**Mini Project Three**

**Alain Edwards**

**11/22/13**

**Elite Imports**

**Site:** <http://lamp.cse.fau.edu/~aedwar50/MP3>

**Purpose:**

The purpose of this site was to create a simple website that access a database of cars and filter them based on the search criteria. The site is to be implemented using a combination of PHP, HTML, AJAX, & JavaScript. The website will dynamically change with the change of the search filters without the need to submit the form or refresh the page.

The targeted audience for this website is anyone in need of finding a car. Weather it be an exotic car or standard car.

**The Design:**

In terms of the graphical user interface, I decide to go with the must user-friendly design option. By having the search criteria on the left and the resultant car listings on the right.

I also tried to bring in a little finesse into the color scheme by avoiding my usual black on white design patterns. I have chosen to display a picture of a Mercedes-Benz for the background, while also insuring that it will fit every size monitor. In the event a person has a very wide monitor, they will see a black frame around the background picture. I also I have taken the liberty to set min-widths and max-widths on a few elements, to again support resizing of the browser window as well as monitor size.

You will also notice some special effects, such as fading and error handling. I have chosen to only display the resultant cars on selection of the search options, which will fade in from the top. To help explain what I’m trying to say, I have provided a few test cases at the bottom of this document. I also imported google fonts for most of headers/titles.

**Search/Filters**

The search filters allow you to search a car by specific make (brand), model, color, and select extra features like alloy wheels, navigation (GPS), and premium audio.

In order to begin a search, select one of the makes from the list. Upon selecting a make, the model selection box will dynamically populate. Select the desired model, again the color selection box will dynamically populate. Once you have selected a color listing will start to fill the right side of the screen. From here you can select some extra features to further filter the listed results. Keep in mind that you could have also selected these features from the beginning, followed by the selection of make, model, and color.

**Listing**

Listings include the specific details of the car, like the make, model, color, price, engine, transmission, etc. and a unique image for every car.

**Functionality:**

To go into a little detail about how this is done, I will go over some key aspects of the process.

Selection boxes: On change of the selection box a java script function will be called which will implement the Ajax components of that JavaScript file. This Ajax operation cannot be completed without its counterpart “command.php“. This file is in charge of querying the MySQL server for data and posting this data in an orderly fashion while using html tags.

One really cooling thing I thought I implemented was the ability for the listings to be updated dynamically on the change of any of the features. In other words if you change an extra feature parameter it shows the changes in real time.

**Time:**

The most time consuming part of building this site was the CSS, JavaScript, and MySQL manipulation. In terms of CSS, I had to make sure that everything had the proper height and width. When designing the JavaScript and MySQL side of the website, I had to make sure that I wasn’t over complicating the site and making sure I did everything in the most efficient way. Like the old famous say “there’s more then one way to skin a cat”, there is more then one way to code a site.

**Tools:**

The tools I used to complete this project were Text Wrangler, Adobe Dreamweaver CS5, Gimp, Photoshop, FileZilla and an assortment of browsers. Text Wrangler and Dreamweaver were used only as an IME. I did not use the drag and drop part of Dream weaver, the only reason I like to use Dreamweaver is because of the auto text filler. In other words it’s easier and faster to type in Dreamweaver; also you don’t have to search for things you forgot it’s all there. Gimp and Photoshop were used to create the top banner and some other logos. FileZilla was used to upload all the files to the FAU server through SSH. In the end every single tool had its unique capabilities and was all essential to the completion of this project.

**Sitemap:**

* Home: index.php

Behind the scenes:

* Command: command.php
* Misc. JavaScript: js/misc.js
* Main JavaScript: js/main.js
* Main Cascading Stylist Sheet: css/main.css

**Webpages:**

Home – displays the car searching tool and the resultant listings.

Command – handles all MySQL querying and data reporting.

Main JavaScript - used to consult command.php in order to query the MySQL server.

Misc. JavaScript – holds JQuery statements and used for special effects.

Main Cascading Stylist Sheet – holds everything to do with css.

**Browser Compatibility/Testing:**

**Google Chrome: (Fully Operational)**

**Firefox: (Fully Operational)**

**Safari: (Fully Operational)**

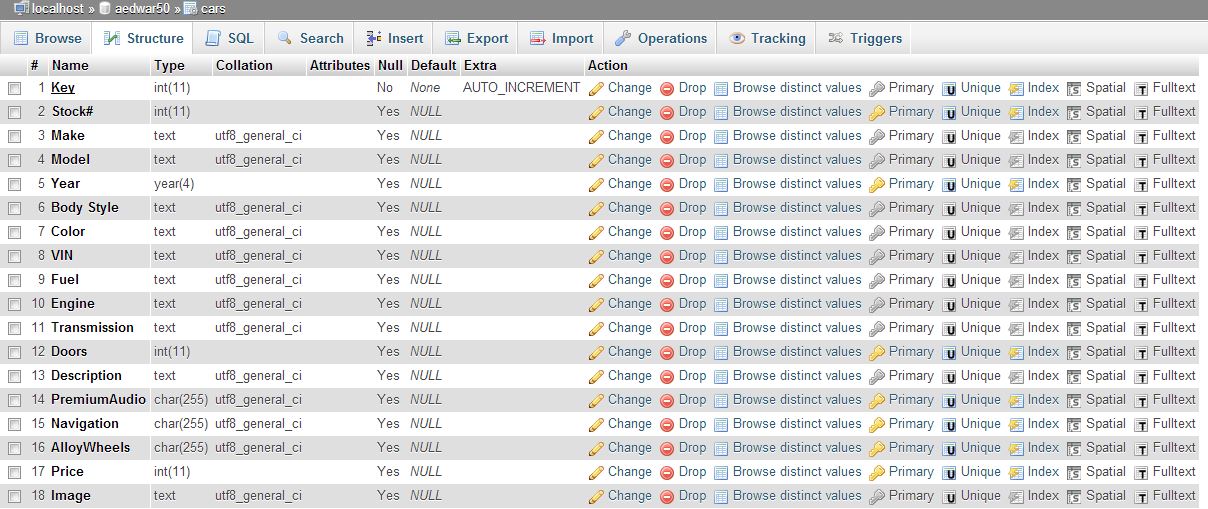
**Others (Not Tested)**

**MySQL User Database**

**SQL Injection Prevention (Fully Operational)**

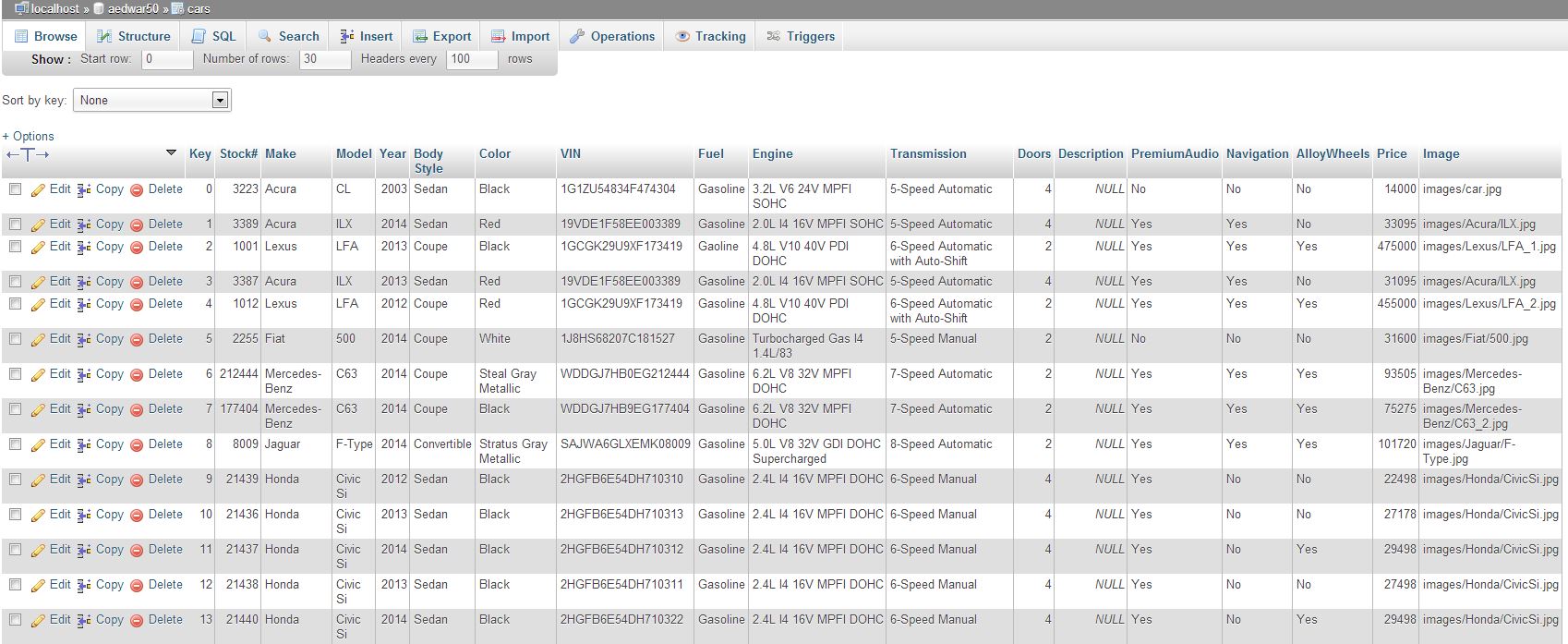
I have taken proper information sanitization steps to insure a SQL injection free zone. By checking for GET method variables, by using the isset() function, I tried ensure an error free environment; the same goes for JavaScript. I have minimalized the event of an SQL injection-taking place. This is not to say, that an injection cannot happen but this will minimize the risk, because one of the key components of MySQL injection is to cause a PHP error.

**SQL Architecture:**

****

**(Full image in same directory – Architecture.jpg)**

**SQL Database Snippet:**

****

**(Full image in same directory – Database.jpg)**

**Mobile Website: (N/A)**

Unfortunately, I was not able to put together a mobile compatible site before the dead line. Seeing that this was not in mini projects threes criteria, my attempt to get extra credit was cut short.

**Test Cases:**

**Case1**

1. Select Acura
2. Select ILX
3. Select Red

This will display all the Acura ILX’s that are red in stock.

**Case 2**

1. Complete Case 1
2. Under Options: Select Alloy Wheels

If you notice you will receive an error that states:

*“I'm sorry but there are no cars matching your search criteria at this time. Please try again later or try changing the options under search “*

This is because we currently do not have a red Acura ILX with alloy wheels.

**Case 3**

1. Under Make: Select Buick

This will also print out an error because we do not currently have a Buick in stock, but this time you will notice that the error was presented in a JavaScript alert box. The reason for this is to cover the scenario that a customer visits the site and right of the bat select a car make which we do not currently care and the listing results have not revealed it’s self because there were no cars.

If you would like to continue to test the website, please consult the SQL Database Snippet for what is currently in stock.

**Conclusion:**

The overall project was a great experience. I learnt a great deal of information from this project. Some of this information includes things like Ajax and how to remove appended selection options in JQuery. This new experience helped me to fine-tune some of my skills as a web developer.