Table of Contents

[Lab 1 (AS3 – BIG-IQ) 1](#_Toc7005850)

[Lab 2 (Facts and AS3 with ansible – BIG-IP) 2](#_Toc7005851)

[Lab 3 (GIT) 3](#_Toc7005852)

[Lab 4 (POSTMAN) 3](#_Toc7005853)

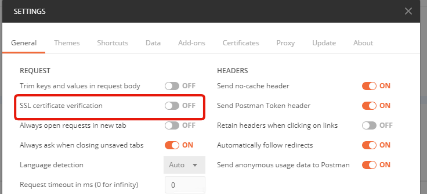
[Lab 5 (Validating declarations) 3](#_Toc7005854)

## Lab 1 (AS3 – BIG-IQ)

Goal

* Understand AS3 schema
* Create AS3 applications
* Read, update and delete AS3 applications
* Use the schema to create any type of application

Study the schema reference - [ <https://clouddocs.f5.com/products/extensions/f5-appsvcs-extension/latest/refguide/schema-reference.html> ] – (take some time going over it). This will be your guide when creating a new declaration.

* RDP to your windows machine in UDF
* Open postman (you can login to your account or not login at all), make sure to turn ssl verification off.
* 
* Import the provided collection and environment (desktop 🡪 lab files)
* Use environment variables to create the provided apps
  + <https://hive.f5.com/docs/DOC-50438> (instructions on using the collection)
* Create the following applications using the collection provided (IP ranges for VS – 10.1.10.201 – 210)
  + HTTP
  + HTTP with LTM policy
  + HTTPs Offload
  + HTTPs Bridging
  + TCP
  + UDP

(observe the response, use the ID in the response to check the status of the deployment task)

* Read the above applications and observe the output using the collection provided
  + List all apps
  + List one app
  + If you were designing a page which supports CRUD, think optimization while listing multiple apps
* Study the jinja2 templating used (in postman)
* Create the following applications (not provided) [ use the schema]
  + HTTP with SNAT pool
  + Based on SSL SNI match conditions, steer traffic to separate pools
  + A declaration using the metadata – how can you leverage metadata?
  + Compare declarations created using HTTP template and generic template
  + Can you disable a pool member when using HTTP template Vs Generic template?
* Study and play with the age parameter (<https://clouddocs.f5.com/products/extensions/f5-appsvcs-extension/latest/refguide/as3-api.html?highlight=history#method-get>)
* Study the “AS3 will all properties” page (have it as a ref) - <https://clouddocs.f5.com/products/extensions/f5-appsvcs-extension/latest/declarations/all-properties.html>

## Lab 2 (Facts and AS3 with ansible – BIG-IP)

Goal – Use ansible for BIG-IP fact gathering and creating applications using AS3

1. bigip\_device\_facts – Collect facts from F5 BIG-IP devices

<https://docs.ansible.com/ansible/latest/modules/bigip_device_facts_module.html#bigip-device-facts-module>

Device facts outputs data about existing BIG-IP devices in JSON format. For example, it can provide data about virtual servers, pools, ssl-certs, profiles etc. This data can be leveraged to analyze the existing environment to understand the existing applications (specially in large environments). Also, since its JSON it can be programmatically leveraged.

Lab steps

* Go to the directory /home/ubuntu/ansible/techdayansible/lab-bigip-facts
* Run the playbook (ansible-playbook -i inventory/hosts playbooks/site.yaml )
* Watch the output (on the screen as well as in the host\_vars),what is it obtaining ?
* Change the playbook to get facts for what you want (see docs above)

## Lab 3 (GIT)

Goal – Understand basic GIT concepts by experimenting with GIT bash and GIT GUI

Lab steps

* Create a bitbucket/github account (if you do not have already)
* Add your public key to your account
* Initialize a repo in the ‘gittest’ folder(desktop) using git bash
* Set name and email globally using git config
* Check status
* Add all files to the staging area
* Check status
* Commit with a message
* Create a repo in bitbucket
* Add the repo created as a remote
* Push to remote

## Lab 4 (POSTMAN)

Goal – Get comfortable with postman

Lab steps

* Observe the collection used for the AS3 lab
* Examine the templating used
* Check out the pre\_request\_scripts tab, what is it doing ?

## Lab 5 (Validating declarations)

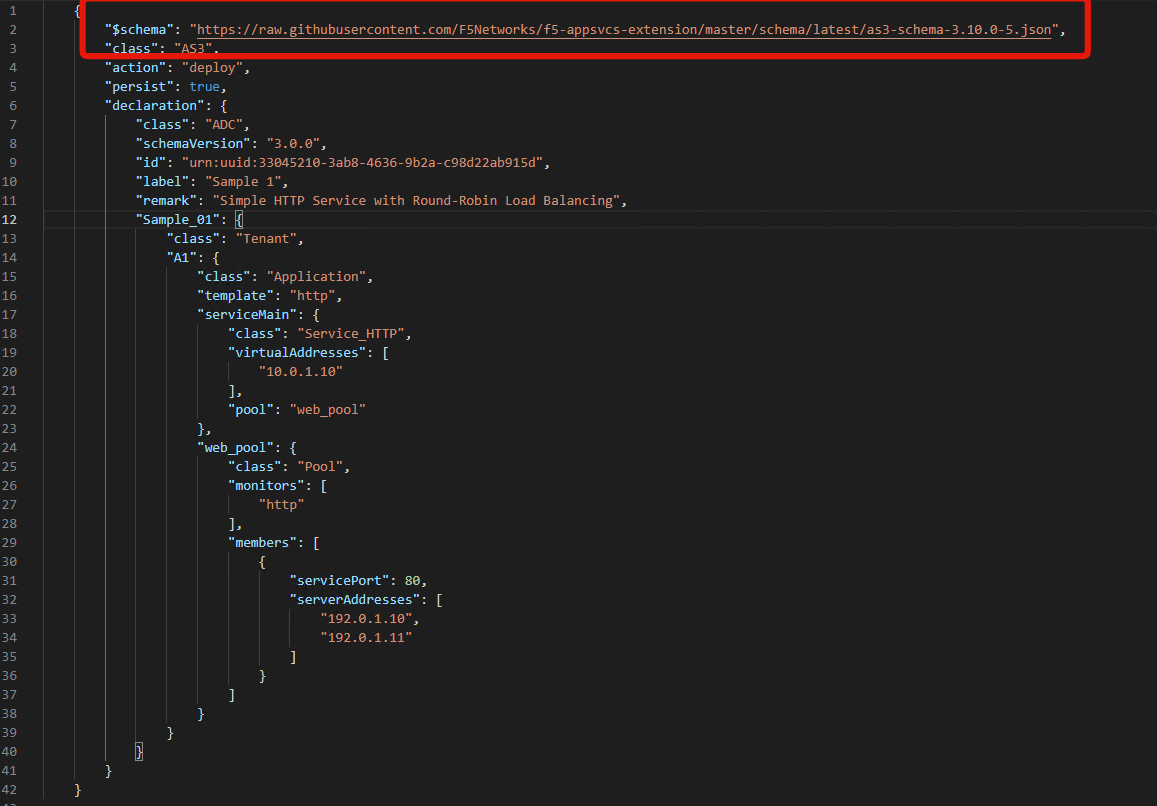
Goal – Explore validating declarations

1. VS code for validating declarations – This is used for manual validation of declarations. This comes with F5 official docs.

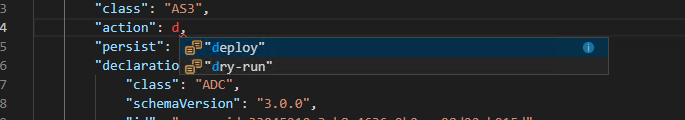
<https://clouddocs.f5.com/products/extensions/f5-appsvcs-extension/latest/userguide/validate.html>

Lab steps

* Study the link above
* Open VS code on the desktop
* Open the file VScode-validation.json (lab-files)
* Look at the $schema line up top (pointing to the latest schema)



* Notice that as you write the declaration, the editor will give you suggestions



* Notice that the editor will also do syntax highlighting if the declaration syntax is incorrect



1. Validation declarations programmatically – Most of the time, the declarations will be handled by a piece of code, and there should be a way of programmatically verifying if the declaration is valid or not. There are many 3rd party validators available for that.