Experiment 01: Foundation Installation

1. Install Kind
a. For **MacOS**

Docker

$ **brew cask install docker**
$ **open /Applications/Docker.app**

KIND

$ **brew install kind**

$ **kind version**

Kubectl

$ **brew install kubectl**

$ **kubectl version**

Git

$ **brew install git**

$ **git version**

b. For **Windows** platform we'll install Chocolatey, Kind, and Docker Desktop. Kind will automagically install Docker Desktop

Open a Powershell window with "Run as Administrator"

For installing Docker Desktop & KIND we must have your execution policy set to bypass or something even less restrictive.

PS C:\Users\kubelord> **Get-ExecutionPolicy**

Bypass

NOTE: If the response comes back with Restricted d you will need to execute the execute policy update noted below

PS C:\Users\kubelord> **Set-ExecutionPolicy Bypass**

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): **A**

Set-ExecutionPolicy : Access to the registry key
'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell' is denied. To change the execution
policy for the default (LocalMachine) scope, start Windows PowerShell with the "Run as administrator" option. To
change the execution policy for the current user, run "Set-ExecutionPolicy -Scope CurrentUser".
At line:1 char:1
+ Set-ExecutionPolicy Bypass
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~
+ CategoryInfo : PermissionDenied: (:) [Set-ExecutionPolicy], UnauthorizedAccessException
+ FullyQualifiedErrorId : System.UnauthorizedAccessException,Microsoft.PowerShell.Commands.SetExecutionPolicyCommand
PS C:\Users\kubelord> **Get-ExecutionPolicy**
Bypass

Install Chocolatey, a Windows Package Manager. Now that we've confirmed or updated and confirmed our execution policy is correct.

PS C:\Users\kubelord> **Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))**

Creating ChocolateyInstall as an environment variable (targeting 'User')

Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'

WARNING: It's very likely you will need to close and reopen your shell

before you can use choco.

PS C:\Users\kubelord > **choco /?**

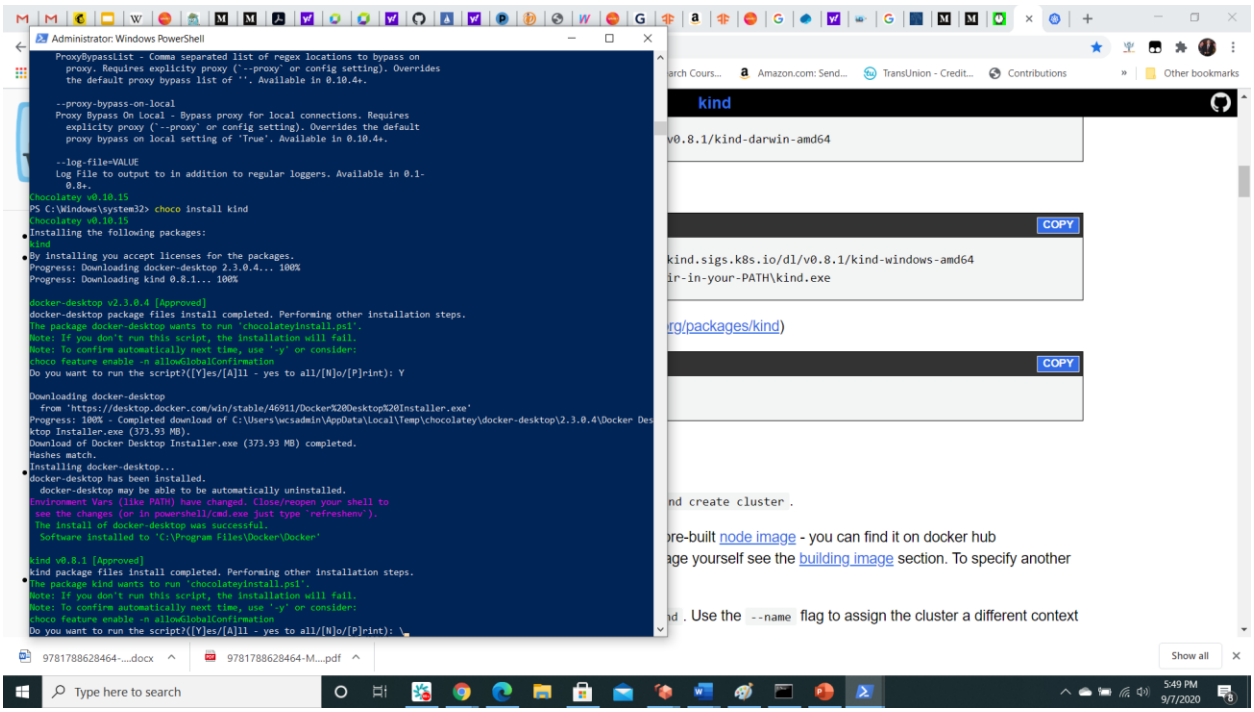
If the above does not return the help for Chocolatey, then close the Powershell prompt and open another one.

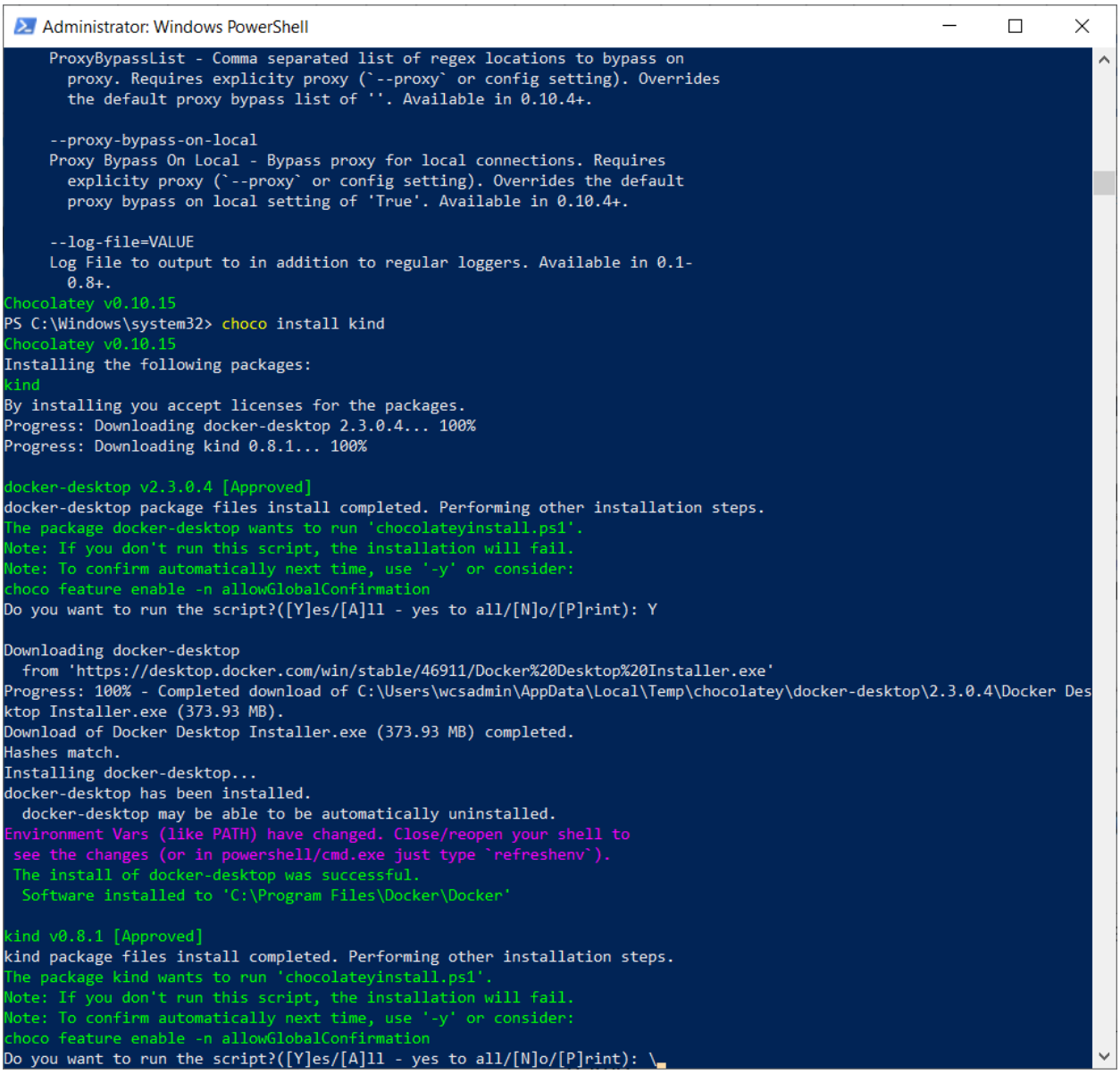
PS C:\Users\kubelord> **choco install kind**

Chocolatey v0.10.15
Installing the following packages:
kind
By installing you accept licenses for the packages.
Progress: Downloading docker-desktop 2.3.0.4... 100%
Progress: Downloading kind 0.8.1... 100%

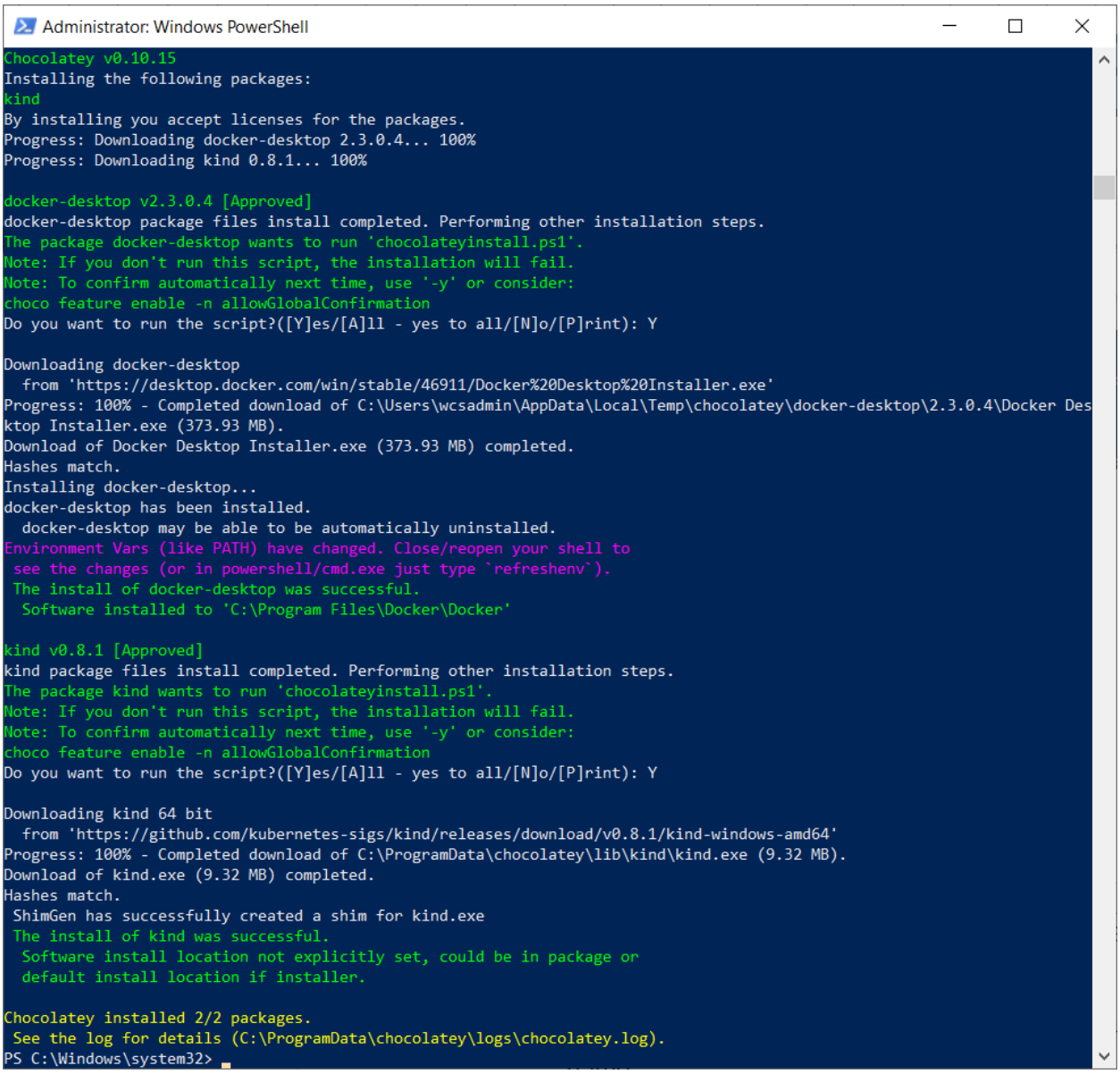
docker-desktop v2.3.0.4 [Approved]
…
 The install of docker-desktop was successful.
  Software installed to 'C:\Program Files\Docker\Docker'

kind v0.8.1 [Approved]
kind package files install completed. Performing other installation steps.
…
 The install of kind was successful.
…
PS C:\Windows\system32>
You can also install manually. If you were going to install manually, load the following URL in a web 'https://github.com/kubernetes-sigs/kind/releases/download/v0.8.1/kind-windows-amd64' for the Windows binary for other platforms view https://kind.sigs.k8s.io/docs/user/quick-start/

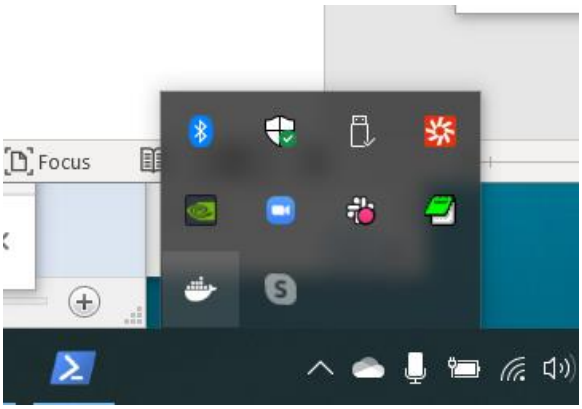




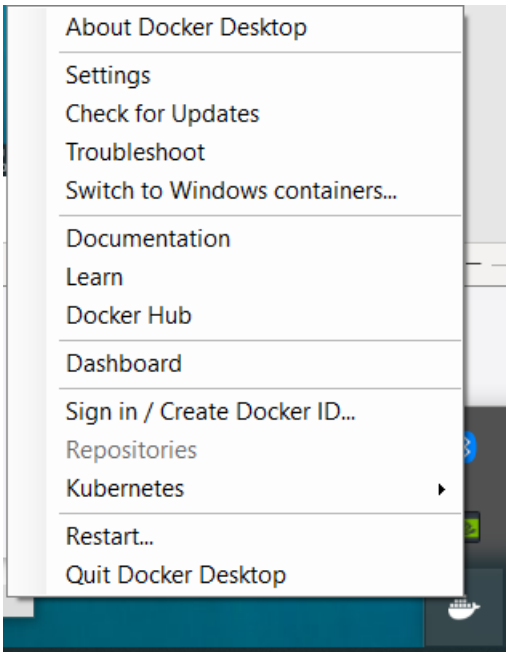
docker ps -a --filter label=io.x-k8s.kind.cluster=kind --format '{{.Names}}'



If you don’t see the little Docker whale in the system tray, you will have to restart your Windows VM or machine



After restart or if you have the docker whale in your system tray, we can open the dashboard. When you start Windows you should see notification from your System Tray that Docker for Hyper-V backend is starting and then in the system tray you should see our favorite containerized whale.



Next install **kubectl**.

Based on your operating system you could homebrew to install the kubectl or chocolatey for Windows.

https://kubernetes.io/docs/tasks/tools/install-kubectl/

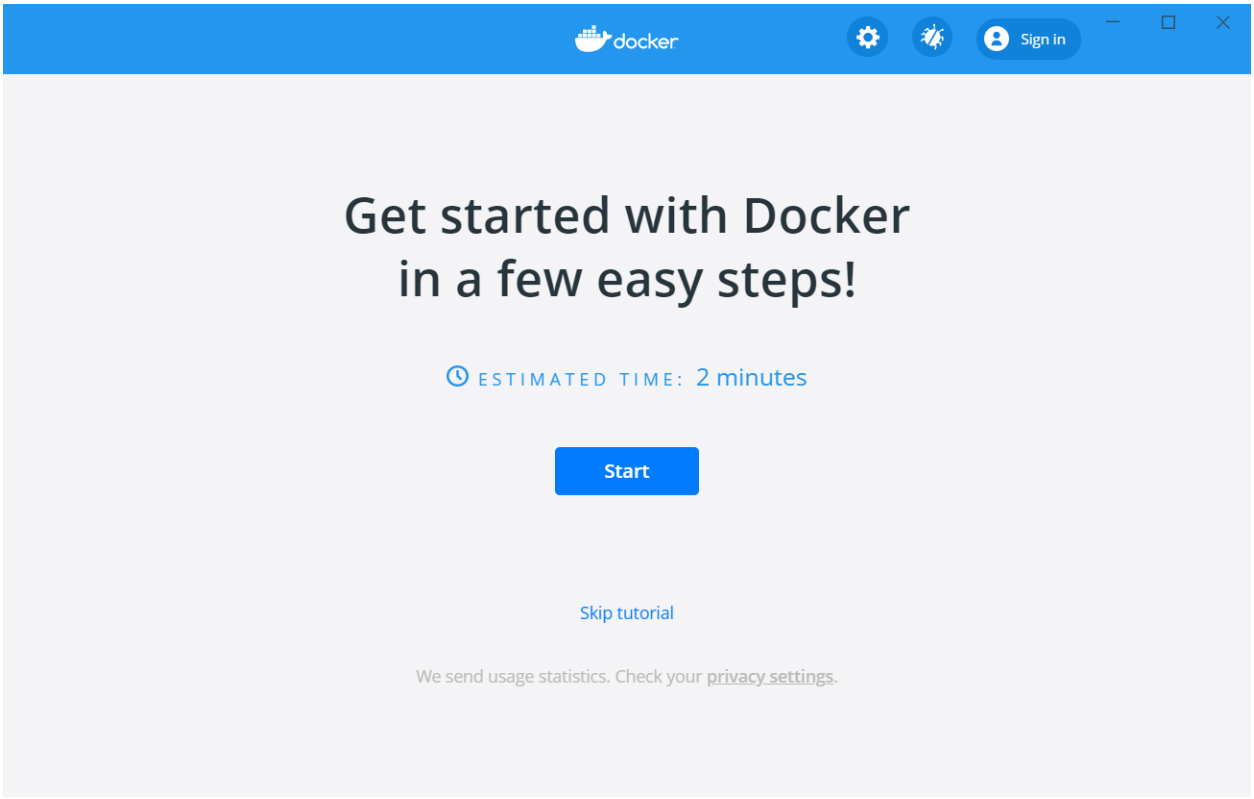
For Windows we're using Chocolatey for the installation

```
c:\> choco install kubernetes-cli
```

Verify your install

```
C:\>kubectl version --client
Client Version: version.Info{Major:"1", Minor:"19", GitVersion:"v1.19.0",
GitCommit:"e19964183377d0ec2052d1f1fa930c4d7575bd50", GitTreeState:"clean",
BuildDate:"2020-08-26T14:30:33Z", GoVersion:"go1.15", Compiler:"gc",
Platform:"windows/amd64"}
```

2. **Docker Validation via Tutorial (Docker for Desktop is fairly new for both Windows and MacOS)**



Take the tour, if this is your first time, or you're reinstalling. It's quick and painless.

Platform Note: The docker commands and just about every other command for docker, kubectl, k3d, and KIND in the experiments for our session are the same. The only platform specifics are tied to configuration and installation. Installs for Windows are done with Chocolatey, for MacOS with Brew and obviously there are differences in the way we set and reference environment variables between the platforms.

Windows:

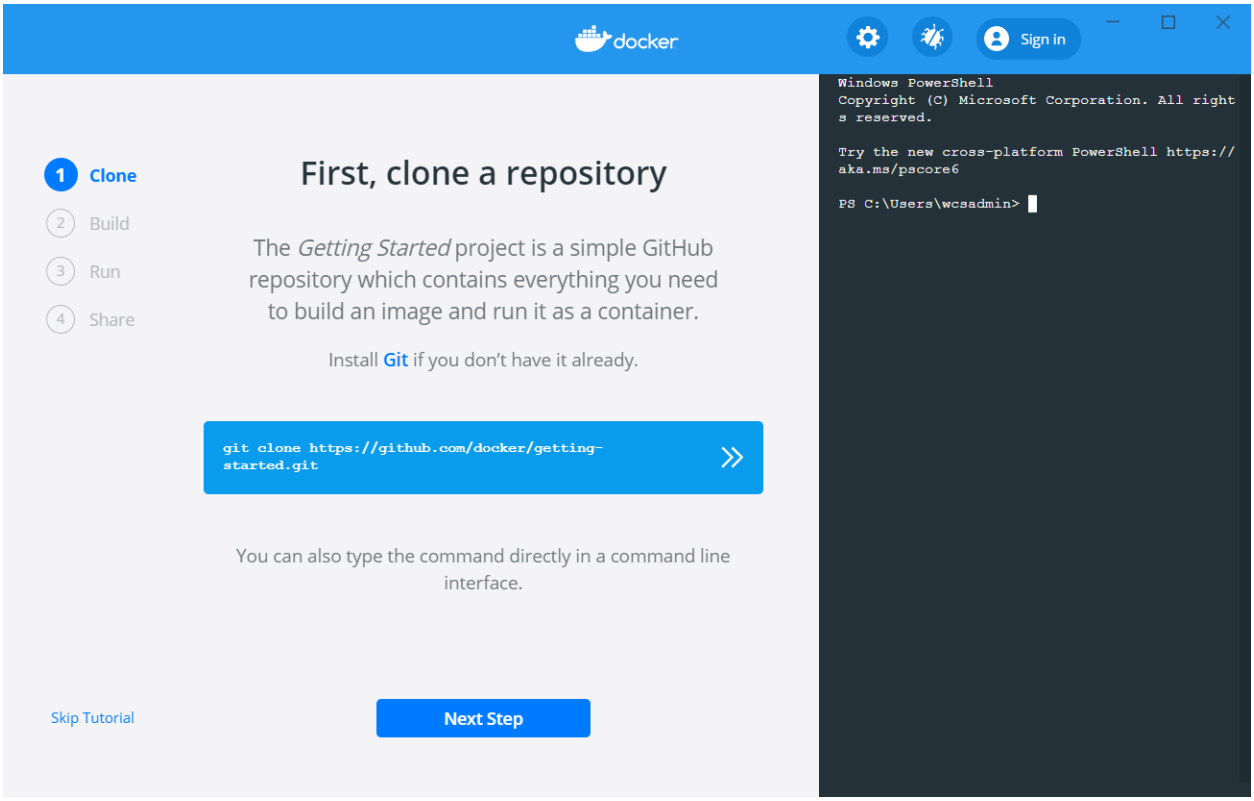
set ENVVAR-1=Some-value

type %ENVVAR-1%

MacOS

export ENVVAR-1=Some-value

cat $ENVVAR-1

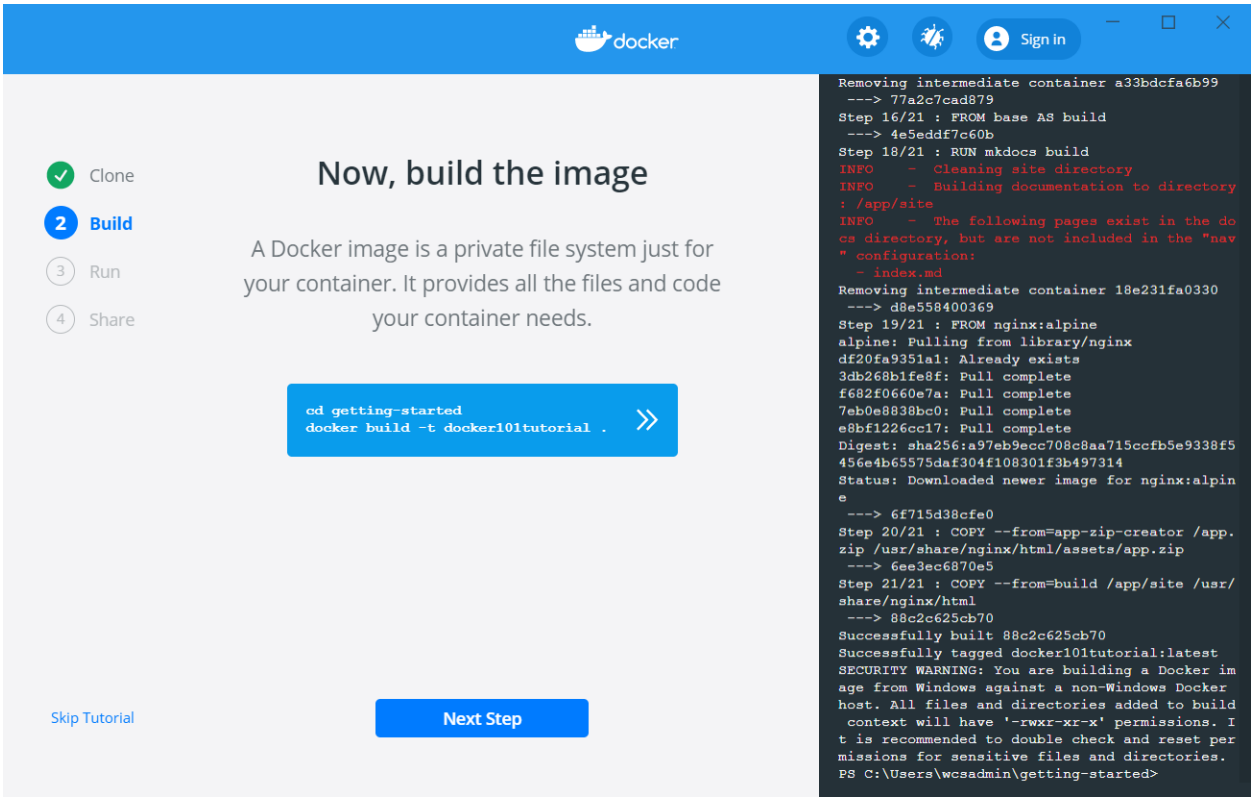


If you have Git installed, which is necessary for the class continue to click the Command Button and it will be pasted automatically (like Katacoda) into the Powershell window and clone the repo for the docker getting started.

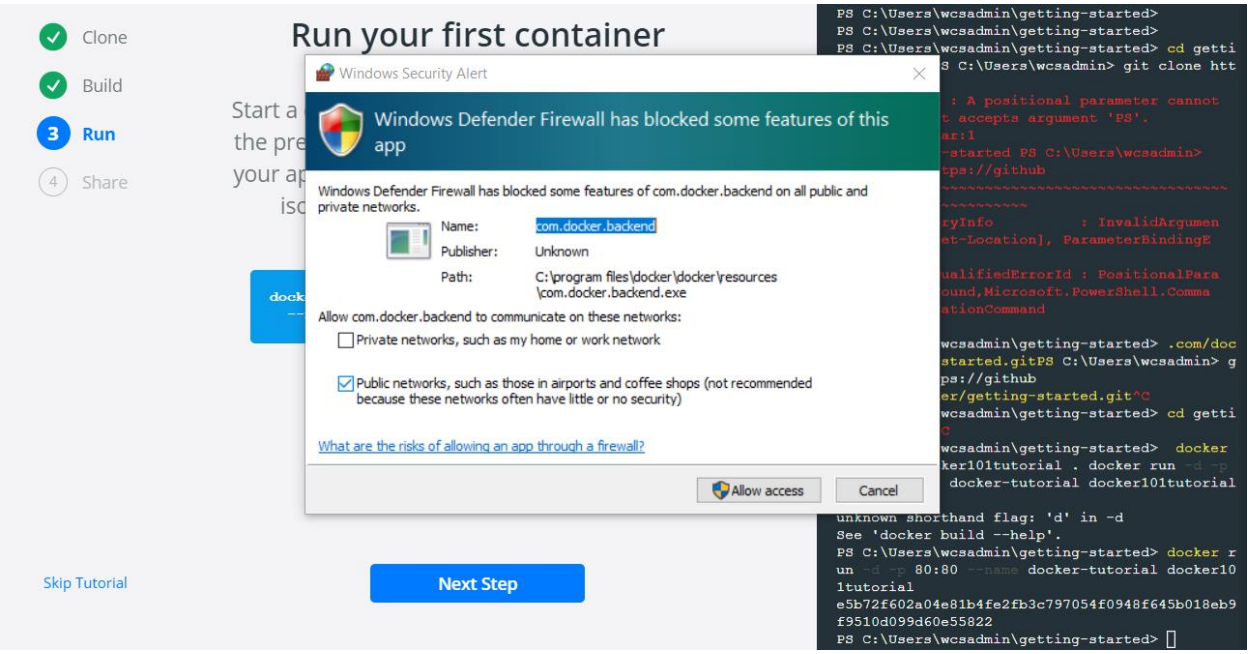
The next step is to change to the folder with the repo we just cloned and build a docker container

PS C:\Users\kubelord> **cd getting-started**

PS C:\Users\kubelord> **docker build -t docker101tutorial .**



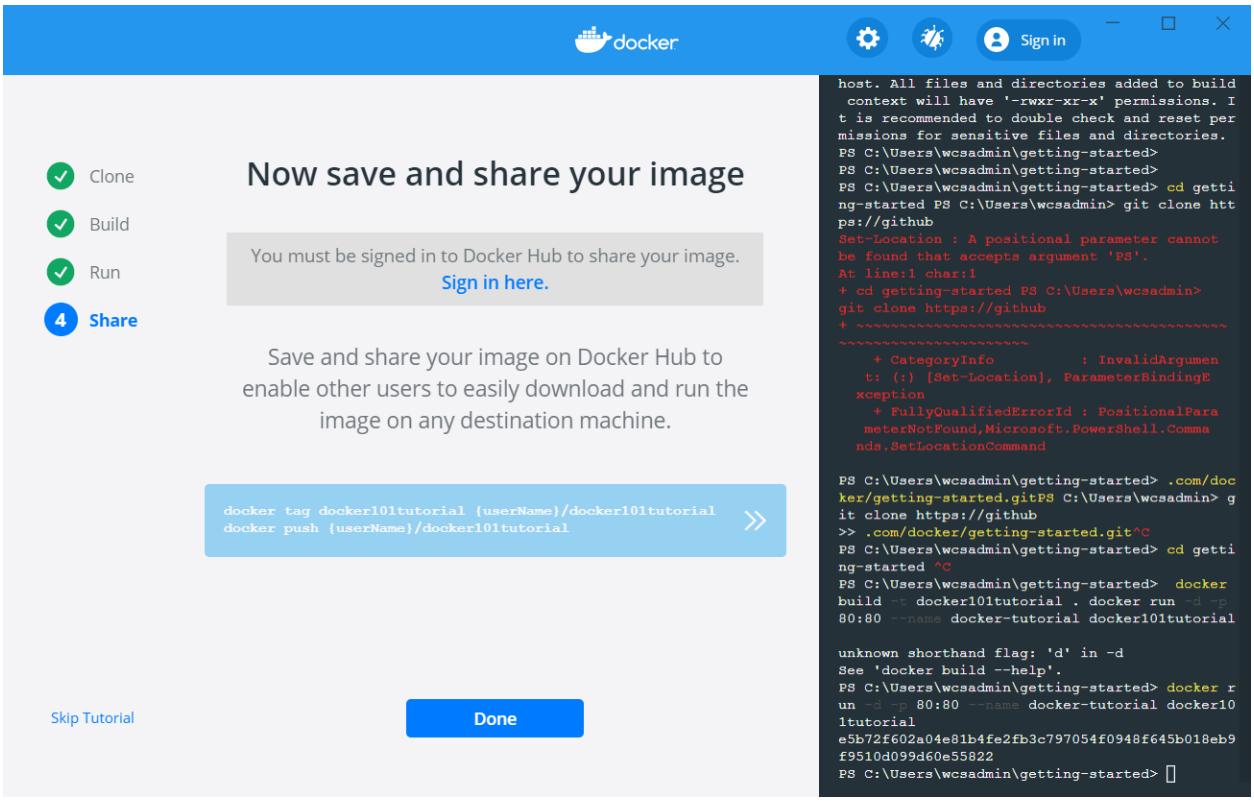
PS C:\Users\kubelord\getting-started> **docker run -d -p 80:80 --name docker-tutorial docker101tutorial**



Click “**Allow access**” to allow your container to be run

We should see the ID for our container returned. It will look similar to below

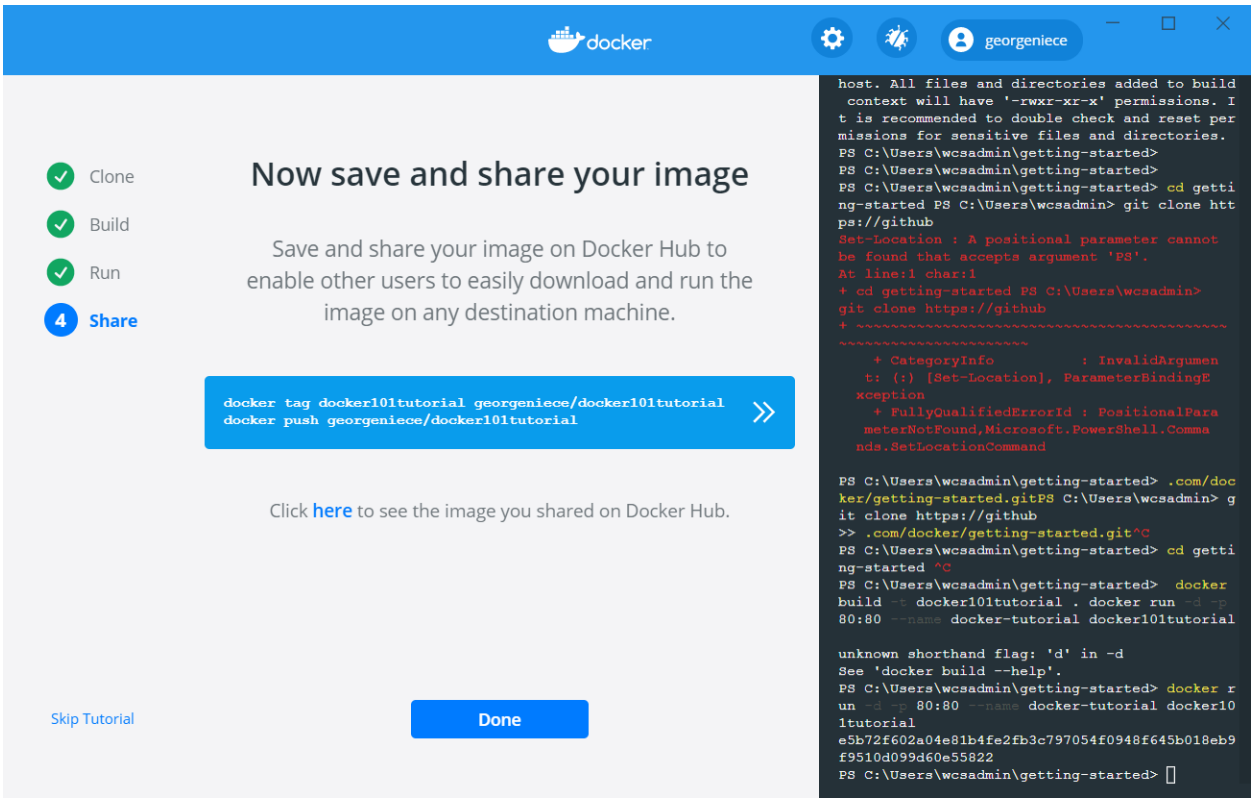
e5b72f602a04e81b4fe2fb3c797054f0948f645b018eb9f9510d099d60e55822



Sign into your Docker Hub account, if you don’t have one load hub.docker.com in a Web browser and create one.

3

That will update the screen for the next step in the quick start tutorial



Click the Command Button to tag our tutorial image and push to Docker Hub.

PS C:\Users\kubelord\getting-started> **docker tag docker101tutorial georgeniece/docker101tutorial**

PS C:\Users\kubelord\getting-started> **docker push georgeniece/docker101tutorial**

The push refers to repository [docker.io/georgeniece/docker101tutorial]

After the push completes we can check the status of the docker tutorial container we started with

PS C:\Users\kubelord\getting-started> **docker ps**

```
CONTAINER ID        IMAGE                  COMMAND                  CREATED
STATUS
PORTS               NAMES
e5b72f602a04        docker101tutorial   "/docker-entrypoint.…"   13 minutes ago      Up 13
minutes       0.0.0.0:80->80/tcp   docker-tutorial
```

PS C:\Users\kubelord\getting-started>

If we open a web browser to https://hub.docker.com/repositories we'll see the image under the repo we just created.