By clicking on any row in the grid view, you can filter the table between high to low, low to high and alphabetical order.

Interactive Features:

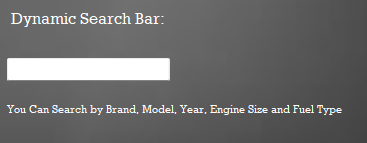
* Select Car Dropdown

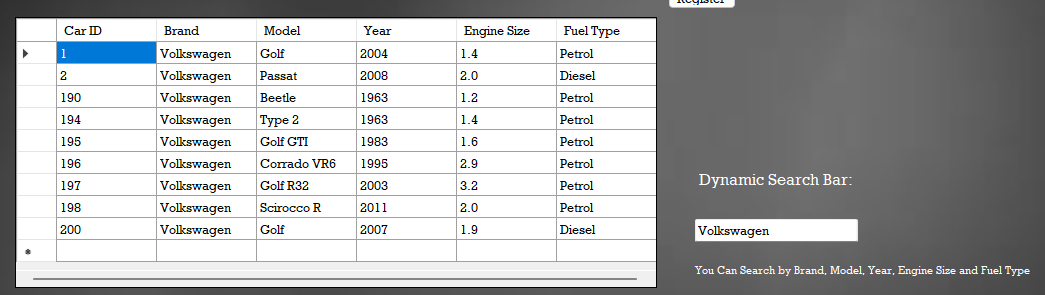


* + Click on the first dropdown to view the list of car brands arranged in alphabetical order.
  + Select a brand to populate the second dropdown with corresponding car models.
  + Choose a model from the second dropdown to view more detailed information about the car model and the brand logo displayed in the picture box.
* Adding a New Car

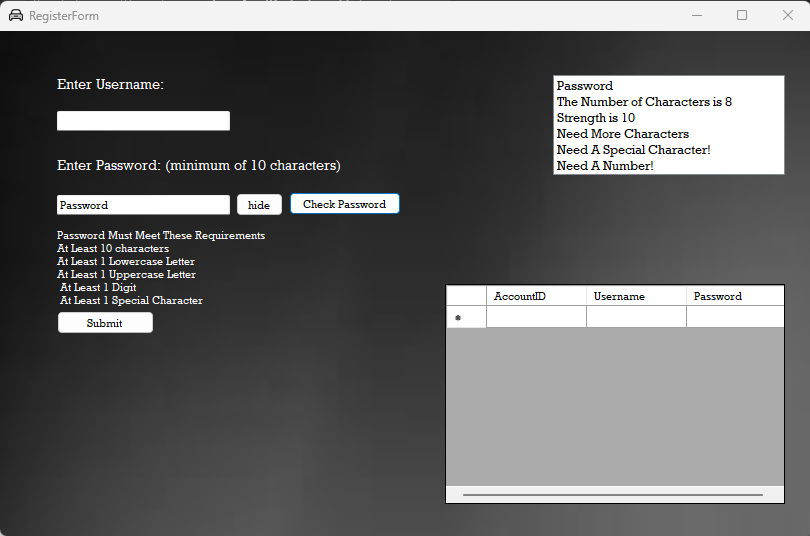
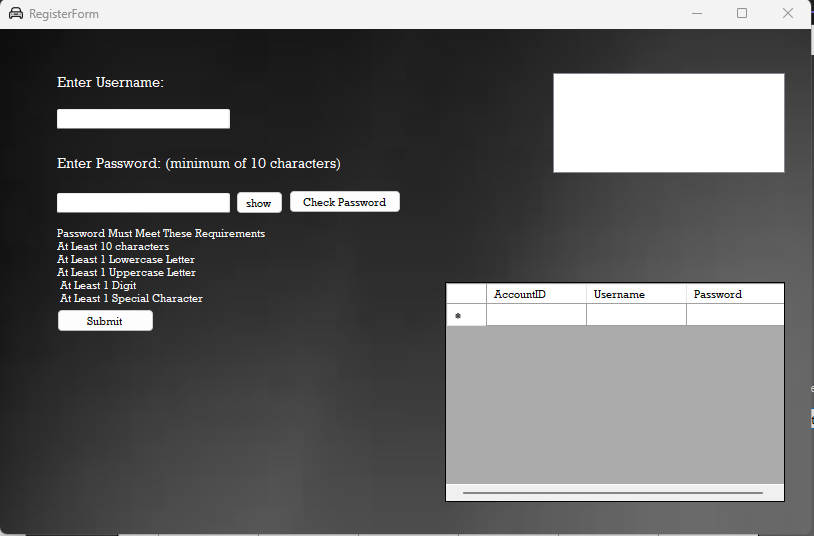


* + Fill in the text boxes with the brand, model, year, engine size and select the fuel type using the three radio button options (Petrol, Diesel, Electric).
  + Next click the “Add New Car” button to insert the car data into the SQL table.
  + The Form will clear all text boxes after insertion for ease of adding multiple cars.
* Dynamic Search Bar





* + Type in the search bar to filter results by brand, model, year, engine size or fuel type.
  + The data grid view will update live to display search results.
  + Brand logos will appear in the picture box when searching for a specific brand or when searching of a specific fuel type.
* Register Form



* + Enter a username and password that contains at least 10 characters, one uppercase and lowercase letter, one special character and at least one number.
  + You can show and hide your password in the textbox by clicking the show/hide button next to the text box.
  + You can test the strength of your password by clicking the ‘Check Password’ button.
  + When you click submit, your registration details will be created and inserted into the Accounts database. This can be viewed in the data grid view.
* Known Issues and Future Improvements
  + Brand Duplication – If a brand already exists in the database, adding a new car may create duplicate brand entries.
  + Brand Logo – At the moment, the picture box displays logos based on predefined if-else statements for existing brands and therefore, adding new brands may disrupt this functionality.
  + Data Inserted into Database saved – If any data is inserted into either the car or accounts database, it can be viewed in the data grid view, however it will not save when you exit or close the form.
  + Future Improvements – Enhancements could include refining the insert query to prevent brand duplication and implementing a more efficient solution for displaying brand logos and adding new logo images to the form. Any new data inserted into the database should be saved automatically and should not require any manual input from the user.