**SALESFORCE AUTOMATED PERFORMANCE AND REGRESSION TESTS FRAMEWORK**

**Requirement:**

* + Laptop or Workstation
    - Modern processor
    - Minimum 8 GB RAM

**Software needed:**

* Install [Git](https://git-scm.com/downloads) and GitBash.
* Install Java 1.8.0\_41
* Maven
* Your Preferred IDE, IntelliJ IDEA recommended.

**SET UP**

* Download repo <https://github.ec.va.gov/EPMO/empwr_qa_selenium>
* Edit shell script file **properties-builder.sh**

WORKDIR: Directory where you cloned the repo.

SALESFORCE\_NAME: first\_name.

SALESFORCE\_USERNAME: Your Salesforce username.

SALESFORCE\_PASSWORD: Your Salesforce password.

You can also leave the file as is and export environment variables in your .bashrc file:

export SALESFORCE\_NAME=yourfirstname

export [SALESFORCE\_USERNAME=yourSFusername@va.gov.fasaoa2dev](mailto:SALESFORCE_USERNAME=yourSFusername@va.gov.fasaoa2dev)

export SALESFORCE\_PASSWORD=yourSFpassword

* Compile and create artifacts locally to the default target directory from your working directory or through your IDE.

mvn clean install : will compile and run all unit tests.

mvn clean install -DskipTests

Installs dependencies and compile framework skipping unit tests.

mvn clean package -Djavax.net.ssl.trustStore=${PWD}\security\cacerts :

Compiles and runs all unit tests

mvn clean jacoco:prepare-agent package jacoco:report -Djavax.net.ssl.trustStore=${PWD}\security\cacerts :

Compiles, runs all unit tests and creates coverage report in jacoco default directory.

See https://mkyong.com/maven/maven-jacoco-code-coverage-example/

mvn clean -Dtest=ParticipantSearchTest test

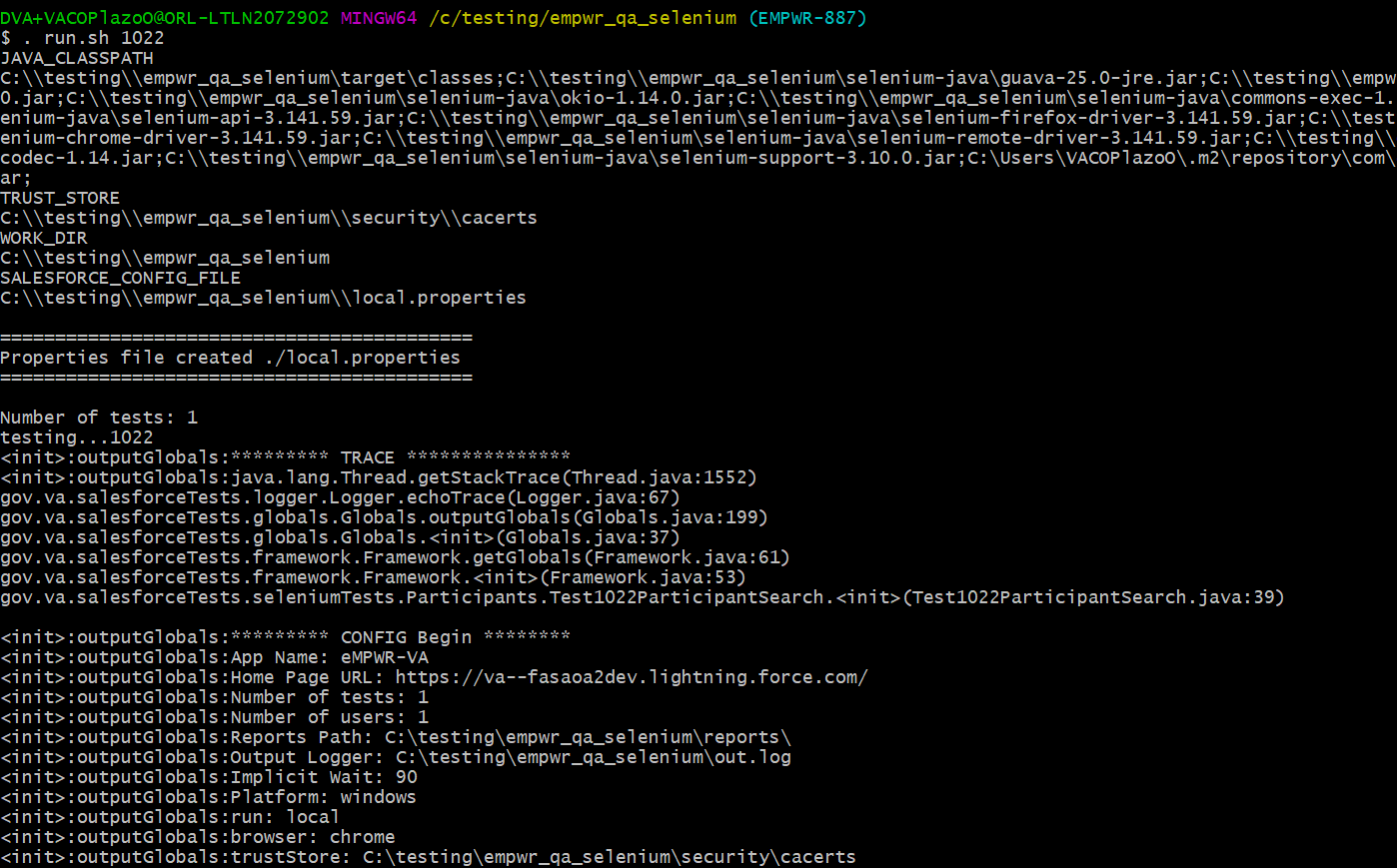
compiles and runs one particular test in this case the ParticipantSearchTest test.

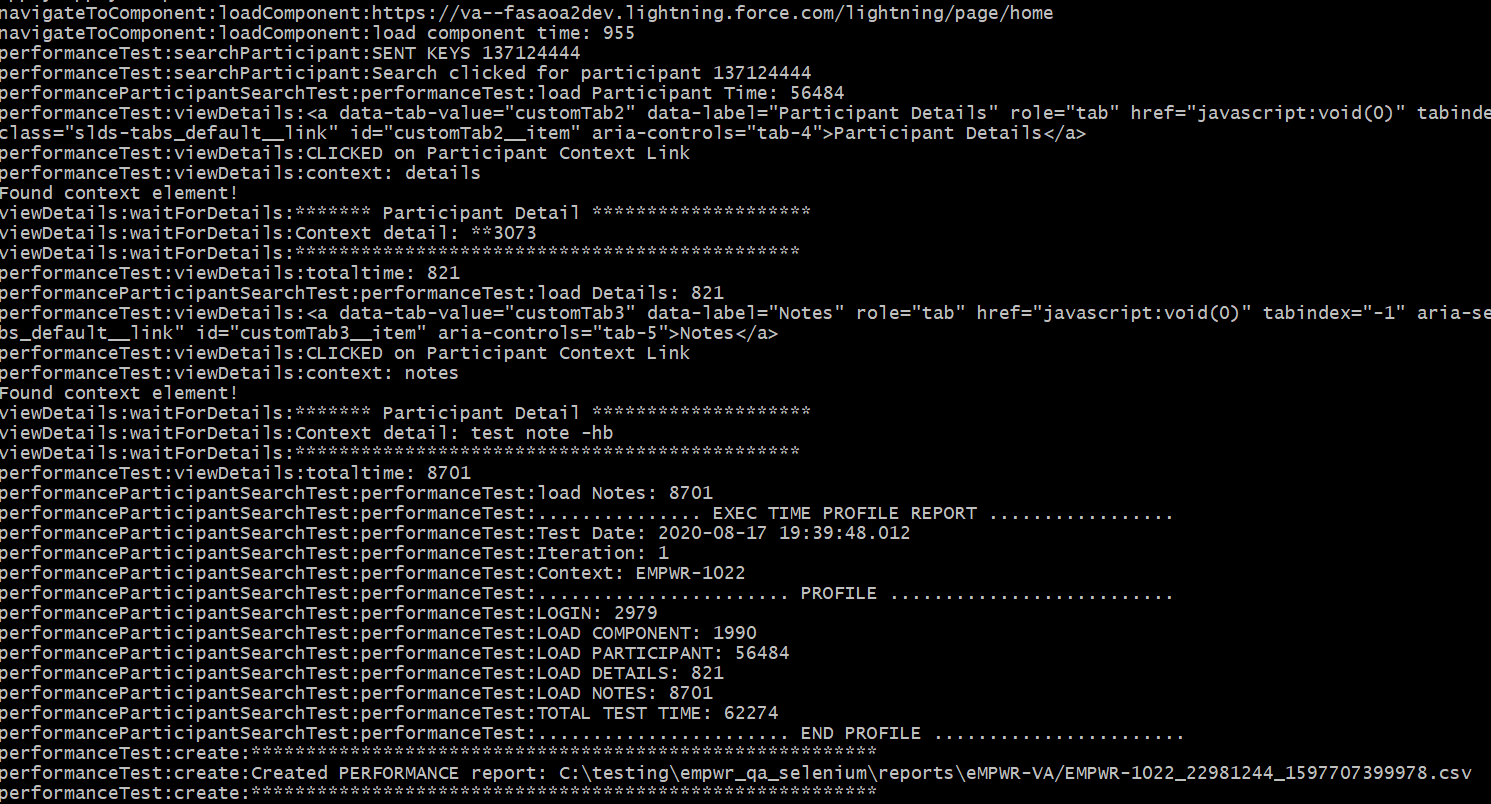
mvn javadoc:javadoc : generates latest javadoc documents in target/site/apidocs directory.

* To run a specific test in this case Jira issue EMPWR-1022:

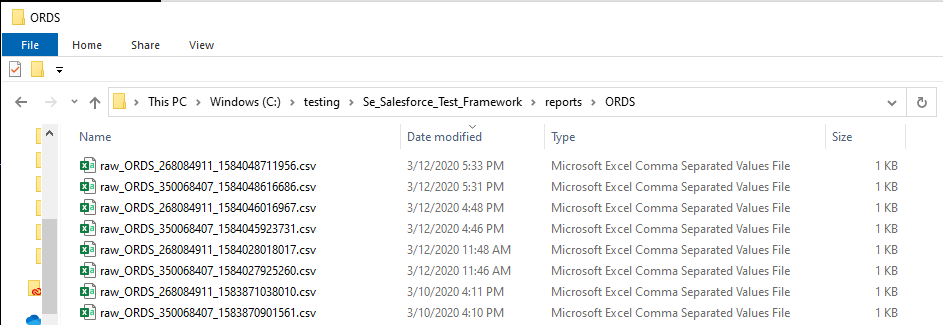
. run.sh 1022

You should see output on the git bash screen:

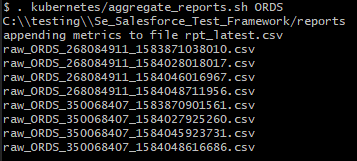




* Check that you have a couple of performance tests raw metrics reports.
  + Go to you prototype reports directory.



* Create your first or Update your existing aggregate report.
  + Run shell script **./kubernetes/aggregate\_report.sh.** The script defaults to MuleSoft but you can add parameter ORDS.

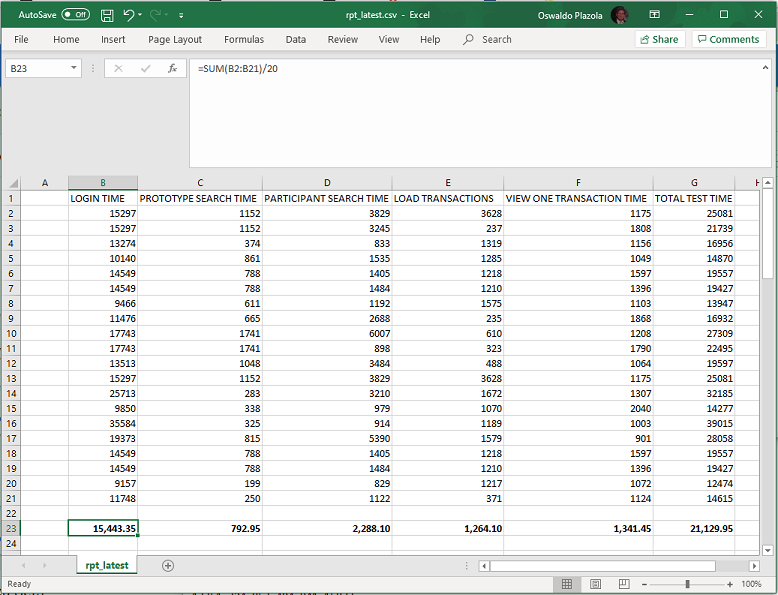


* + You should now have an aggregate performance metrics report in the prototype directory: **rpt\_latest.csv**

The raw performance metrics reports from the tests runs, get appended to the latest aggregate report and then get **deleted.**

* + Edit rpt\_latest.csv in Excel to obtain the average times. This has not been automated yet.

Save as a different filename to keep aggregating to rpt\_latest.csv on subsequent test runs.



# Local Automated Concurrent Test Environment with Minikube Setup

Install Minikube and a new empty cluster

* Pre-Requisites
  + Laptop or Workstation
    - Modern processor
    - Minimum 12 GB RAM
    - Minimum Storage to install the tools and deployments
  + [VirtualBox](https://www.virtualbox.org/wiki/Downloads) - Hypervisor
    - Go to <https://www.virtualbox.org/wiki/Downloads>
    - Click on Windows hosts under VirtualBox version # platform packages and save the file
    - Once downloaded, double click on the installer located in your downloads directory
  + [Kubectl](https://kubernetes.io/docs/tasks/tools/install-kubectl/) - Kubernetes Command-line tool
    - Go to (<https://kubernetes.io/docs/tasks/tools/install-kubectl/>)
    - Follow the instructions for your preferred installation method such as PowerShell, Chocolatey or Curl
  + [Minikube](https://kubernetes.io/docs/tasks/tools/install-minikube/) - Local Kubernetes platform
    - Go to (<https://kubernetes.io/docs/tasks/tools/install-minikube/>)
    - Click on latest release under Install Minikube
    - Scroll down the page and click on Windows/amd64 under the Distribution section and save the file
    - Once downloaded, create a directory for the executable (ex. C:\minikube)
    - Copy the minikube executable to the new directory and rename it minikube.exe
    - Note: Minikube can also be installed using the Chocolatey package manager
  + Install [Git](https://git-scm.com/downloads) and GitBash
  + Install Java 8, Maven and your favorite IDE

TEST MINIKUBE INSTALLATION

minikube status

minikube version

* Create Minikube cluster

From GitBash command prompt $ run the following:

* + Create a new empty Minikube local cluster called minikube:

minikube start --vm-driver="virtualbox" --memory 4096

kubectl version

* + Show all resources in the cluster

kubectl get all

* + Setup for using the Minikube docker environment

eval $(minikube docker-env)

* + Start your local DTR, Docker Trusted Registry:

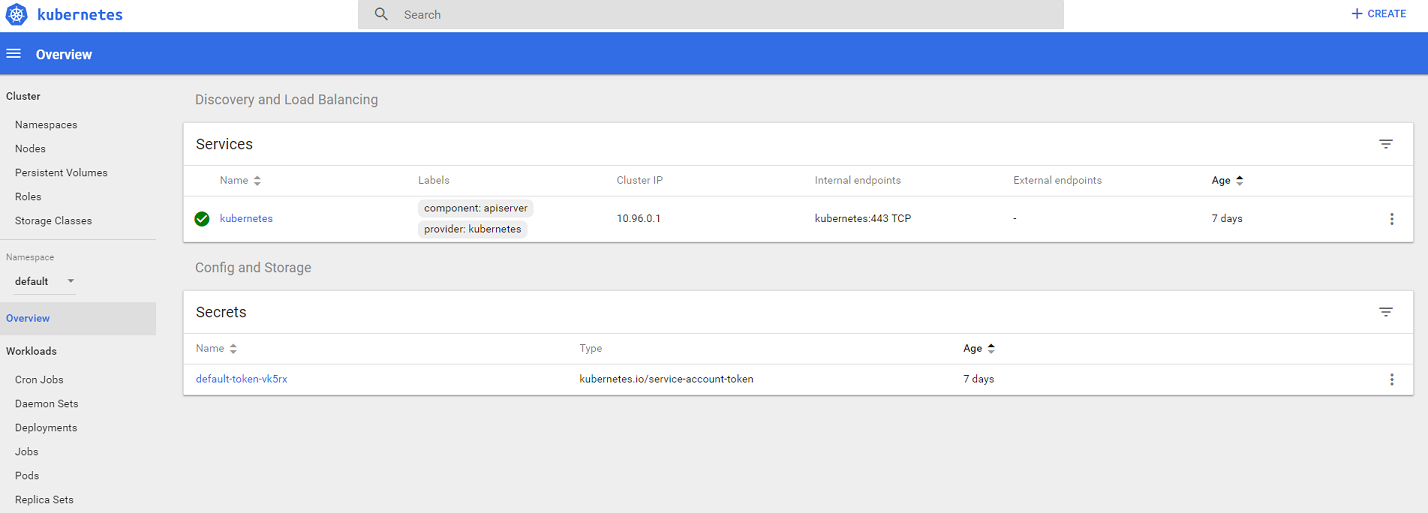
docker run -d -p 5000:5000 --restart=always --name registry registry:2

* + Start the Minikube Dashboard and show the URL.

minikube dashboard --url=true &

* + - Note the URL to view the minikube/kubernetes dashboard. It will be something like the url below. Go to that url on your favorite browser.

[http://localhost: 49376 /api/v1/namespaces/kube-system/services/kubernetes-dashboard/proxy](http://localhost:8001/api/v1/namespaces/kube-system/services/kubernetes-dashboard/proxy)



**SALESFORCE CONCURRENT AUTOMATED PERFORMANCE TEST STEPS**

* Edit **Dockerfile**

SALESFORCE\_NAME: Your first name.

SALESFORCE\_USERNAME: Your Salesforce username.

SALESFORCE\_PASSWORD: Your Salesforce password.

* Edit **installMax.sh** shell script for prototype, users, iterations, browser, etc…

# Test Settings

test.prototype=ORDS or MuleSoft // prototype to test

test.platform=linux

test.environment=docker // local or docker/k8 run

test.number\_of\_tests=2

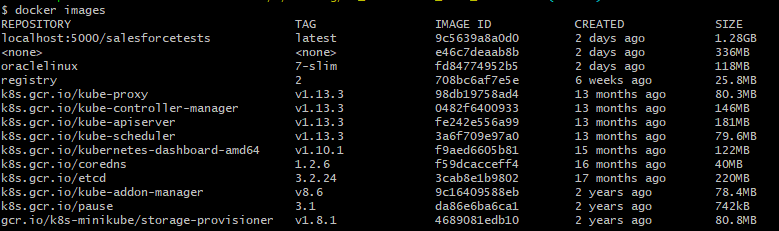
test.number\_of\_users=2

# WebDriver Settings

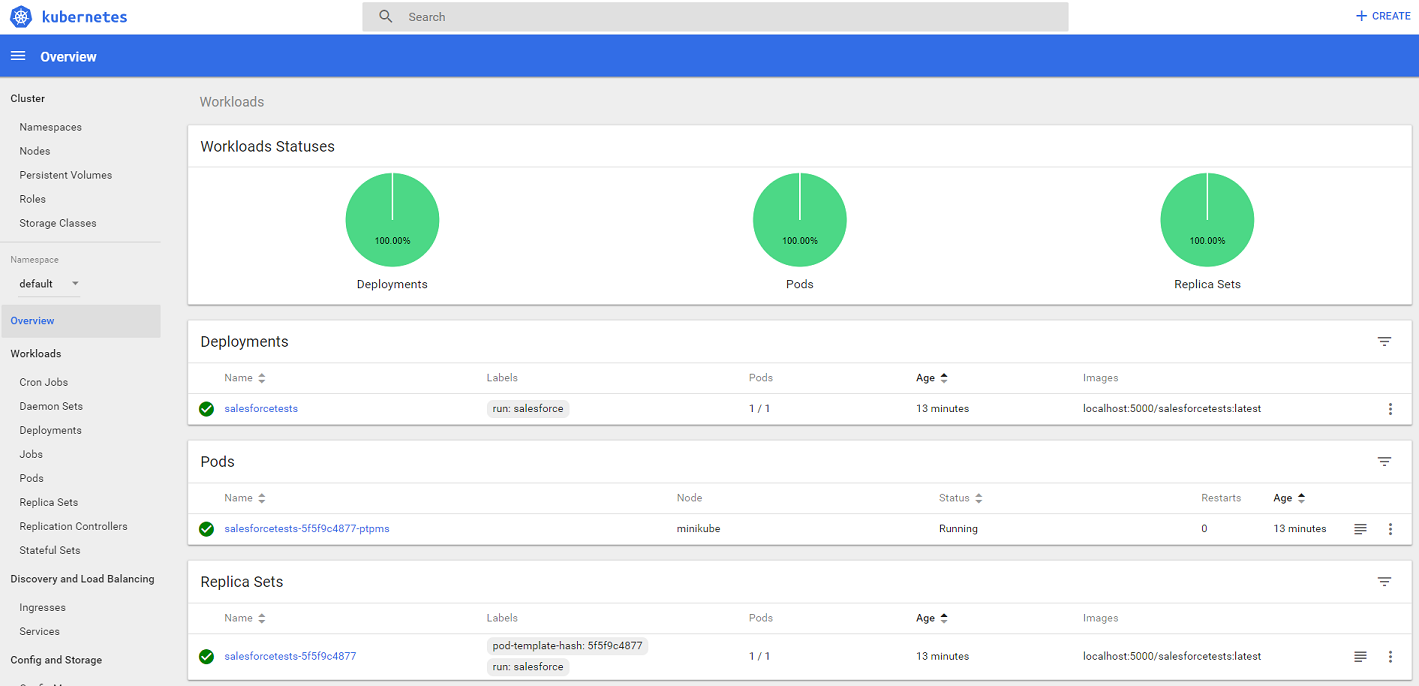
webdriver.type=firefox or chrome // depending on what browser/webdriver to test on

* Run shell script **build\_deploy.sh** 
  + Deletes unused docker images.
  + Maven compiles and creates artifacts in target directory.
  + Creates new docker image **localhost:5000/salesforcetests:latest.**
  + Deploys container to minikube.
  + May take up to 10 minutes.
* Confirm that docker image **localhost:5000/salesforcetests:latest** was built:

docker images



* Confirm deployment to Minikube:
  + Go to your minikube Dashboard. (The noted url above):

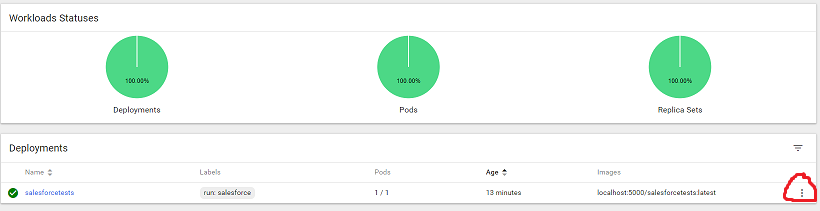


* **Scale Deployments**

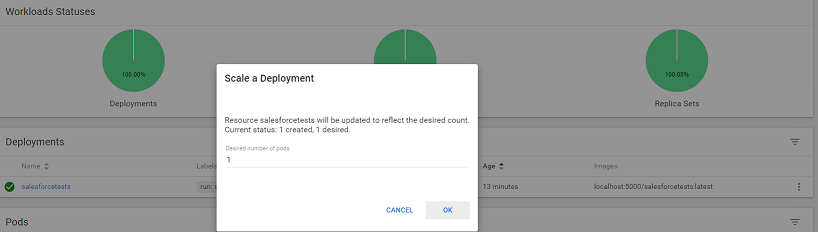
Increase number of docker images (k8 pods) in minikube.

\*If you don’t have enough resources on your laptop, minikube will crash and you will need to delete and rebuild the cluster.

* + Click on vertical 3 button icon in the Deployments pane:



* + Click on Scale



* + Increase number of pods (containers) in the ‘Scale a Deployment’pop up drop down.
* Check progress of test in a container instance.
  + Login to minikube:

minikube ssh

* + Locate ids of running salesforce tests containers.

docker ps | grep salesforce | grep -v pause

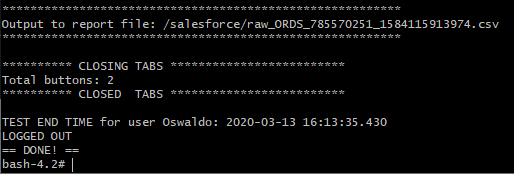


* + Log into one of the containers with a bash shell.

docker exec -it d1cc61eb7b57 bash

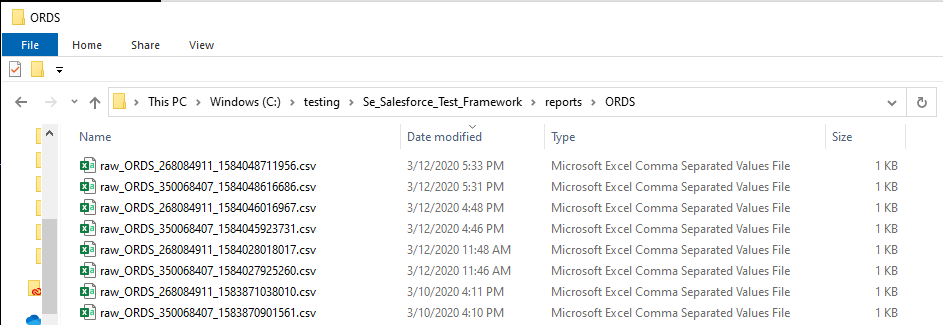
* + Check progress of tests in the log file out.log.

tail -f out.log

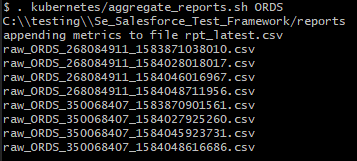


If you see the ‘== DONE! ==’, that container performance tests are finished.

* Download all raw metrics performance reports from all the containers to your local WORKDIR.
  + Run shell script ./kubernetes/**get\_reports.sh**
* Check that performance tests raw metrics reports were downloaded from the containers.
  + Go to you prototype reports directory.



* Create your first or Update your existing aggregate report.
  + Run shell script **./kubernetes/aggregate\_report.sh.** The script defaults to MuleSoft but you can add parameter ORDS.

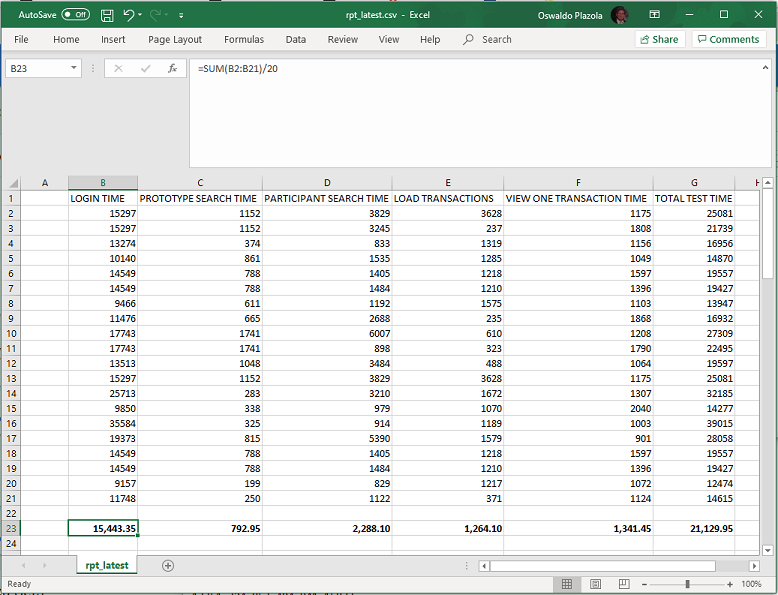


* + You should now have an aggregate performance metrics report in the prototype directory: **rpt\_latest.csv**

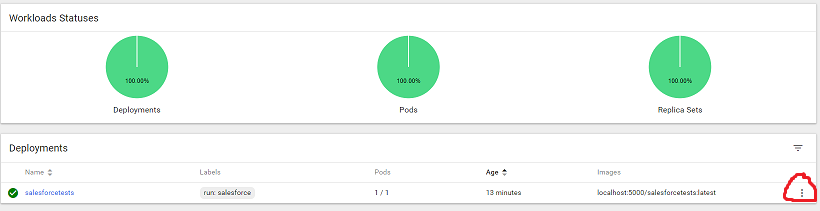
The raw performance metrics reports from the tests runs, get appended to the latest aggregate report and then get **deleted.**

* + Edit rpt\_latest.csv in Excel to obtain the average times. This has not been automated yet.

Save as a different filename to keep aggregating to rpt\_latest.csv on subsequent test runs.



* Delete minikube Deployments.
  + Click on vertical 3 button icon in the Deployments pane:



* + Click DELETE.

Your minikube deployment will take a minute to shut down. Refresh you browser to confirm

* To Stop the Minikube VirtualBox VM and remove it completely

minikube stop

minikube delete