# Basic Web Application Rationale

This document describes the development process and outcomes for a basic web application. The web application must:

* Monitor something
* Allow a user to register
* Allow user to login
* Create database entries
* Edit database entries
* Delete database entries
* Display data on the front-end

## About the Web Application

This Basic Web Application, named Fuel Watch, will monitor fuel prices in Western Australia. It will allow Usersto register, search the database, display the search results and display a search result location on Google maps. Also, Registered Users will be allowed to login, search the database, display the search results, display a search result location on Google maps, add a favourite search pattern, edit a favourite search pattern and unregister.

## Fuel Watch will be available to users through a web browser. Since part of this application requires client side software, the application is distributed and its architecture will be that of client server.

There are two main processes:

* Build and Configure the server
* Develop server-side application

## Server Configuration

The process of building and configuring the server consisted of choosing a hosting organisation and determining what technologies (software stack) will be leveraged to produce the web application.

AWS provides free tier web services to students. Amazon Elastic Compute Cloud is a free tier option that allows users to launch a virtual server with a server operating system installed.

Following a tutorial provided by AWS, the instance was successfully launched with an Ubuntu Server operating system. The next step was to configure the server environment with the chosen software stack. The software stack that was chosen was LAMP, since these were the main technologies taught in the unit.

After installation of the chosen technologies some configuration was needed. Most notably, this included configuring Apache to use php, setting up a virtual host, installing Apache modules and adding directives to virtual host configuration file.

The outcome of the build and configuration *thus far*, was a working live server that could be accessed remotely through a web browser or CLI. During the development of the server-side application code further configuration was needed. Specifically these were setting up databases and altering virtual hosts configuration files.

## Develop Server-Side Application

The key considerations for developing the server-side application were:

* Setup databases and tables
* How to get and store fuel prices periodically
* How to get and store location information
* How to organize the code

The setup of the database was achieved by secure shelling into the server and the CLI. The following tables were created. Primary keys are underlined and Foreign keys are italisied.

Stations := (title, trading\_name, location, phone, latitude, longitude, address)

ulp := (*title*, product, brand, date, price)

ulpp := (*title*, product, brand, date, price)

diesel := (*title*, product, brand, date, price)

lpg := (*title*, product, brand, date, price)

ron98 := (*title*, product, brand, date, price)

e85 := (*title*, product, brand, date, price)

dieselbrand := (*title*, product, brand, date, price)

wa\_locality : (locality\_pid, locality\_name)

wa\_locality\_point : (locality\_point\_pid, *locality\_pid*, latitude, longitude)

users: (username, password, date\_created)

favourite: (locality\_pid, *username, locality\_name ,distance, product*)

Once these database, tables and privileges were created it was possible to start to populate the database with fuel prices and location information.

The fuel price data is available from <https://www.fuelwatch.wa.gov.au> as rss/xml data. A script was written and the systems ‘crontab’ was edited so that the script runs periodically. This script is found in the [yiifuelwatchapp](https://github.com/a-a-w-136/yiifuelwatchapp)/[fuelwatchapp](https://github.com/a-a-w-136/yiifuelwatchapp/tree/master/fuelwatchapp)/[admin](https://github.com/a-a-w-136/yiifuelwatchapp/tree/master/fuelwatchapp/admin)/fuelprices/ directory. This data makes up that which is searchable by the user.

The location data is available from <https://data.gov.au/search?q=gnaf>. Appropriate data was downloaded and a script was written to populate the address data. This script and data can be found in the [yiifuelwatchapp](https://github.com/a-a-w-136/yiifuelwatchapp)/[fuelwatchapp](https://github.com/a-a-w-136/yiifuelwatchapp/tree/master/fuelwatchapp)/[admin](https://github.com/a-a-w-136/yiifuelwatchapp/tree/master/fuelwatchapp/admin)/waaddressscript directory. This data is used to provide a list of valid locations in Western Australia and when a search is performed. Specifically each location has latitude and longitude and so does each petrol station.

The previous two scripts are administrative to the web application, however the data that is gathered and stored is used by the Fuel Watch app. This leads into how the code is organized.

Due to the features that a framework can provide to rapidly develop web applications Fuel Watch was to be designed under control of a framework. Yii2 was chosen because it is php framework that is built around the MVC pattern. MVC eased development by way of separation of concerns.

After installation of the framework the following occurred:

* A template was created for use by multiple pages
* Views were created and styled
* Models were created
* The Controller was configured
* Files were configured for database queries

## Develop Server-Side Application

This web application was developed both with and without a framework, both of which are available at <https://github.com/a-a-w-136>. The version without the use of a framework followed an MVC pattern and the problems with the application are outlined in its repository. The main difference when developing the project with the framework was alleviation of developing code that handles the controller, database and session initialization functionality.

In the real world I would consider changing some important parts of the application. For instance this application gets and stores fuel prices and the search is only ever for a particular day. Given that it would have been more efficient to have some client side scripting that request the XML, parses it and displays it to the user.