CarVendor - A Used Car Marketplace

**Project Title:**

CarVendor

**Group Number:**

Group 12

**Team:**

Thomas Koutsidis, Jayat Rateria, Kelly Craig

**Section 1:**

Website URL: add link here

Public Github URL:

<https://github.com/JAYAT-24/Project-CarVendor-Django-Sqlite.git>

**Section 2:**

**Technical Details**

**Database**

The application uses an SQLite database called Full\_Car\_Database.db to store information about cars, car attributes, car history, dealers, and pricing. The SQLite Full\_Car\_Database was created and integrated using the SQLite3 python module (Sqlite3, 2023). This allowed us to write the code for data processing and database creation in a jupyter notebook. We started with a csv file containing web scrapped used car information that was obtained from Kaggle. In our jupyter notebook, we read in the file and completed data processing steps including removing rows with na values and changing the data types. Then we created and inserted the data into the database which contained primary keys, foreign keys, and database constraints. Some of our database constraints included not allowing null values, max character lengths, and requiring floats to be within a specific range. We chose a SQLite database because it is a serverless, self-contained database system that is simple to set up and use (About Sqlite, 2023). The database is easily portable because it is contained in a single file. This made sqlite a perfect option for our small application which needed a relatively small amount of data storage.

**Website**

The project was set up using Django, a high-level Python web framework (Django, 2023). The necessary Django files and directories were created using the django startproject and startapp functions to organize the application. The main components to create a django website are a url file, views file, HTML templates, models file, manage file and more (Django Tutorial, 2023). The url file defines the url patterns to allow users to access a specific url on the website and the corresponding view functions that handle requests (URL File, 2023). The views file contains the view functions that generate dynamic content for the website. View functions receive requests from the url and interact with the database to fetch data, process it, and render the HTML templates (Django Tutorial, 2023). HTML templates define the structure and presentation of the website's pages. The models file defines the data models, which are Python classes that represent database tables and their fields. The manage.py file is used to run development servers, create database tables, apply migrations, and perform other administrative tasks related to the Django project (Manage, 2023). These files that we created and modified, work together to produce a functional Django web application. The specific functionalities that we built using the Django framework will be described in the functionalities section.

**Visualizations**

The visualizations in the CarVendor website were made using Matplotlib which is a comprehensive data visualization python library (Visualization, 2023). In the Django views file, we created functions that queried the Full\_Car\_Database database and created plots with the retrieved data. The database queries were written in SQL and they performed complex operations including retrieving data, joining tables, calculations, filtering, and more. This data is displayed as plots using the python library Matplotlib and the plots are saved as png files. The data is converted to a data URI, which can then be embedded directly into the image\_page HTML page. This process allowed the Django website to display the plots on the visualization page.

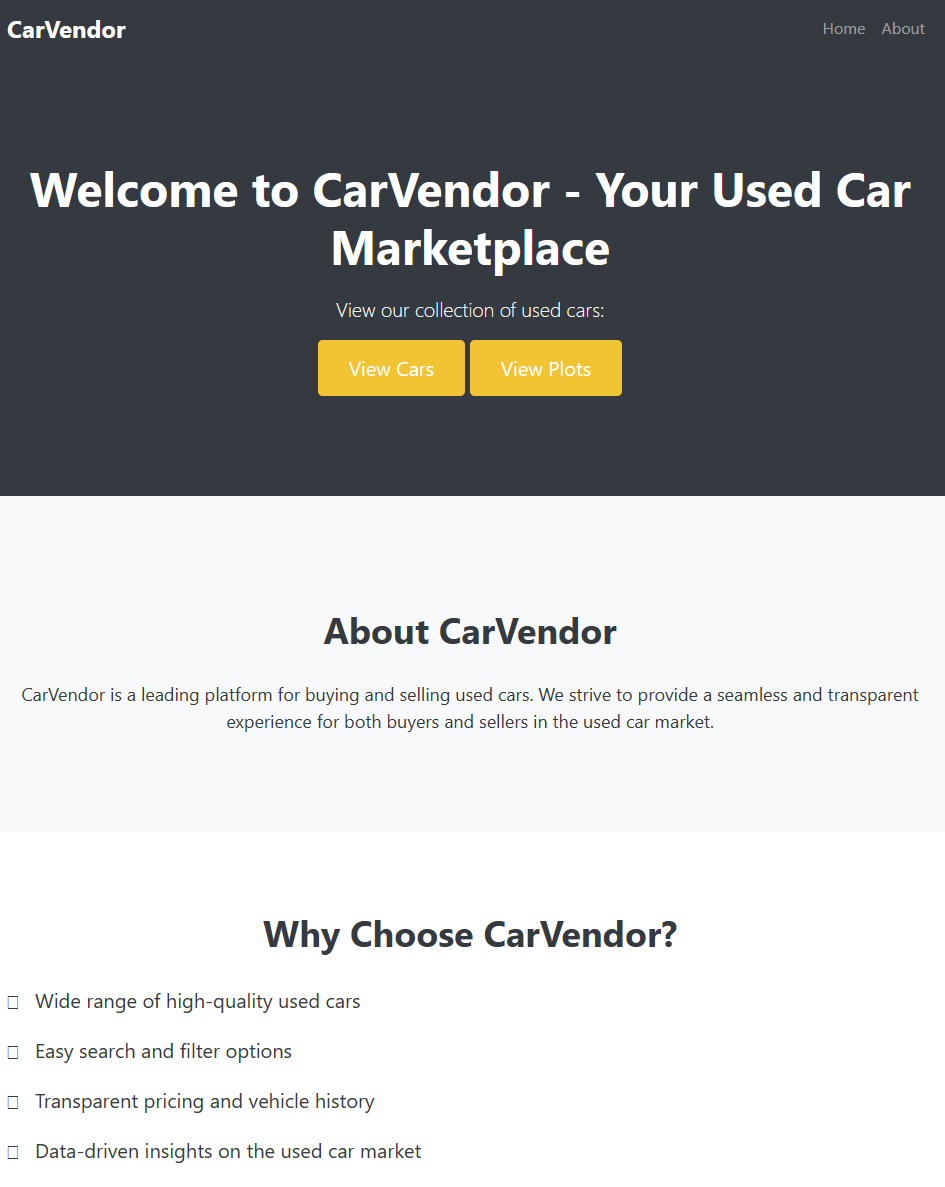
**Lessons**

Building a database and website was a highly collaborative effort and we learned many lessons while completing this challenging project. We applied what we learned during the class to design and create a sqlite database using the SQLite3 python module. In addition, we applied what we learned in class to write SQL queries and retrieve data from the Full\_Call\_Database. After database creation and writing queries, we independently learned how to create a website and visualize the data with plots. It can be challenging to understand how each aspect interacts when developing a website for the first time. We had to learn the core concepts of website design and the components of the Django framework including models, views, templates, and URL routing. After understanding the components of website creation, we learned how to integrate a database with Django and how to interact with the database to perform CRUD operations and manage data efficiently. We learned how to use matplotlib to create key visualizations and how to integrate these visualizations into the views, url, and template files to display on the website. A critical component of the django framework is template design, so we learned how to create dynamic and reusable templates for rendering data and structuring the website. To effectively view the pages of the website, we learned how to create organized urls to access all of the website views. Finally we learned about version control, maintaining code integrity, and how to effectively collaborate using the version control software Git. Creating a database and building a Django website was a valuable learning experience and we have become more experienced developers throughout the process.

**Functionalities**

Our goal for this project was to create and deploy a used car website which allowed users to perform CRUD operations, filter through a table displaying key information, and view important visualizations on the dataset. We successfully accomplished each of these features and included even more functionality in addition to meeting these goals. Building off of our original plans, we decided to create a Django web application for a used car marketplace called "CarVendor." It allows users to view a collection of used cars, filter them by manufacturer, view detailed information about individual cars, and it also displays visual insights about the used car market. The details about each feature are described here:

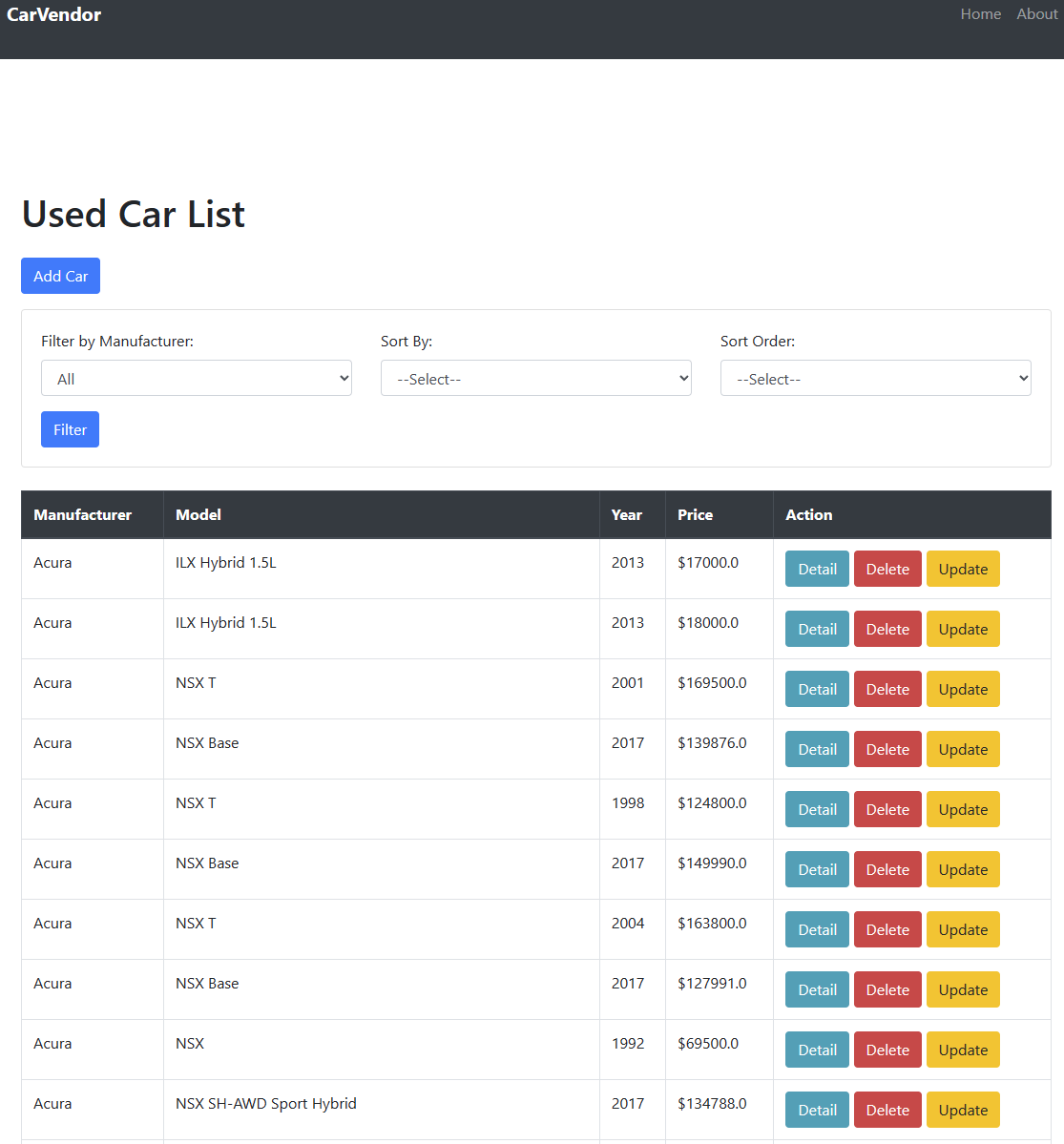
* Home page: The website features a home page which introduces the user to CarVendor and displays several buttons including a button to view the available cars, a button to view data visualizations, and a button to view the about page. The home button on the website allows users to return to the home page of the website from any page.



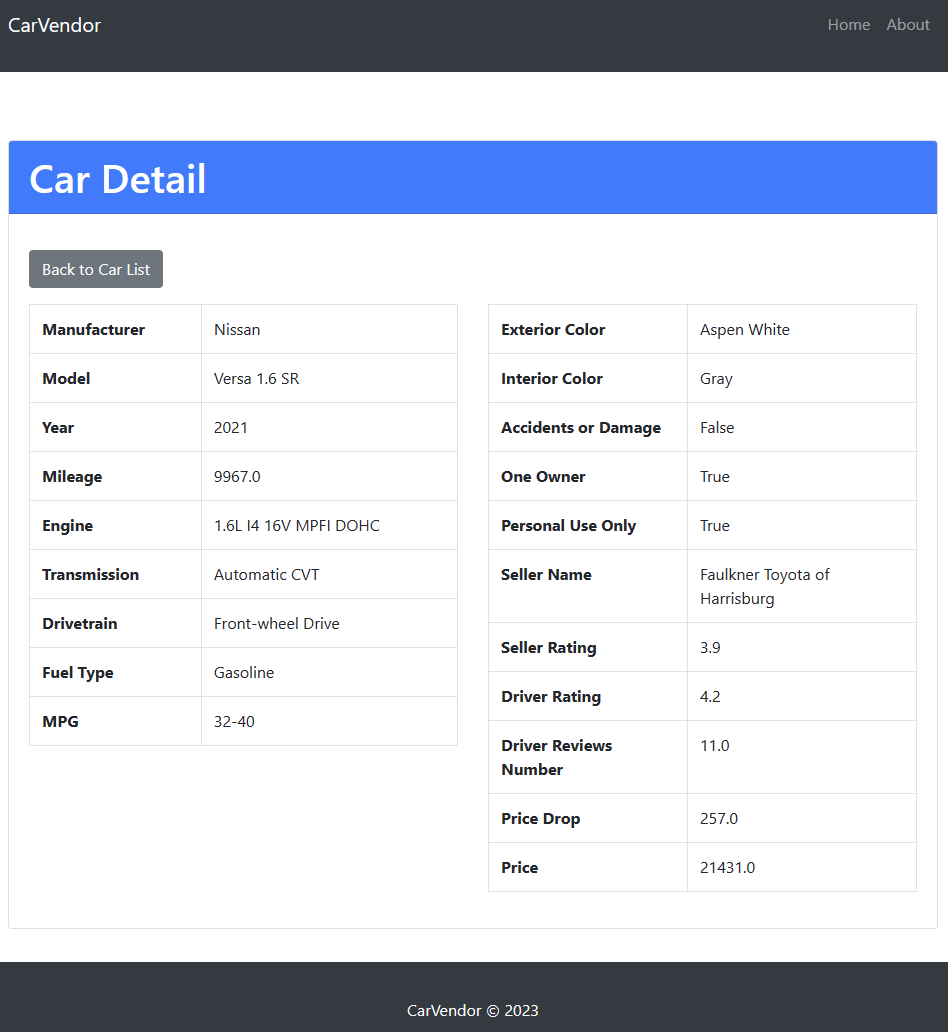
* About Page: The user can enter the about page by clicking on the about button in the top right corner of the home page. This page displays information about the CarVendor website. The user can return to the home page by clicking the home button.



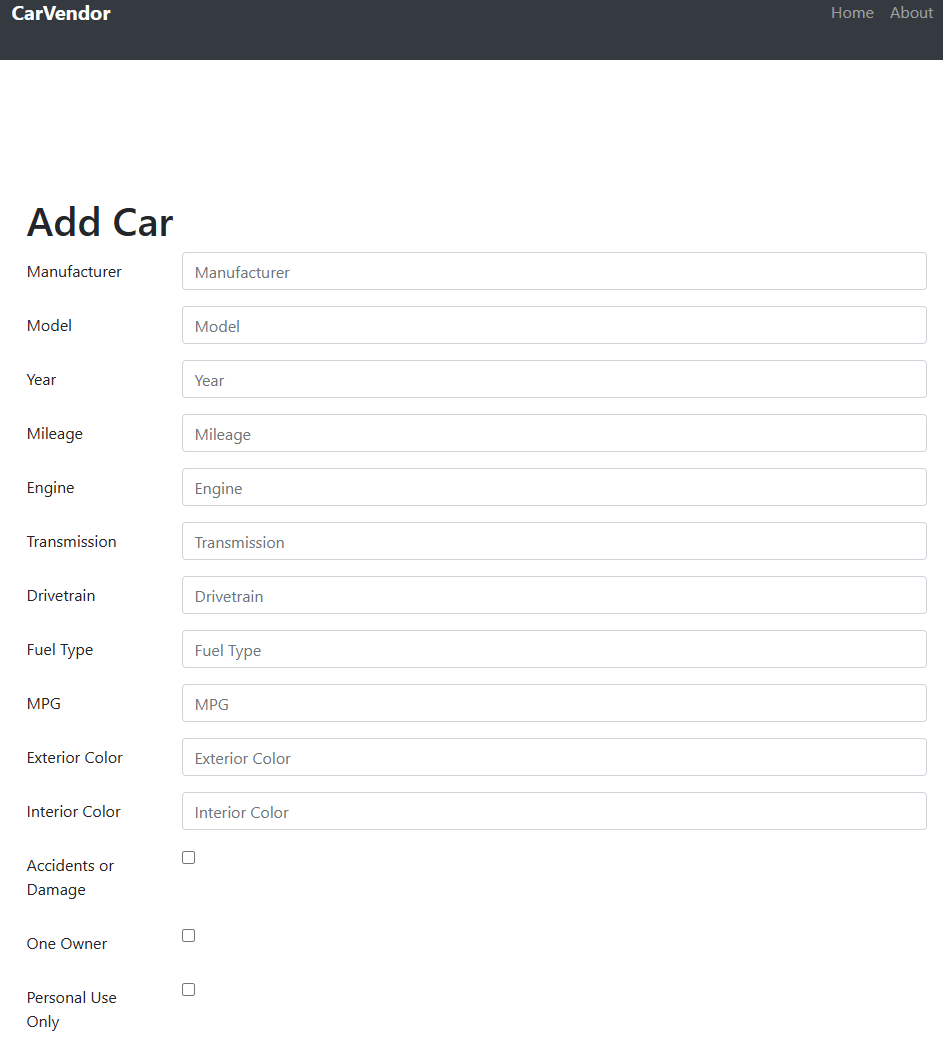
* View Cars Page: The user can enter the view cars page by clicking the view cars button on the home page. A table of available used cars is displayed on this page. The user can select filter options which will filter the data by the manufacturer or by year. There are several buttons on this page including add car, details, delete, and update.



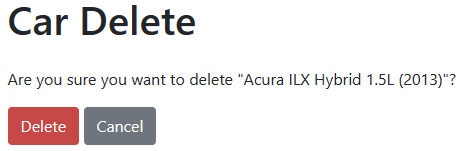
* Details Page: The user can enter the details page by clicking the details button on the view cars page. This page provides more information about the specific car for sale that the user selected including mileage, engine, transmission, fuel type, mpg, accident history, sale price, and more. The user can return to the view cars page by selecting the Back to Car List button.



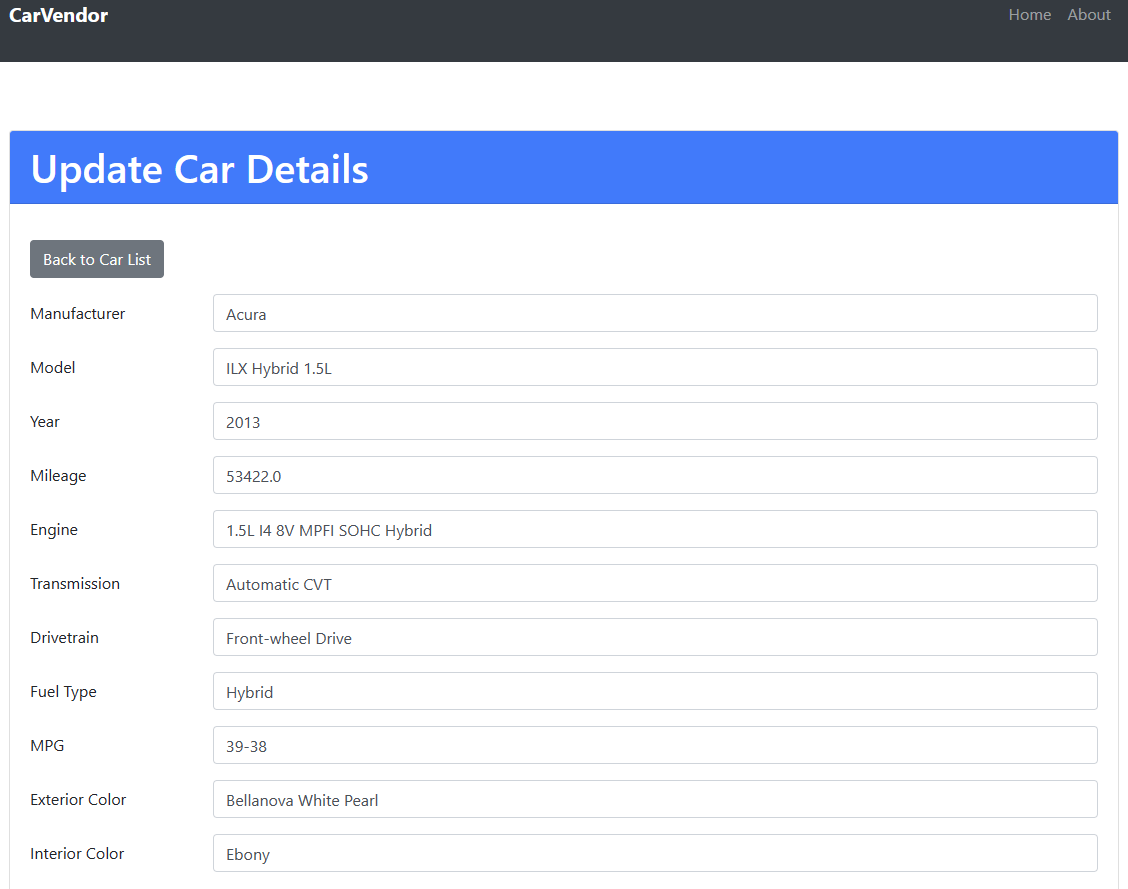
* Add Car button: The user can add a new car to the table by clicking the add car button on the view cars page. This brings the user to a page where they can enter all the details about the new car listing.



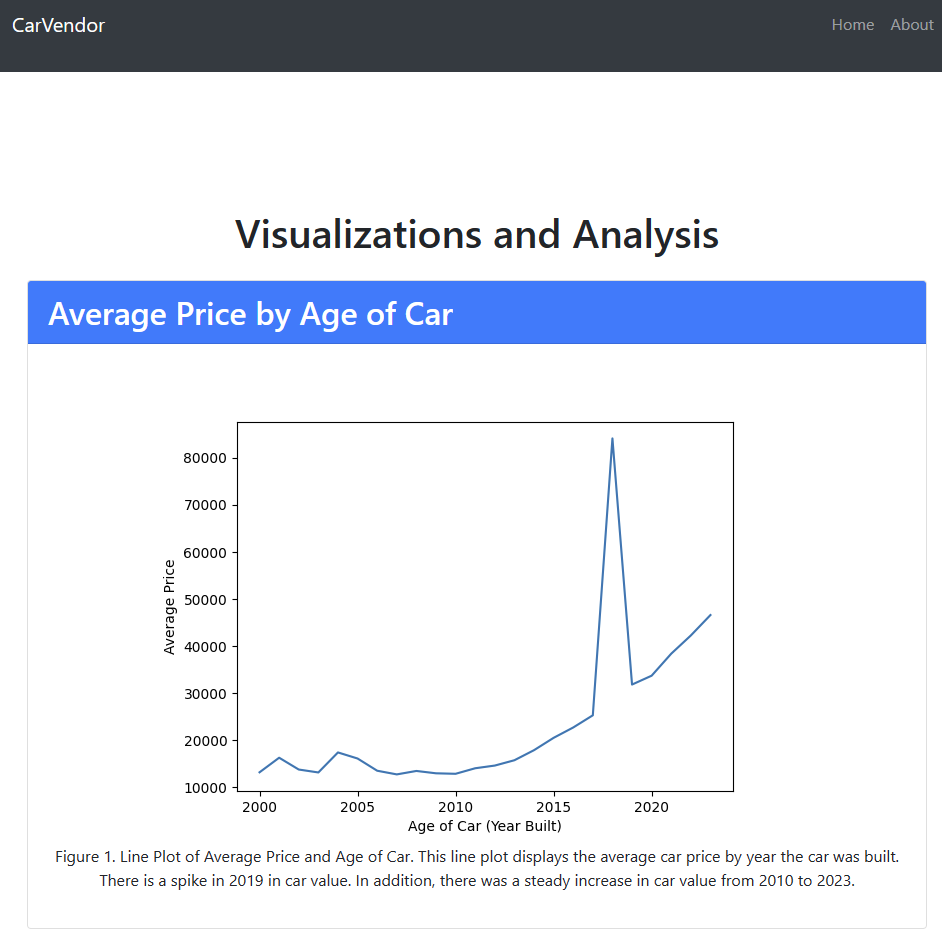
* Delete button: The user can delete a car listing by clicking the delete button on the view cars page. The user is asked to confirm if they would like to delete or cancel. If the user selects delete then the listing is removed.



* Update car button: The user can update a car listing by clicking the update button on the view cars page. This brings the user to a page where they can enter the details about the car listing that needs to be updated. There are 20 fields available to update.



* Visualization Page: The user can enter the plots page by clicking on the view plots button located on the home page. This page displays key visualizations and provides analysis on each plot. There are six visualizations and analyses provided on this page to aid the user in selecting a car. This image displays the visualization page and the first visualization.



**References (Paper):**

*SQLITE3 - DB-API 2.0 interface for SQLite databases*. Python documentation. (2023.). https://docs.python.org/3/library/sqlite3.html

About sqlite. (2023). <https://www.sqlite.org/about.html>

Django. Django Project. (2023). <https://www.djangoproject.com/>

Django Tutorial Writing your first Django app. Django Project. (2023). <https://docs.djangoproject.com/en/4.2/intro/tutorial01/>

Manage File. Django Project.(2023). <https://docs.djangoproject.com/en/4.2/ref/django-admin/>

URL File. Django Project. (2023). <https://docs.djangoproject.com/en/4.2/topics/http/urls/>

Visualization with python. Matplotlib. (2023). <https://matplotlib.org/>

**References (Website Code):**

MozDevNet. (n.d.). Django tutorial part 2: Creating a skeleton website - learn web development: MDN. Learn web development | MDN. <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/skeleton_website>

Knivets, D. (2019, October 23). *How to use django with an existing database?*. Django Adventures. https://djangoadventures.com/how-to-integrate-django-with-existing-database/

Django. Django Project. (n.d.-a). <https://docs.djangoproject.com/en/4.2/intro/tutorial01/>

SQLite group by. SQLite Tutorial. (2022, April 3). <https://www.sqlitetutorial.net/sqlite-group-by/>

SQLite inner join with examples. SQLite Tutorial. (2022b, April 3). <https://www.sqlitetutorial.net/sqlite-inner-join/>

SQLite where - filter rows in a result set. SQLite Tutorial. (2022c, April 3). <https://www.sqlitetutorial.net/sqlite-where/>

Pyplot tutorial. Pyplot tutorial - Matplotlib 3.7.2 documentation. (n.d.). <https://matplotlib.org/stable/tutorials/introductory/pyplot.html>

GeeksforGeeks. (2022, September 30). How to rotate X-axis tick label text in matplotlib?. GeeksforGeeks. <https://www.geeksforgeeks.org/how-to-rotate-x-axis-tick-label-text-in-matplotlib/#>

Seaborn.boxplot. seaborn.boxplot - seaborn 0.12.2 documentation. (n.d.). <https://seaborn.pydata.org/generated/seaborn.boxplot.html>

Hamedani, Mosh. (2021). Python Django Tutorial for Beginners. YouTube. Retrieved July 10, 2023, from <https://www.youtube.com/watch?v=rHux0gMZ3Eg&amp;t=834s>.

Schafer, Corey. (2018). YouTube. Python Django Tutorial. Retrieved July 12, 2023, from <https://www.youtube.com/watch?v=UmljXZIypDc&amp;t=10s>.

GeeksforGeeks. (2021, August 27). Django crud (create, retrieve, update, delete) function based views. GeeksforGeeks. <https://www.geeksforgeeks.org/django-crud-create-retrieve-update-delete-function-based-views/>

Representing data in Django using Matplotlib. Section. (2021). <https://www.section.io/engineering-education/representing-data-in-django-using-matplotlib/>