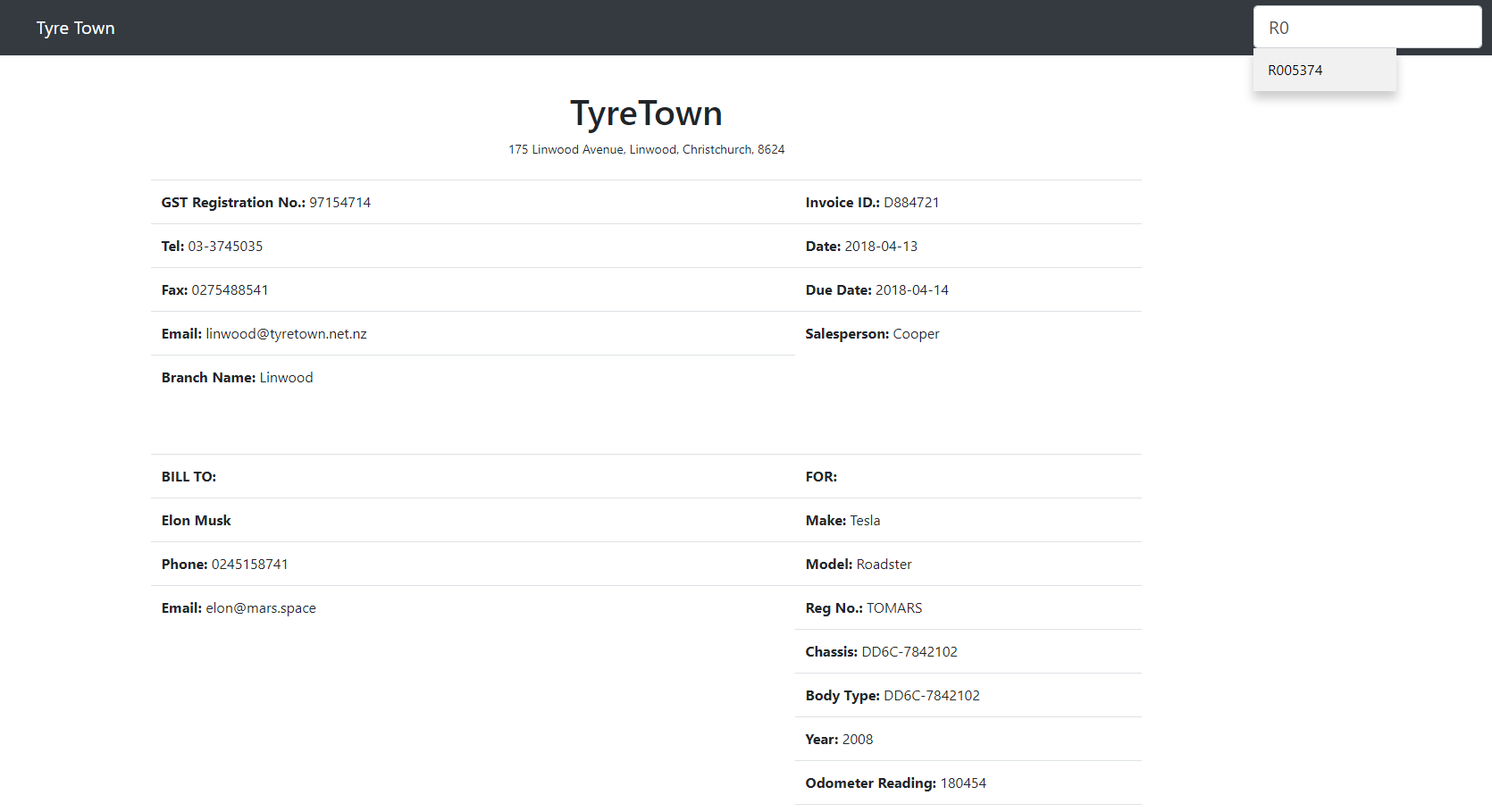
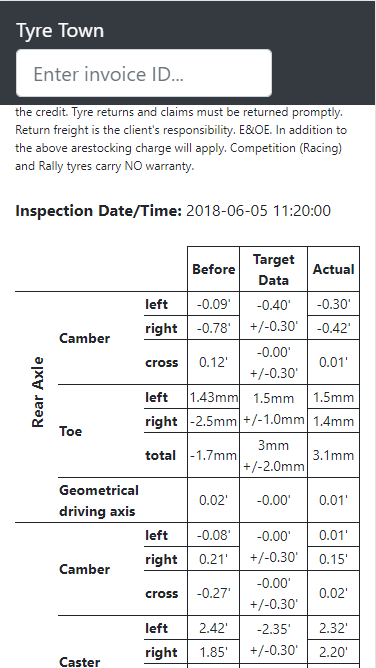
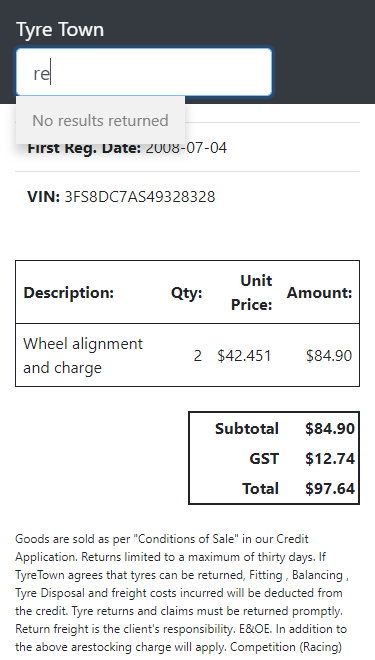
# Group B TyreTown Prototype Overview





The aim of this project was to develop a Wheel Alignment Invoice report for a fictional company TyreTown in a dynamic HTML webpage format so that the information can be accessed online.  During the project, we used PHPStorm, Xampp, Notepad++, GitHub and Google Chrome developer tools to develop the prototype.

The structure of the page is written in HTML, and we used the Bootstrap CSS library to make sure we had a consistent and modern theme throughout our page, as well as using a small amount of our own CSS (NOTE: we decided against using a Bootstrap CSS table for the wheel alignment data table, as it wasn’t as flexible as we desired for that particular table. We do use a Bootstrap CSS table for the tables at the top of the page). We have a search bar in the top navigation bar, which can be used to search for an invoice. When any characters are entered in here, any Invoice IDs which match include the search string populate a drop down list of buttons. Clicking on one of these buttons populates the page with the information about that invoice and wheel alignment from the database.  To achieve this on the client side, we use Javascript (and the JQuery library) to send asynchronous AJAX GET requests to our server side PHP files. When a valid JSON response is returned, we insert the information into the correct places on the page using JQuery, which selects our page elements by their ID.

As our top priority is to develop a mobile friendly prototype of web application, the CSS style is also written with this in mind. The desktop version of the application will only show when the screen size is at least 660px in width. We achieved this by relying on the developer tools of Google Chrome to adjust accordingly using Galaxy S5 as our Android standard (360X640) and IPhone 6/7/8 for our Apple standard (375X667). Instead of having a different view for each smartphone, both smartphones would have the same view as it is more convenient for either smartphone user. Going into details, to make the application look neater, we made some changes to the styles from the Bootstrap CSS library to allow for space management through text and padding. In the mobile view, we modified the table to allow for the easiest viewing for the users. For example, one of our changes to help with the size optimization was rotating the labels vertically and fixing then in place in the table with a rowspan, and making some of the table text smaller.

When the page loads, we automatically fetch all of the target information for the report using a JQuery AJAX request in $(document).ready() to the server side file target\_information.php. This target information does not change between requests, and it saves requesting this from the database later (future requests can be done in only one database access), so it makes sense to load this information only once and at the very beginning when the page loads.

One of the calculations that were requested was the calculation of GST at the value of 15% of the pricing for the services. We achieve this by designating GST = 0.15 (var GST\_RATE=0.15) in the script.js as a constant global variable. The calculation for the total amount is acquired by [subtotal = Unitprice\* Quantity] then [subtotal \* GST rate = GST amount] and [total=subtotal + GST amount].

On the server side, we are using PHP which connects to a MySQL database using a *mysqli* connection in PHP. We make sure to sanitize any input in the PHP script which has been sent as part of the request, and use this to build appropriate queries which return information from the database. We then process this information and put it into the correct format, and then encode it as JSON, which we then echo as the response. We decided to remove NULL values and replace them with hyphens, as otherwise we end up with empty boxes in the tables which could look like an error has occurred. Our server side scripts fail gracefully with invalid requests, and return appropriate headers with any relevant error information, which makes any issues easy to debug, and protects against invalid information being parsed or put into the webpage.

# Extra Feature

We decided to combine a search feature for invoices with the ability to select different invoices through AJAX requests as our extra feature, as it is a very practical and useful feature and would be imperative as the database of invoices grew larger. The search bar is along the top in the navigation bar, and the search term is matched with any part of the invoice ID. From here, a dropdown list of matching invoice IDs appears, which can be clicked on to bring up the information to the page corresponding to that invoice.

The search bar is implemented as an input box using a few Bootstrap CSS classes to keep a consistent theme. We check for the ‘oninput’ event, which sends an AJAX GET request to the database.php file with method=search and search\_term equal to the term in the search box. The database.php file queries the MySQL database using *LIKE %$term%* in the WHERE clause. We use the sanitize input function from tutorials to protect from SQL injection. The database.php file returns a JSON object of matching invoice IDs from the database. From here, the javascript function which handles a successful request adds these IDs to buttons in a dropdown list appearing below the search bar using jquery. The buttons change colour subtly when hovered over to indicate that they are ‘clickable’. Each button has an onclick method which sends an AJAX request to the database requesting all information about that invoice, which then populates the page using jquery with the invoice information. If no results are returned, we add a “No results returned” label, which does not change colour when hovered, and has the text greyed out, to show that it is just a message and not a button.

Clicking outside of the dropdown list on the page hides the dropdown list. Selecting an invoice from the dropdown list then removes the search term from the input box and hides the dropdown list, and then populates the page once the information is returned from the database (once the asynchronous request returns with a successful response). These changes make it more intuitive, as it is how many other websites handle similar events.

To fetch the information about a specific invoice from the database, we send an AJAX GET request to database.php with method=retreive\_invoice and id equal to the matching ID. In the database.php file, we again sanitize the input and access the database, using a few JOIN statements in the query to pull all of the information (about the owner, vehicle, branch, wheel alignment and invoice) in a single database access. We then decided to replace any NULL values with a hyphen (similarly to how this is done when requesting the target information) so that the user isn’t faced with unexplained empty boxes which could look like an error.

The database.php file fails gracefully with invalid requests, and with any errors occurring while accessing the database, and sends back appropriate headers with any relevant error information.

# Teamwork Description and Reflection

Team members first made contact via student email, initiated on Saturday 15th of September. It was during this initial contact phase that the members discussed the time and location of the first meeting. Team members gathered for the first time on Thursday 20th of September to discuss plans for the project. Post-meeting, a Facebook group was established so that communication can be performed via desktop and mobile device; improving the project’s overall time efficiency. The group actively communicated the progress, tasks and priorities using this medium. The team decided to regularly every Friday 11 am during laboratory time.

Tasks were separated between members based on each person’s level of programming knowledge. Most of the work was done in laboratory meetings, with some research and bug fixing done at home and committed to the repository.

The code was shared through a web-based service called GitHub. File sharing was simplified with the use of the GitHub Desktop application, which enabled us to make changes to the code using our own home devices. This meant that different tasks could be completed simultaneously as each member could commit a change to master, while others can fetch the changes onto their device.

Communication was adequate for the size of the project. Most team members were responsive when feedback was required and the level of participation regarding the coding part of the project was satisfactory, considering the programming knowledge of each member. Regular set days for the laboratory meetings contributed to the success of the project; allocating the meetings in existing lab classes enabled tutors to provide feedback about ideas that the team had, and hints that may assist in the function of the web-page.

Despite the web-page's adequate functionality, features were left out due to the time constraint. With more time, there were potentially room for us to integrate more extra features to the webpage, such as highlighting in green/red any values in the wheel alignment data that were within/not within the target value range. The team was also unable to have every member present at any given time, which had impacted the task allocation for each member, and meant that we had meetings that required a member who knew about GitHub to explain how the web service and application works in two separate instances, eating up the time we could have potentially spent debugging the files and adding extra features.

In the future, communicating about time availability among members could dramatically improve the team’s performance, as well as clearly defining each person’s role and what work they were expected to complete throughout the project. Additionally, using a log file to record major progress and updates each week stored in the repository will help build a more comprehensive technical report and would better allow us to monitor our progress. Moreover, having a log will show specific instances where a task was difficult to complete, and where time spent working on a task could be better improved for the next project by pinpointing individual strengths and weaknesses of each member. This will result in a better task allocation for the next project we may be assigned with.

Overall, the teamwork for the project was not the best it could have been, considering members’ ability and availability. It was, however, adequate to complete the primary task we were given.