|  |
| --- |
| Cloud Computing: Assignment 1 |
| Electronic Vehicle Application |
| Documentation guide |

|  |
| --- |
| Jenom Kassim  3-18-2020 |

Contents

[Introduction 2](#_Toc35417043)

[The App.yaml file 2](#_Toc35417044)

[User Login / Logout and Jinja 2](#_Toc35417045)

# Introduction

EVHub is a PaaS application that houses a database of electric vehicles. The application allows users add and update vehicles in the database as well as perform comparisons of multiple cars in the database and leaving reviews on the cars in the database.

The application was built using Python 2.7, webapp2 and the jinja templating framework for deployment on the Google App Engine.

# The App.yaml file

The yaml file is responsible for informing Google App Engine about the runtimes and libraries needed for this application to run. It is also responsible for defining where requests should be routed in the application. The yaml file used in this application is called “app.yaml”.

The first part with the comment “RUNTIME USED” states which runtime was used (python 2.7 version of the runtime because the 3.0 wasn’t stable enough as at the time of development), it also states version 1 of the Google App Engine API was used (just like the python runtime, this is the only one available as at time of development), and finally the threadsafe states that our application is threadsafe so that the App Engine knows it is safe to launch multiple instances on the same server using threads and that they will synchronise correctly and will not cause data loss or race conditions.

The second part with the comment “LIBRARIES USED” states what libraries the App Engine should include to ensure the application functions properly. The libraries included are Jinja, which generates dynamically rendered HTML templates at run time to provide dynamic contents to individual users. Like the previous code blocks, we also specify which version of the library to use, which is the latest available to avoid as many security vulnerabilities as possible.

The third part with the comment “ROUTES FOR THE STATIC ASSETS USED” states which python modules and objects are responsible for handling requests sent to certain URLs. This application directs all requests to any url to the app variable that is defined in main.py as that contains our routing table. It also directs the application to the static directories housing the javascripts, css and image assets of the application when rendering of the html pages occurs.

# User Login / Logout and Jinja

The login and logout functionalities are part of the main cores of the application as they determine and control a lot of what users have access or are restricted to on the application. The Google user login / logout service was used to implement the login / logout functionality using Google user accounts as it is a proven, tried and tested authentication service.

# Classes and Methods

We will also introduce Jinja which we will use to generate dynamic HTML content at runtime in response to user requests. Both of these

are necessary features in modern web applications as if we wish to provide a

service to a user then we need a mechanism for determining which user we are

interacting with.

If we are going to tailor content to individual users we will also need to generate HTML content dynamically. One way of doing this is to embed a lot of

self.response.write() comments in code with all the necessary HTML commands needed but this is cumbersome and prone to error. A better way to deal

with this is to use HTML templates and provide a mechanism for parameterising

it and rendering it on the fly. This is what Jinja will provide.

In this example we will show how a user can login or logout and will use a

Jinja template to render two versions of the same page based on whether a user

is logged in or not