INFO263 Course Project

Auckland Transport Vehicle Tracking Web Application

Project Overview

So far in INFO263, we focused on developing client/server applications which involve only two parties or two computers, to every data request. The first party is a web browser, the client application, running on your computer; the second party is a web server running on another computer connected to the network. In our case, the Apache web server is usually running on the same machine alongside a MySQL database server. In this simple scheme, the server sends the data in the form of an HTML document to the client in response to the client's request. However, many modern web applications include more than two parties to each transaction. The client/server part with the web server sending HTML pages to the client remains almost the same, except that the web server may need to obtain data from a third-party server.

What is an API?

To put it simply, an API is a means of communicating with a server or other online data service. For example, one can send a request to Google to request the search results for "API", which in a sense this is using an API, the Google search API. The only difference being that traditionally APIs give the response back in a format that's suitable for machine processing/parsing. The post below has a great explanation:

https://medium.freecodecamp.org/what-is-an-api-in-english-please-b880a3214a82

If videos are more your thing, then the following video is highly recommended for your viewing pleasure:

https://www.youtube.com/watch?v=s7wmiS2mSXY

Third Party APIs

There are countless servers that provide access to all sorts of data services which can be integrated into web applications. While the web browsers usually require data in HTML format, the third party servers often provide data in formats other than HTML. These formats may include XML and JSON. It is said that such servers provide something called an Application Programming Interface (API). Our web server may request data in one of these formats from the API server, process it and integrate it into the web page that will be returned to our client (web browser). Similarly, our web browser may connect to other computers to request data which will be integrated into the web page displayed to us.

In this project for your INFO263 course, you are required to use two web APIs: Google Maps and Auckland Transport Developer APIs. The project involves a group effort on integrating those two API into a dynamic web page. You are required to demonstrate your ability to acquire new web development knowledge and skills by implementing the web page for vehicle tracking on a live map displayed in a web browser.

Project Objectives

Following are the major objectives set for this project:

- Learn about APIs, and how to use them with AJAX,
- Use Google API to display a map, with interactive data,
- Access a database via PHP to perform basic queries,
- Write a dynamic web page, using PHP and Javascript (JQuery),
- Learn about CSS styling (possibly using Bootstrap or PureCSS libraries),
- Work in a group

Project Scenario and Tasks

The manager of Auckland Transport wants to track and manage their fleet of vehicles via an interactive web-based application which includes an interactive map, a connection to a live API, and a database.

In this project, you are required to produce a prototype of such an application. The application should be a single dynamic web page. The following are the requirements and features to be implemented in this prototype:

1. When the page loads, it should show the list of all available routes which are recorded in the database.
2. On selection of a route in the table, the map should display all current vehicles on this route. The interactive map should be able to display more than one vehicle at a time.
3. On selection of a vehicle on the map the application should show additional information about this vehicle.
4. The map should auto resize to show the additional information for the last item added. If more than one item is added to the map, it should be resized so that all added items are in view.
5. The map should be updated every 30 seconds to reflect the updated vehicle positions.

Resources

The resources for the project are to be found on UC Learn under the Assessments | Course Project (http://learn.canterbury.ac.nz/mod/folder/view.php?id=631180) section. Apart from this project specification, the resources include the following:

1. Introductory information about APIs is included in the *What is an API In English, Please* PDF document.
2. The instructions for obtaining your own key for the Auckland Developer Portal PDF document.
3. An **akl_transport** database has been set up which contains tables for **routes**, **stops** and **trips**. These tables (the corresponding entities) have the following meaning. Stops are where vehicles let passengers board and disembark. A trip is a link between two stops. A route is a sequence of two or more trips. If you're more into graph theory stops are vertices, trips are edges between stops, and a route is a path. The **akl_transport** database permissions are limited so that only **SELECT** queries can be performed against it. You should not need to write **INSERT** or **DELETE** queries. The details for connecting to the database are as follows:
    - Hostname/IP: **csse-info263.canterbury.ac.nz**
    - Database: **akl_transport**
    - Username: **info263**
    - Password: **info263**

   These details should allow you to create a new database connection in MySQL Workbench just as it has been done in the tutorials. The above details are also in skeleton code **include/config.php** file. Currently the database server is not available outside of the university network, i.e. you cannot access it from home. In the near future it will be accessible off campus; we will let you know when this is possible.
4. You are not required to create a local database every time you want to work on the code, but we have provided an **akl_transport.sql** file if you wish to do so. If you import the data into the local MySQL server running on a lab machine, please remember to amend the database details in the **include/config.php** file.
5. A project skeleton including the basic structure of the web application to be developed. The following files are included:
    - The landing page of the application, **index.php**, calls PHP code to include the **include/header.php** and **include/footer.php** files from the **include** directory. The footer and header files contain the code for importing JQuery (JavaScript library), Bootstrap (CSS library) and other components which improve the appearance of the web page. You do not need to modify the header and footer files, but you are required to complete the JavaScript anonymous function which gets triggered after the browser has loaded the page. This function should be used to populate the web page with available routes.
    - The **requests.php** file includes code to perform requests against the Auckland Developer API, particularly of interest is the **apiCall()** function
    - Example code showing of how to perform a manual API request using **requests.php** `can be found in` **example_request.php**.
    - The **certs** directory includes a ca-bundle list of SSL certificates used in PHP to perform secure requests to the Auckland Transport API.
    - The **css** directory includes the basic styling for the header and footer in the main application page. Feel free to augment the basic CSS rules with your custom entries.

- ○ The **`scripts`** directory contains an empty **`map.js`** file where you should place the code for interacting with the Google Map. Hint: it's probably a good idea to put your **`initMap()`** function/code here.
- ○ **`include/config.php`**, contains credentials for connecting to the database, you should also put your Auckland Developer Portal API key here.

Getting Started Tips

It is suggested here that you start your work by figuring out how to display a live map from Google Maps API. The following tutorial, Adding a Google Map with a Marker to Your Website might be very useful:

https://developers.google.com/maps/documentation/javascript/adding-a-google-map.

The tutorial link given above walks you through the steps of creating an HTML page, adding a map with a marker and getting an API key for Google Maps API.

Next, you may want to write the code (in PHP) to access the **`akl_transport`** database. The route name **`select`** query to get the route information from the database can be a simple query without the requirement to sanitise or prepare input because no user input is required in this case. The route lookup query, however, has to use sanitised input parameters and query parameter binding.

Next, you might want to figure out how to interact with the Auckland Transport Developer Portal. You will also need to get an API key for the portal. Follow the instructions provided in the document for this purpose. The portal also contains a lot of useful examples and information. You can use Javascript (JQuery) to send API requests to the Auckland Developer Portal; however, we recommend to perform the requests via your web server and use the included **requests.php** file. You will need to look through the Auckland Developer Portal API to find the type of API request you want to perform. Hint: it probably has something to do with vehicles.

Next, it might be a good time to write the JavaScript (JQuery) code to send API requests to your php web server. This is a tricky part, and you need to understand what you are doing in this part. There is a JQuery intro section in the course textbook. Alternatively, you can use online resources to get started with JQuery, such as this one:

https://www.w3schools.com/jquery/default.asp

In addition, JQuery, along with CSS HTML page styling will be introduced in lectures in due course. But it would be pragmatic on your part to be proactive about using online resources as you are developing your project. For Bootstrap resources you may want to start here:

https://www.w3schools.com/bootstrap/default.asp

You will likely want to finish your work by using the JavaScript **`setInterval`** timer to query the API at a specific interval in order to refresh the vehicles displayed on the map for any specific route.

Getting Frustrated Over Programming

In all likelihood, everyone in the class will have to go through the frustration of having things not working and not knowing why. This is inevitable with any non-trivial development project. This is a good time to plan your work carefully, working in small increments, testing the new code and, most importantly, keeping track of your work.

It is highly recommended that you consider the use of a version control system for helping you keep track of your code and collaboration. In any realistic development project it would be unfathomable to rely on manual code backups and email to share the code with your colleagues. If you have not yet come in touch with version control, you may think of this opportunity in terms of getting a skill with is invaluable for your future. Follow this link to find out about version control in general, and, specifically, use of Git for version control:

https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

Alternatively, here's the same information recorded in an online video:

https://git-scm.com/video/what-is-version-control

For a practical tutorial on using Git and GitHub follow these tutorials, which are essentially equivalent, but present the same skills and information in a different way:

- https://rogerdudler.github.io/git-guide/
- https://swcarpentry.github.io/git-novice/

Please note, that it is your individual responsibility to make sure none of your work is lost. And yet, despite all your best effort, you will most likely find yourself stumped by your code. The first, most important thing in this situation is to try and get to this point early, rather than leaving it to the last minute. The second most important thing here is to remember that you are not alone:

https://www.codementor.io/matstc/avoid-frustration-as-programmers-ge54ddszr

Getting Help

If you are mercilessly stuck, your first point of contact should be your group members. It is expected that you learn from each other and work together quite closely.

The next thing you might want to do is to post your questions on UC Learn. You are encouraged to use the rest of the class as your sounding board, but, needless to say, solutions can not be posted on UC Learn.

Also, you are advised to keep an eye on the INFO263 General Forum messages, as there are likely to be updates and clarifications posted there in due course of time.

Please feel free to use the standard lab streams to ask your tutors for help and advice on the development of your project. Office hours too can be used to get help with your project.

As a last resort, you may email the INFO263 teaching team, Constantine and Dion, if you are stuck with programming or don't know what to do.

However, if you have problems within your group, please notify your course coordinator, Constantine, immediately. It is inevitable that some groups will struggle to communicate and collaborate. These issues should be taken into account, noted on record, and, hopefully, resolved as soon as possible.

Submission Expectations

Your group is required to submit your code and a report. The specifications for the code and report are provided next. All files should be packaged as a zip file for submission.

- Submit your project via UC Learn by 9:00 p.m. on October 16th.
- Late submission will be accepted until 9:00 p.m. October 20th with a 15% penalty.

Code

Include all the code required to run your website. Since you are not required to alter the data in the database, you do not need to include a database file in your submission. If you find yourself writing more than a 1000 lines of code, come and talk to the teaching staff, because in all likelihood the project can be completed in about 300 lines of code.

Report

The report should contain a brief description of how your website works, explicitly separating the client-side and server-side functionality and your decision-making process related to this. The reports should also provide a reflective summary of your group's teamwork. For example, please state how you communicated with each other, how you managed and shared your code, how you separated the tasks between team members, what went well in this project and what didn't go well, and what you would do differently next time around. The report should be two to three pages long. As a guide, the report is going to be satisfactory if you give a paragraph to address each of the report requirements listed above.

Marking Schedule Out of 100

Task	**Description**	**Value**
1	Showing routes: making routes selectable from the main application page.	15
2	Show vehicles on a route: when a route is selected the vehicles for that route should be displayed on the live map in the browser. To get the full marks for this question, the input data should be sanitised, and a prepared query should be used to retrieve the response data.	30
3	Show vehicle information: formatted output of the specific vehicle information.	15
4	Map resizing	10
5	Vehicles’ position updating	10
Web page style	Marks for using CSS, and styling the web page	5
Codestyle	Codestyle marks: code formatting, basic commenting, sound code structure, separation into functions	5
Report	Project summary: teamwork description, reflection	10
	Total	100