INFO263 – Auckland Transport Development Project

Louis Creed, Lewis Garton and Jos van Olst

Website functionality

* Client-side
  + An event listener is attached to the drop-down route selection box. The listener gets the currently selected route and runs the server-side functionality detailed below.
  + Within the obtain vehicle API information the code interacts with the Google Maps API to add vehicle markers and relevant vehicle information.
  + setInterval, a Javascript function, is set to update the currently selected route with a new AJAX call every 30 seconds.
  + The Google Maps fitBounds function resizes the map to the new markers that have been layered onto the map onto one screen.
* Server-side
  + Queries to the akl\_transport database are achieved using PHP’s MYSQL functionality. When PHP connects to the database it queries the routes table for each distinct ‘route\_short\_name’ and collects the appropriate route\_ids in an array. PHP then iterates through each ‘route\_id’ and obtains the trip\_ids relevant to each route\_id.
  + The API Call function is called to return the real time vehicle locations of the relevant trip\_ids.
  + The vehicle\_locations are returned in JSON format. The JSON format function pulls the relevant information from the Auckland Transport API.

Decision-making

We chose to query the akl\_transport database by route\_short\_name as we thought users would more likely think of the name of the bus route that way rather than route\_long\_name.

We chose to implement JSON as it was the most convenient format to pass on the vehicle location data to the Google Maps API and meant writing less code. JSON was embedded into Javascript.

Teamwork

Communication

The bulk of the group communication was done over a Facebook messaging group. Relevant material, such as the Google Maps API resources, the 263 forum and StackOverflow posts were linked which allowed us to research and solve the majority of our problems remotely. We met in person occasionally to make sure the project was moving forward, and when we encountered problems that could not be solved quickly.

We used a shared GitHub repository to manage the code. We also made some use of screenshots of proposed code sent via the Facebook group when someone had found a solution.

Task separation

Lewis and Jos completed most of the PHP portion of the project to set up the Auckland Transport API query. Jos and Louis completed the AJAX functionality and completed some of the client side functionality, identified some of the client-side functions that needs to be implemented to polish the Google Map and completed the documentation.

Reflection

What worked

Meeting in person was effective for several reasons. While we communicated over Facebook in the large part, meeting in the same place allowed us to efficiently work through problems when they arose and ensured there were no major problems with the progress of the project. It also allowed us to explain and learn from each others ideas or proposed solutions.

GitHub was an effective tool for managing the project and allowed us to incrementally complete the requirements of the project without coding parts that had already been completed.

What didn’t

In hindsight, we would have avoided a lot of frustration if we had made better use of the opportunity to research, plan and design the website functionality before we began writing any code. We found that when we had started working on the project as a group, the complexity of the code, and the fact there were multiple people working on the project made it more difficult than it should have been to get an accurate view on the project requirements. It became harder to identify what stage of the project we had reached, and once we encountered problems we had no template or blueprint to fall back on, or make necessary changes to.

We could have also made more use of communication. There were several occasions when communication broke down completely for several days, or where people had different understandings as to the state of the project and what was required. Being able to fix those problems sooner would have enabled the group to be more productive.

Time could have also been better managed along the course of the project. The presence of bugs was underestimated and having enough to time to make necessary changes to the code would have improved the quality of the final prototype.

Improvements

The vehicle information could be improved to show useful information from the point of view of a transport user.

There is a bug present in the prototype where some of the markers are not removed from the map when they are no longer relevant. This occurs only when another separate route has been loaded after an existing route is already there.

The prototype hasn’t been styled to facilitate the use of the application. Ensuring the elements on the page are appropriately set would have improved usability.

We also ran out of time to sanitise the route\_short\_name input strings. This leaves the application database compromised if an AJAX request was run manually by an attacker.

Final Reflections

Time management was a crucial determinant of the final output in the web design and development process. The better time was managed along the course of the project, the more optimal the final prototype would have been.