Task 4.2HD – Particle Function Web to Device

# Student ID: 219011171

# Name: Peter Stacey

## Q1. Submit a video that shows the outcome of the task. Include the link here.

<https://youtu.be/nWV-VrdmPsw>

## Q2: Create a repository named SIT210-Task4.2HD-ParticleCloudFunction on your GitHub account. Upload your code to the repository. Include the link to your repository here.

<https://github.com/pscompsci/SIT210_Embedded_Programming/tree/main/Task_4_2HD>

## Q3. Describe a real-life usage scenario for your system.

**Emergency Vehicle Priority:**

According to Kanungo and Singla (2014), traffic congestion is a serious problem, especially in major cities and following motor vehicle accidents.

One use of the system would be to provide priority lights for approaching emergency services to allow them in a congested area, to reach a traffic accident faster. By using traffic camera systems, controllers sitting in a traffic control room could see the direction of emergency vehicle approach and then manually switch traffic lights to provide them with priority passage and stop cross flowing traffic that is difficult to navigate.

This would limit the number of red-lighted intersections that emergency vehicles would need to cross; and also, by providing traffic flow for vehicles heading in the same direction, also reduce the need for emergency vehicles to cross to the wrong side of the road; which also provides for increased safety.

**Reference:**

Kanungo, A; Chetan Singla, A (2014); *Smart Traffic Lights Switching and Traffic Density Calculation using Video Processing*; in Proceedings of 2014 RAECS UIET, Punjab University, Chandigarh India, 6-8 March 2014, Accessed online on 10 April 2021 at <https://www.researchgate.net/publication/269310721_Smart_traffic_lights_switching_and_traffic_density_calculation_using_video_processing/link/563e5ec108ae45b5d28c563a/download>

## Q4. How would you improve this task using other technologies/libraries?

As described in the video, the existing solution has some issues:

1. Major security risk with access\_token and device ID hard coded in the HTML file
2. Ability to only control one set of lights per webpage
3. Non-expiring token, which also presents a security risk

To fix these three issues, the solution would be improved by:

1. Separating the HTML file into a client front-end and server back-end with the access\_token and device ID managed securely in a key store, so that direct access is not possible. All Particle cloud API calls would occur server-side, with the server code obtaining the access\_token and device ID as needed, without a user having direct access to either of them
2. In separating the code into a server backend, I’d support this with a database of devices, so that a single server backend could control multiple traffic lights from a single front-end interface. The server-side would also be written to decouple the code from the hard-coding of values, allowing a single server to manage multiple traffic lights from a single interface
3. Additionally, as a traffic light management system requires high availability, I might also look at containerising the application and utilising a container orchestration system (eg. Kubernetes) to allow horizontal scaling and load balancing, as well as high fault tolerance in order to deliver zero-downtime
4. One thing I would also look at would be utilising the current public cloud provider IOT platforms (eg. VMware Pulse IoT running on the VMware Cloud, AWS IoT or Azure IoT Hub)
5. Lastly, I’d also incorporate a Particle authentication workflow so that limited lifetime access tokens can be issued when needed, rather than using a non-expiring token

To improve the current task there are additional libraries and technologies that could be used. For example:

1. Adding a camera-based system to detect approaching traffic; and program traffic light changes based on input from the cameras. This would still allow for remote control of the lights, but also partly overcome the current lack of infrastructure. Solar powered embedded systems that include a low-resolution camera are widely available.
2. Cameras could be linked through either particle functions, publish-subscribe or IFTTT for example.