# 

# **Claire’s Cars Website**

## Video Demo:

<https://northampton.mediaspace.kaltura.com/media/Claire%27s+Cars+-+2018+Apr+21+10A18A52/1_cs0cc1pl>

Content

[**Claire’s Cars Website** 1](#_Toc512184625)

[Video Demo: 1](#_Toc512184626)

[Requested changes checklist: 3](#_Toc512184627)

[Analysing existing code: 3](#_Toc512184628)

[Testing 9](#_Toc512184629)

[Bugs and Weaknesses 10](#_Toc512184630)

[Bugs 10](#_Toc512184631)

[Weaknesses 11](#_Toc512184632)

[Evaluation 11](#_Toc512184633)

[Before 11](#_Toc512184634)

[After 12](#_Toc512184635)

[Appendices: 12](#_Toc512184636)

[References 15](#_Toc512184637)

# Requested changes checklist:

|  |  |  |
| --- | --- | --- |
| No: | Requested Change: | Completed: |
| 1 | Change opening hours for Sunday to closed |  |
| 2 | Add Claire’s Careers page and have it say Claire’s Cars currently has no job opportunities available, but keep checking as new positions become available regularly! |  |
| 3 | Fix adding new manufacturer not showing issue |  |
| 4 | Can you make it so I can “Archive” cars that are listed on the website? |  |
| 5 | Store new and old prices for cars |  |
| 6 | Allow admin accounts to be created |  |
| 7 | Add mileage, engine type fields |  |
| 8 | Allow at least 4 pictures to be uploaded for a car |  |
| 9 | Allow articles to be posted with optional images and show when they were posted and who by. |  |
| 10 | Create an enquiry form on the contact page. |  |

# Analysing existing code:

1. Repetitive code:

Issue: The code shown in the image is copied on all of the pages. This is repeating code throughout the site. The issue with this is that coding the site will take twice as long simply because you must add the same code throughout the website. Examples of this is the heading section of each page.

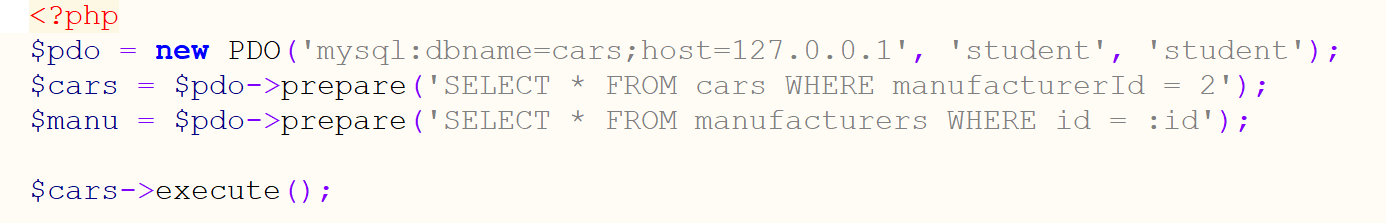
Solution: A solution to this issue is to create a standard layout page which will be one page showing all the code that is repeated on each page. This means all you need to do is change the content and post it into the layout. This is done by replacing the main content on the layout page with a variable. The variable will store the content of the page.



Layout page showing solution:

As you can see from the screenshot there is now two variables that have replaced the title and the main content of the page. This layout will be your website and as you navigate through the site each page will pass it’s own title and it’s own content.

1. Constantly creating the PDO:



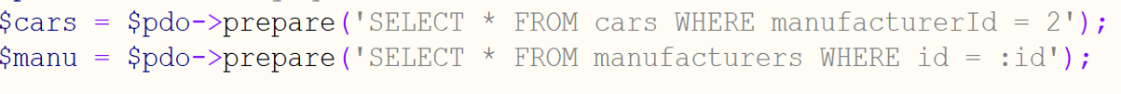
Issue: The database is being created on every page. This is using more repetitive and isn’t required taking up time that could be used elsewhere.

Solution: The pdo variable can be moved into it’s own file which can then be created just the once.



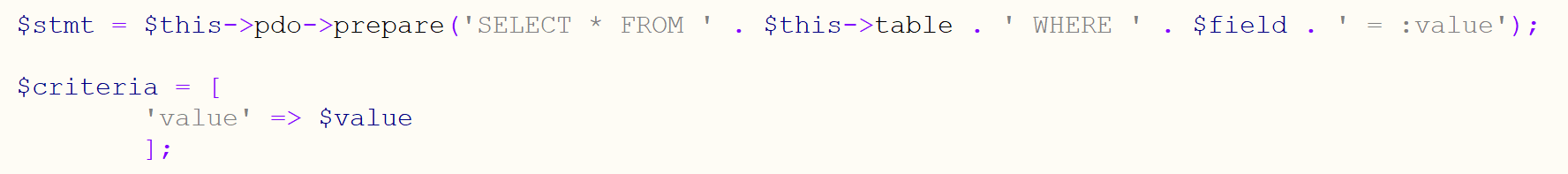
This line is now in it’s own file which can be required on the index page and passed when used to the database.

1. Specific Database queries:



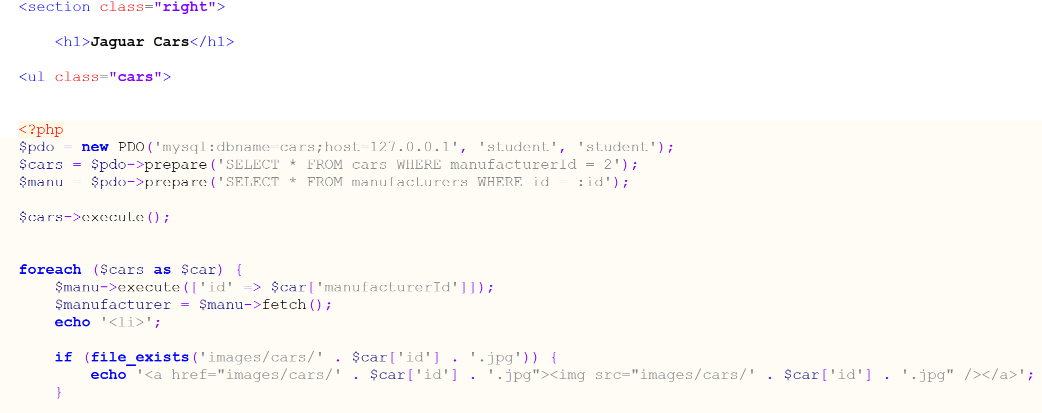
Issue: The above queries are hard coded so that it can only be used on that page. The only time you can use these queries again throughout the website is when the cars table is being searched through and when you want the restriction to be search for manufacturerId that equals 2. This restricts usability.

Solution: in order to make these queries reusable then you can prepare the query and put place holders for the table column and value filter by. This will mean all you have to do is call the function to run the query and pass it the table you want to search through and the condition. This will save on so much repetitive code and it will be easy to find out what the query does.



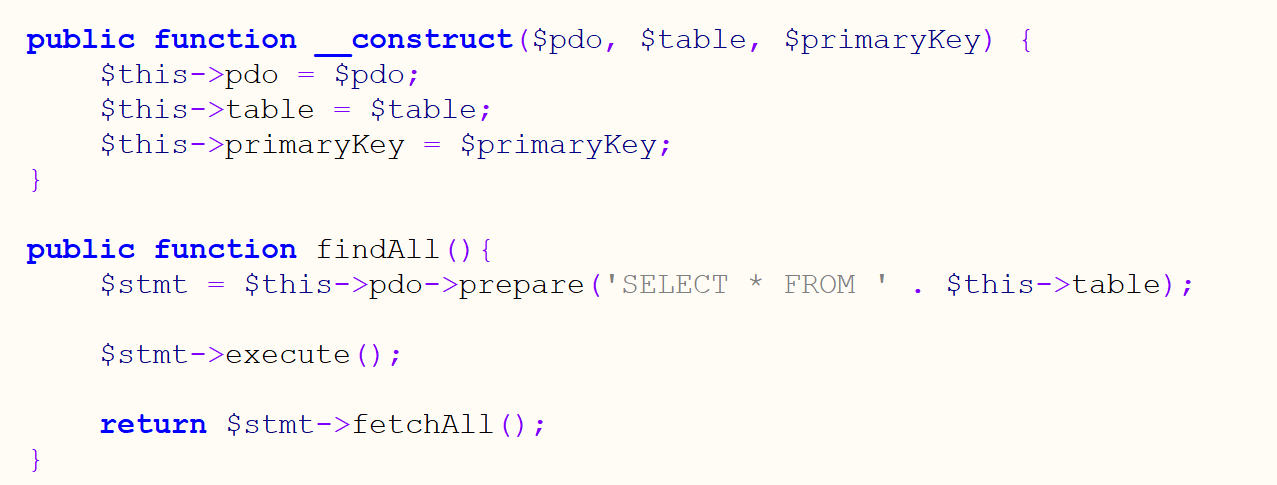
The above code displays the solution in code as you can see there is no hard coded values in the query so it can be used whenever all that is required is for the variables to be passed through.

1. Database code is mixed with php code:

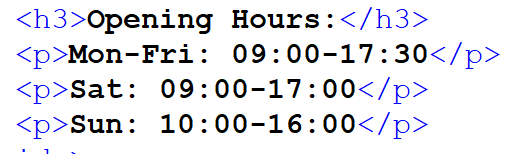


Issue: This makes locating database queries a bit harder as you have to search through each page to find the query and then to make changes if needed.

Solution: An easier way of doing this is to put all the queries in one file that can then be required and used when needed. This will put all requests to your database in one place which is easier to search through and will save time. By taking out the database code from each page it makes the page easier to read and to change.

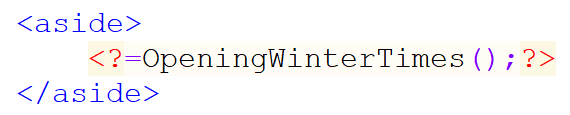


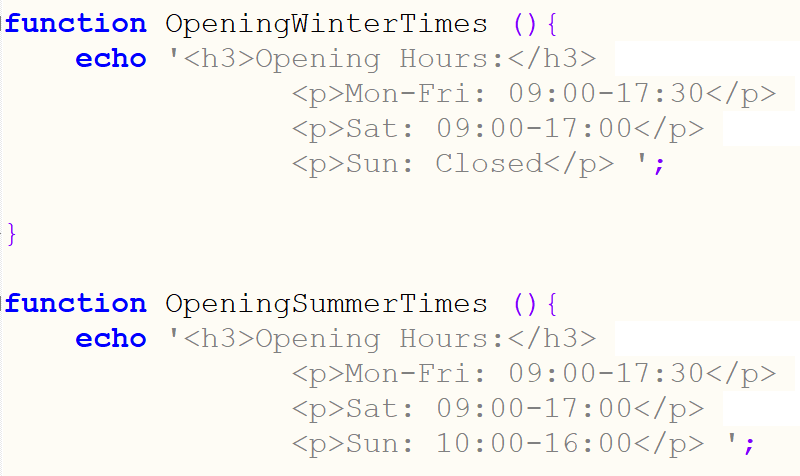
1. Future changes to the website:



Issue: If you wanted to change a time or day on the opening hours its not just changing one thing you have to go into every page and individually change it.

Solution: You can replace this section of the code with a function and put the times in the function. Then you only have to change the code in one place which is in the function. Another way is to have different functions and then all you need to do is call the function you want to change it to. The downside to this is that there will be unused code as not all the functions will be called upon.





As you can see there is two functions which can be called and are easily interchangeable.

1. Admin Access:

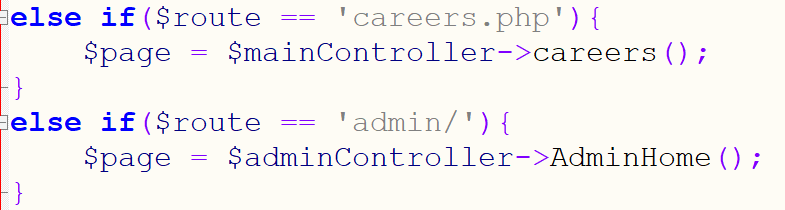
Issue: The problem with accessing the admin side of the site is the face its set up as an entirely different website. You have to enter the url for the admin index page. There is no smooth transition from the customer facing site to the admin site.

Solution: You can easily fix this by implementing the admin pages into the main website. To enter this you can set up a button that calls for the admin home page. This can then have a log in section which makes the page secure and inaccessible for customers. This can save time when making changes to the layout of the website. Instead of having to do the changes on both the customer facing side and the admin side they can share the same layout page and again the changes will only need to be made in one place.

1. Multiple index pages:

Issue: Having multiple indexes is just creating more code which doesn’t need to be there.

Solution: The site can just come from one index page. This works by searching for the url and if it matches then the index page will tell the site to run a certain task this will then change the contents of the page as deemed fit. Below is an example of how the index page can tell between the customer facing side of the website and the admin side of the site.



This is one way the index page can differentiate between the admin site and customers there are other ways where the url is stored and the relevant class is called from that url.

1. Files stored in Public:

Issue: All the files for the site are placed in the public folder. This means that people can access each file by calling the file name eg. Cars.je/AdminPage this can then put your website at risk of being hacked.

Solution: To solve this only pages the user has to access should be placed in the public folder. So in this case the only files in the public should be the css and index files and the images folder. This will then stop users being able to navigate to other pages without going through the index.

1. About page file type needed changing:

Issue: The file type for the about page was set to html. In this file type you can’t implement any php code so it wouldn’t be able to require pages which were needed for the layout.

Solution: change the file type to php which followed the structure for the other templates and allowed the contents to be passed into the layout page.

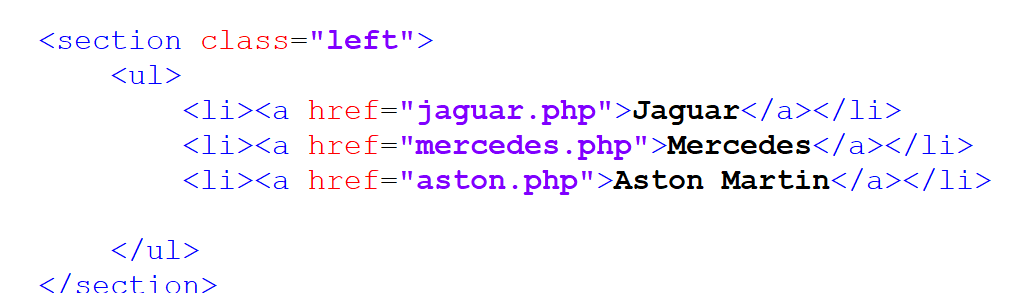
1. Individual pages for manufacturers:

Issue: The website had it’s own individual pages for each manufacturer This means that every time you wanted to add a new manufacturer to the site you would have to create another webpage to display the cars related to them.

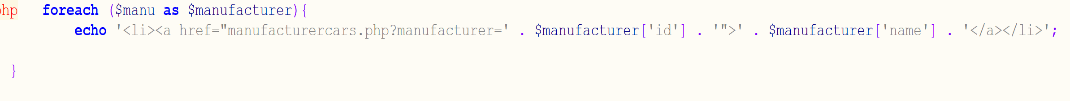
Solution: You can just create one page which pulls through the contents depending on the manufacturer. You can do this by adding the manufacturer’s id into the url and getting it from the url to load the correct page. When you add a manufacturer the list will retrieve them from the database and do the same for them. This then makes it really easy to add manufacturers to the website.

1. Static pages:

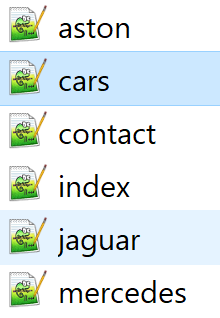
Issue: The issue with static pages and not using dynamic pages is when adding another item in this instance it would be a manufacturer you would have to create an entire new page for them and add the link into the navigation section. The navigation would need to be modified on every page which uses it. This is creating needless work for yourself.



Solution:



The above image shows a way of replacing the navigation so it loops through all the manufacturers in the database and print out a link for them. The url would also store the manufacturers id which helps upload the right page for the right manufacturer. So the new site now uses one page which loades the relevant information depending on the manufacturers id that is being pulled through by using the $\_GET variable. Using this will mean that the pages in the image can be replaced by one file.



The files aston, jaguar and Mercedes can be removed with the solution implemented.

The files are replaced by manufacturerCars file instead this saves a lot of time when implementing further features or changes.

# Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No: | Test: | Expected result: | Actual Result: | Screenshot: |
| 1 | Testing Admin Log in using correct details | User should be able to log in | Logged in as expected | 1 |
| 2 | Testing admin log in using incorrect details | User can’t log in | Worked as expected | 2 |
| 3 | Test log out works | Should log user out. | Worked as expected | 3 |
| 4 | Test left nav bar when admin has been selected. | Logged out should ask user to log in. | Worked as expected |  |
| 5 | Test home page. | Should pull up all articles | Worked as expected | 4 |
| 6 | Test showroom page | Should pull through the cars with details | Worked as expected | 5 |
| 7 | Test contact us page | Should pull through an enquiries form | Worked as expected | 6 |
| 8 | Test showrooms manufacturer navigation | Should pull through all cars related to a certain manufacturer | Worked as expected | 7 |
| 9 | Test admin navigation bar manufacturers | Should link to the manufacturers page | Worked as expected | 8 |
| 10 | Test edit manufacturer section | Should let you edit a manufacturers name | Worked as expected | 9 |
| 11 | Test deleting a manufacturer | Should remove manufacturer | Worked as expected |  |
| 12 | Test admin navigation bar cars | Should display a list of cars | Worked as expected | 10 |
| 13 | Test add cars | Should show form to add a new car | Worked as expected | 11 |
| 14 | Test edit car | Should show form filled out with car to be edited | Worked as expected | 12 |
| 15 | Test archive car button | Should archive the car and remove it from the cars list | Worked as expected |  |
| 16 | Archived car navigation button | Should bring up a list of archived cars | Worked as expected | 13 |
| 17 | Delete an archived car | Should remove the car from both the archive and the cars list | Worked as expected |  |
| 18 | Restore archive car | Should remove car from archive list and place it back in the cars list | Worked as expected |  |
| 19 | Admins navigation button with master account | Should bring up the form to create or delete admin | Worked as expected | 14 |
| 20 | Admins nav button without master account | Should say display can’t use must be a master account | Works as expected |  |
| 21 | Enquiries nav bar button | Should bring up list of incomplete enquiries | Works as expected | 15 |
| 22 | Enquiries complete button | Should mark the enquiry as complete and remove it from the list | Works as expected |  |
| 23 | Create articles nav bar button | Should bring up a form to create articles with | Works as expected | 16 |

# Bugs and Weaknesses

## Bugs

|  |  |  |
| --- | --- | --- |
| Error No: | Error | Solution |
| 1 | When logged out and selecting the admins link in the navigation bar brings up empty page | Need to implement a if $\_Session not logged in then load log in page. |
| 2 | When logged out clicking on the navigation bar when selecting admin. If you don’t use the first log in and use one of the other pages it will bring up an error page. (Shown in demo) | Need to link it to the same page so the index page calls the same function in the controller. At the moment they don’t have a log in implemented into the function. |

## Weaknesses

|  |  |  |
| --- | --- | --- |
| Weakness no: | Weakness | Solution |
| 1 | The sub navigation bars in the main content area is coded in the template which creates duplicated code. | This could be solved by either creating a variable to pass into the main content section. You can also use a require to call a file with the sub nav bar code this will mean there is only once place to change the nav bar if needed. |
| 2 | Index has one long If statement | Would be better to pass the pages with a get variable in the url and use single point entry to read the get variable and run the functions required. |
| 3 | There is no where to edit articles | I haven’t created a feature to edit articles you can only create and delete them. |
| 4 | Multiple templates just output a completion message on tasks | Can create one standard template that has a variable passed into it that has the output message set in the function. (Similar to setting the title for the page) |
| 5 | Templates implement some css styling in the form or table. | Should have a different class for the look of the form or table. Then you can call which class you want to use when styling the form or table. |
| 6 | Database class contain all the functions for querying the database. This can get very big and very confusing | Should split it up into multiple files and classes depending on what the function is asking from the database. |
| 7 | The log in for the admin sub nav bar is repeated on all of the templates which is a lot of repeated code | This should be taken out and put into a separate file that can be required on each of the pages this then can save time when making changes to it. |

# Evaluation

In this section I will be summarising and evaluating the site before the changes and after the changes.

## Before

The customer facing side of the website was working and all the details were viewable and the site had a good layout to it. It was easy to navigate through to find what you were looking for.

There were many down sides to the site as stated previously it was very hard and time consuming to try and implement changes to the site as you would have to make the same change to multiple pages. Another downside to the site was that you had to directly type the url in to get the admin index page up otherwise you wouldn’t be able to work through to the admin side of the site through the customer facing side of the website.

I thought a big issue with the site is that it’s very restricted as to what it can display and the functionality of the site for example if I wanted to add a new manufacturer I could but a customer seeing the website won’t be able to view any cars for the manufacturer as it wouldn’t exist as a webpage.

## After

The site is now set up so it is really easy to add new pages and implement new features as and when. The site directly links to the admin side so an admin user can flick between the two parts of the site. This will now save a lot of time for admin users to add details and complete tasks and too view the changes as they can quickly switch with a click of the button instead of changing the URL.

The website is clear when looking at the code to make additional changes in the future it’s nicely laid out. The directory for the site is split up so you can easily find the files related to the change you want to make. For example make a text change to the content you can just find the page in the templates file and change it.

The website will now take a quarter of the amount of time it did previously to add a new feature or to change because it focuses on only having to make the change once on the site.

The code for this can still be taken further for example sub navigation bars can be implemented by using require’s. This will reduce repetitive code even more. Another change could be the index page at the moment it works by having a big if statement. The site could work from having a single entry point and use $\_GET variables in order to find what page has been called. Once this has been called the content would get populated by with functions. This reduces the amount of code in the index file and doesn’t require a big If statement.

# Appendices:

|  |  |  |
| --- | --- | --- |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |
| 13 |  |  |
| 14 |  |  |
| 15 |  |  |
| 16 |  |  |

# References

Online, Website link : <http://php.net/manual/en/reserved.variables.files.php>, used for: This was used to learn about using the $\_File variable in regards to images.

Online, website link: <https://stackoverflow.com/questions/4422210/how-to-add-simple-image-upload-to-a-form>, used for: This was used for adding an option to load an image in a form.