# Assignment 1

Name – Sati, Ankit Date – 2/13/2022

Section – 001

SID – as14128

Total in points (Maximum 100 points)–

Professors Comments –

Affirmation of Independent Effort – Ankit Sati

**STEP 1 – Setting up all the platforms on a windows machine.**

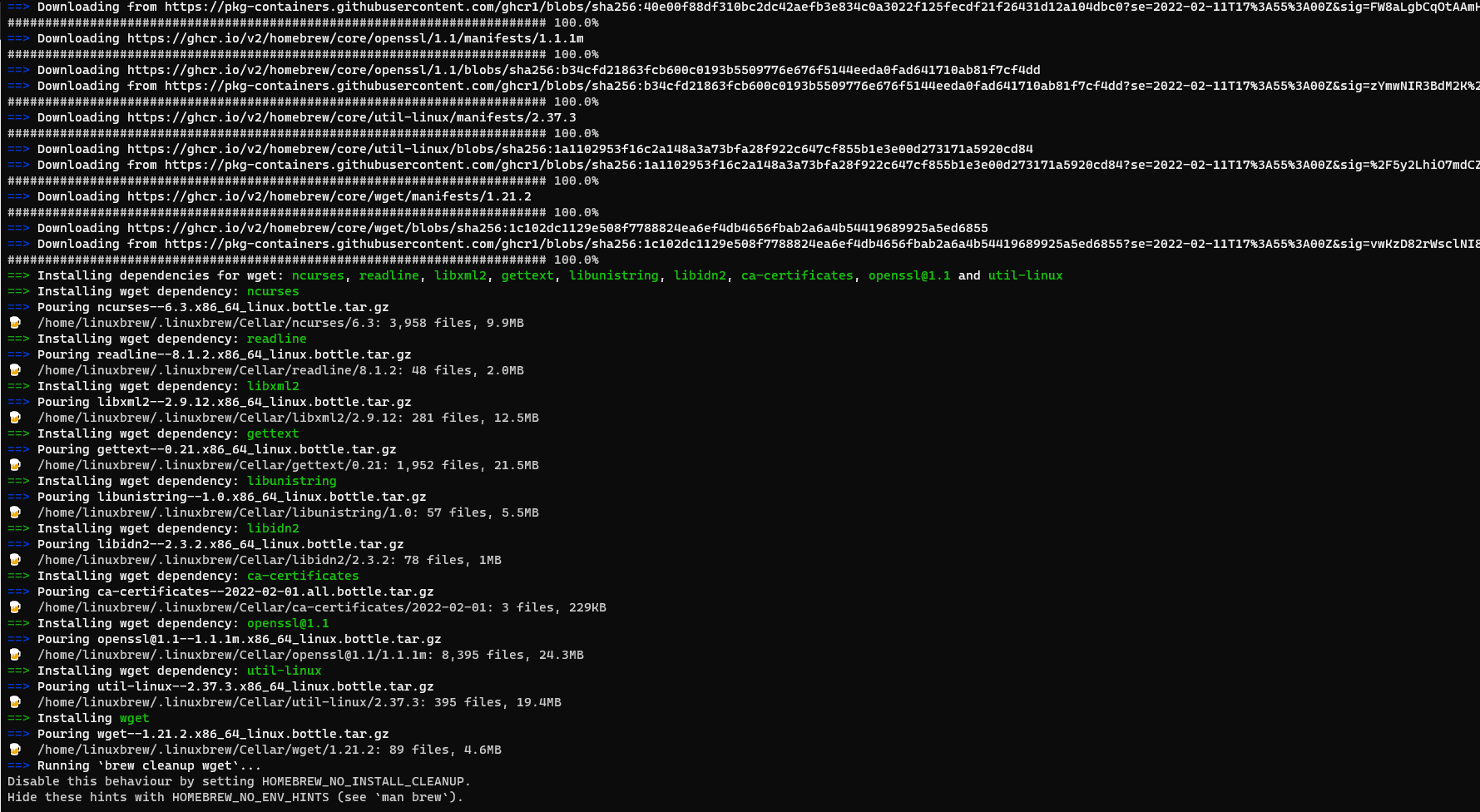
1. **Setting up Homebrew.**

* This is a software that is needed to find and install any missing file during the entire process of setting the cloud and deploying the resources.
* Irrespective of the application being installed on the windows device, homebrew came in handy on many steps of the process for me.
* Used while installing the Tanzu instance.
* Later used in the cli of the AWS deployment.
* While the SUSE(64) was deployed, this was helpful in installing the bash.
* Finally helped me in the shell where many commands failed in the beginning and **Homebrew** was able to download the missing repositories.

Confirmation of installation.



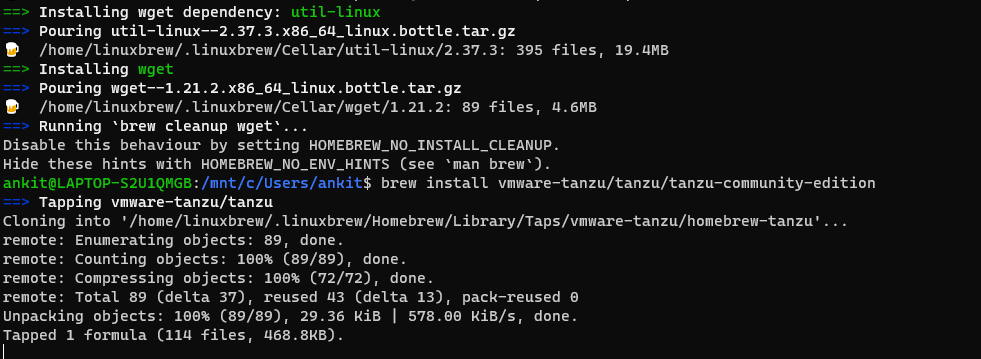
HomeBrew Dependencies



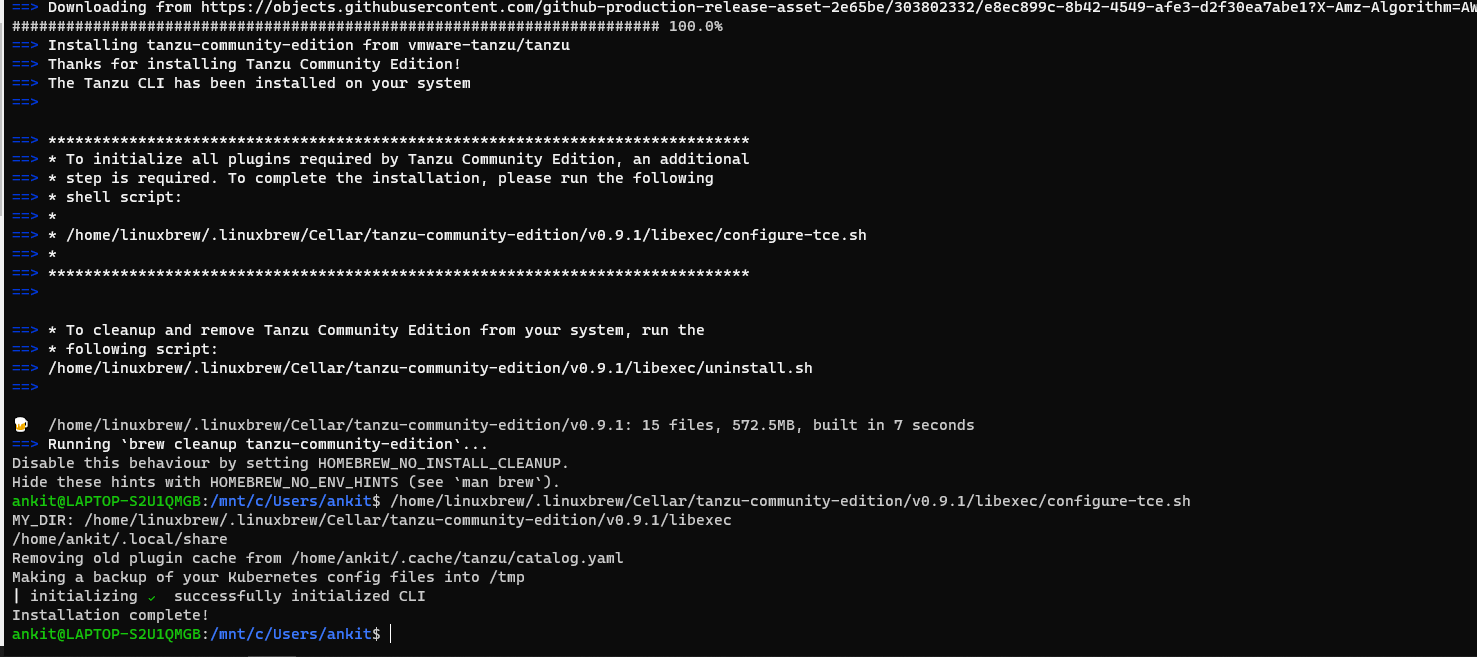
1. **Setting up Tanzu**

* Tanzu is a suite of products that helps users run and manage multiple Kubernetes (K8S) clusters across public and private “clouds”.
* While introducing Kubernetes as a first class VMware product, it still keeps strong ties to the VMware virtualization portfolio. It consists of: Spring Runtime – app framework.
* This came in handy in many places post the installation for managing the resources that I made available to myself on the cloud.
* Post installation of the AWS repositories for managing instances.
* Secondly while creating the VM instances on the cloud space, we could easily track and see the activities of the resources.
* Aligning resources to the SUSE instance.

Confirmation of Tanzu installation



Required resources setup and completed.



Finally Lets get started with the Assignment.

1. **Setting up the AWS account with the required instances.**

Required instances.

* EC2(**Compute server**) – This is basically a **regular server** instance that is used to deploy and the required resources over the VM as per the choices made by the users.

- This is used to deploy the VM.

- Manage resources over those VM’s.

- Finally to migrate services and monitor volumes.

* S3(**Storage utility**) – This is a basic protocol that acts like a storage bucket.

- The prime feature of this protocol is to deal with the data as per service request.

- We need this to store the data in the **data buckets** which are later used to store and move the data across volumes created.

**Initial account setup.**

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application

Description automatically generated

1. **Setting up the SUSE 64 bit and Kubectls Instance to manage services.**

Graphical user interface, text, application

Description automatically generated

* This is just another instance to manage the services of all the cloud applications.
* Firstly We need to download a CLI client to manage all the different shell services like Azure, AWS SUSE etc. I will use the powerShell application as attached below.

Text

Description automatically generated

* Secondly we need to set up the SUSE instance as required in this HW assignment.(Download SUSE rep)

Text

Description automatically generated

* the final Build.

Text

Description automatically generated

* Saving the required keys as mentioned in the assignment.

Key.pub identification

Type – RSA 4096

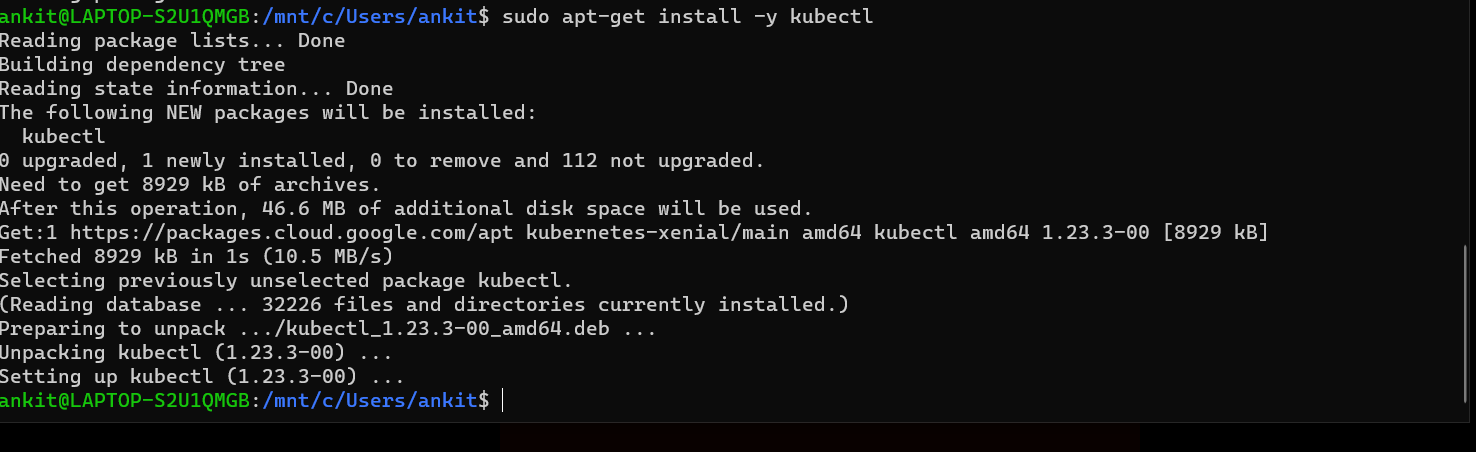
Key – Screenshot attached (Blurred the actual key for privacy)

Graphical user interface, text, application

Description automatically generated

1. **SSH into the instance that we have created.**

* The final build needs to be created on the GUI first.
* Post that we need to setup the EC2 instance and the S3 storage bucket
* Finally, we can **ssh** into the created instance.
* **Screenshot of the final build**
* **Moreover I have installed the KUBEctl application to manage the containers created across the cloud platform.**



1. **Setting up the Tanzu application.**

* I had discussed with the professor regarding cloud foundry. He wanted us to setup Tanzu instead of this service.
* Secondly Tanzu needs to be installed prior to the above so that it can access all the cores that the CPU has to offer.
* Above I have discussed the setting up of Tanzu, but I will paste the same procedure below.

**Tanzu**

* Tanzu is a suite of products that helps users run and manage multiple Kubernetes (K8S) clusters across public and private “clouds”.
* While introducing Kubernetes as a first class VMware product, it still keeps strong ties to the VMware virtualization portfolio. It consists of: Spring Runtime – app framework.
* This came in handy in many places post the installation for managing the resources that I made available to myself on the cloud.
* Post installation of the AWS repositories for managing instances.
* Secondly while creating the VM instances on the cloud space, we could easily track and see the activities of the resources.
* Aligning resources to the SUSE instance.

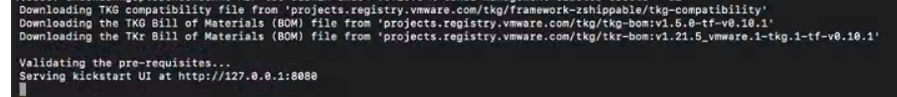
Confirmation of Tanzu installation

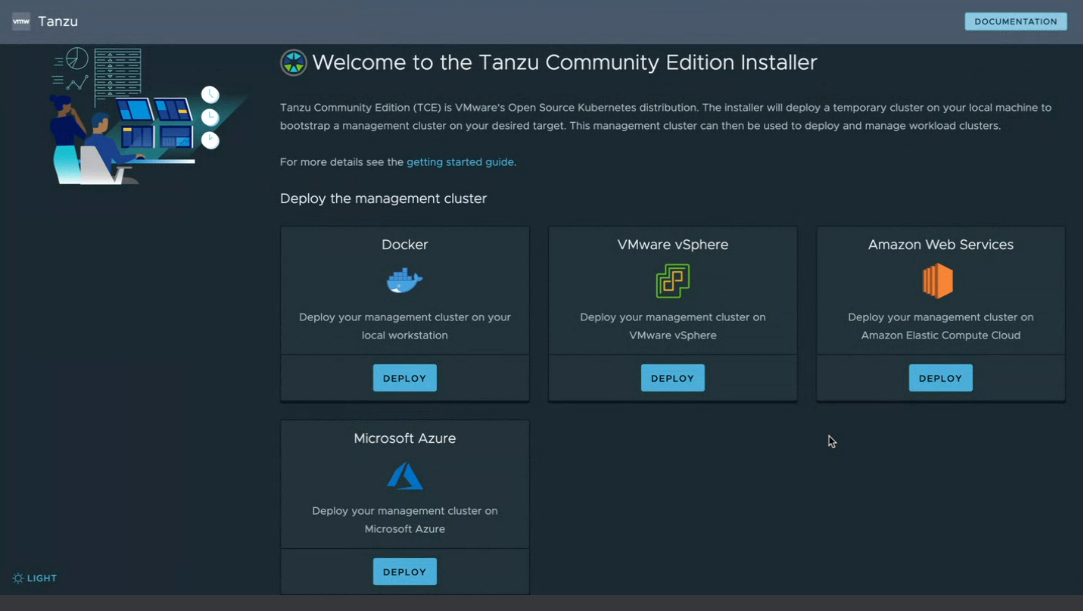
Text

Description automatically generatedRequired resources setup and completed.Text

Description automatically generated

Final Deployment.



****

1. **Kuard and MiniKube Setup.**

* Step 1. Get the kuard application.

remote: Enumerating objects: 1418, done.

remote: Counting objects: 100% (8/8), done.

remote: Compressing objects: 100% (8/8), done.

Receremote: Total 1418 (delta 2), reused 2 (delta 0), pack-reused 1410

Receiving objects: 100% (1418/1418), 2.18 MiB | 2.29 MiB/s, done.

Resolving deltas: 100% (498/498), done.

* Step 2 – Install the GoPl
* Step 3 – Install the kubectl.
* Step 4 – Run minikube and make sure that all the above steps (1-4) are setup.
* Step 5. – Enable and check If the docker is able to deploy all.

Text

Description automatically generated

* Step 6 – Check the sample applications and make sure .sh and touch scripts are running in **Kuard.**

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

1. **(Optional)testing a Hello world application.**

* Clone the application to the same rep where the Kuard is and the Tanzu has been accessed at-least once on the core.

remote: Total 137 (delta 0), reused 0 (delta 0), pack-reused 137

Receiving objects: 100% (137/137), 155.86 KiB | 3.46 MiB/s, done.

Resolving deltas: 100% (20/20), done.

ankit@LAPTOP-S2U1QMGB:/mnt/c/kuard/hello$ ls

simple-go-web-app

* Switch back to Linux and run the Tanzu to touch the application.

Install dependencies.

Installing github.com/uudashr/gopkgs/v2/cmd/gopkgs@latest (C:\Users\ankit\go\bin\gopkgs.exe) SUCCEEDED

Installing github.com/ramya-rao-a/go-outline@latest (C:\Users\ankit\go\bin\go-outline.exe) SUCCEEDED

Installing github.com/cweill/gotests/gotests@latest (C:\Users\ankit\go\bin\gotests.exe) SUCCEEDED

Installing github.com/fatih/gomodifytags@latest (C:\Users\ankit\go\bin\gomodifytags.exe) SUCCEEDED

Installing github.com/josharian/impl@latest (C:\Users\ankit\go\bin\impl.exe) SUCCEEDED

Installing github.com/haya14busa/goplay/cmd/goplay@latest (C:\Users\ankit\go\bin\goplay.exe) SUCCEEDED

Installing github.com/go-delve/delve/cmd/dlv@latest (C:\Users\ankit\go\bin\dlv.exe) SUCCEEDED

Installing honnef.co/go/tools/cmd/staticcheck@latest (C:\Users\ankit\go\bin\staticcheck.exe) SUCCEEDED

Installing golang.org/x/tools/gopls@latest (C:\Users\ankit\go\bin\gopls.exe) SUCCEEDED

All tools successfully installed. You are ready to Go. :)

1. **Addition step 7 – Delete the amazon instance, Docker clean as well as the Tanzu instances.**

* As all the expectations have been met, I have gone ahead and deleted all the **amazon EC2 instances.**
* Deleted all of the Volumes that were used to deploy the apps and the services.
* Finally deleted the security group for the above **Bucket** as well as the **server.**
* Deleted the minikube resources set up for the deployment.

docker system prune -a

WARNING! This will remove:

- all stopped containers

- all networks not used by at least one container

- all images without at least one container associated to them

- all build cache

Are you sure you want to continue? [y/N] y

Deleted Containers:

0c034ae63a0c3b118e5b92170cde19f807378b03aa344b4aba452fa684451f64

4ce88a695e4c3632e161ffb824a406bb00d17c96fcb7a458efbce6750ccde0a1

79e5503c668611a54de623258303a427dcc6f5f231e5fe28725007b42e6316da

Deleted Networks:

minikube

Deleted Images:

untagged: **hello-world:latest**

untagged: hello-world@sha256:97a379f4f88575512824f3b352bc03cd75e239179eea0fecc38e597b2209f49a

deleted: sha256:feb5d9fea6a5e9606aa995e879d862b825965ba48de054caab5ef356dc6b3412

deleted: sha256:e07ee1baac5fae6a26f30cabfe54a36d3402f96afda318fe0a96cec4ca393359

* Purged the **tanzu** Services

deleted: sha256:d55b3a11c2c16903280e578808cfd5c0fa3dedf513a1ee7dd895f0bb622ece5b

deleted: sha256:ae76d11f561d551ee4c904dc7a5ad487923f4c4a68f9871bccedf865cd76c0d2

deleted: sha256:9de65d1e8b2782409b2420bf9347003a43e91bb65c1e4c8fbd7d098d6234f359

deleted: sha256:e0f8e3acb2bf7fe9384463ae7009179d299b211e7cf17c2bf9d8e5e248cfe5b0

deleted: sha256:0e64bafdc7ee828d0f3995bebfa388ced52a625ad2969eeb569f4a83db56d505

**Total reclaimed space: 1.144GB**