Integrated Vision System App Installation via Docker on Linux System

Prepared by: Yap Jun Kang

Contents

1. Docker Installation 3

1.1. Docker Engine 3

1.2. Docker Compose 4

2. NVidia Driver Installation 4

2.1. Nvidia Driver Installation through Terminal 4

2.2. Nvidia Driver Installation through GUI........ 5

2.3. Installation Debugging 6

3. Application Installation 7

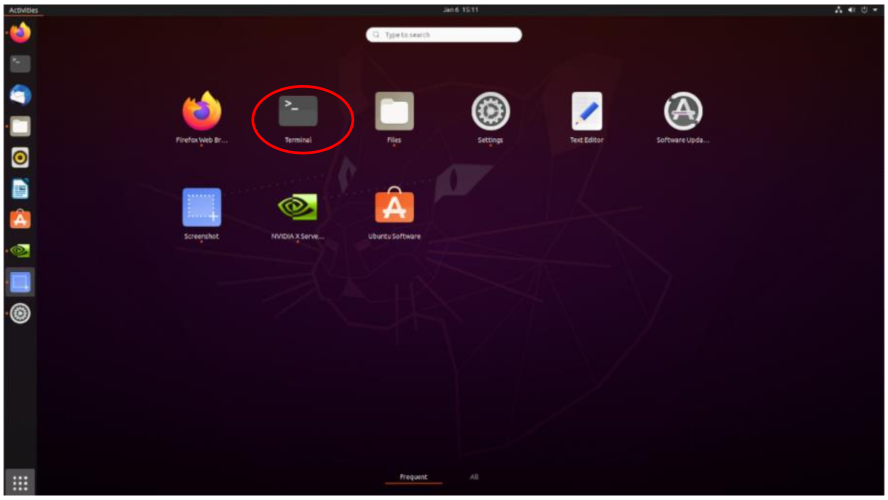
3.1. Docker Pull 7

3.2. Application Launch........ 9

1. Docker Installation

1.1. Docker Engine

1) Visit Docker hub and install docker.
2) Look for the correct system before installing docker, this document shows the tutorial for Ubuntu Bionic for Linux system.
3) Open terminal and run the command shown in the website.



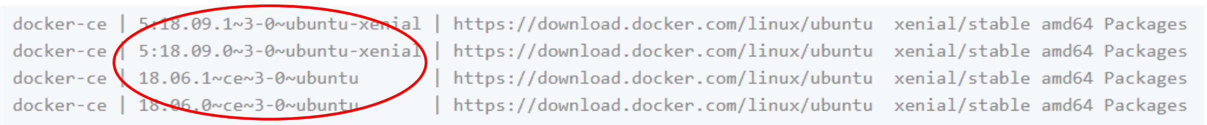
4) Installation will be done through repository here therefore, run the following command lines shown below.

```
$ sudo apt-get update
$ sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

5) The command above is to setup the repository.
6) Next is the command for installing docker

```
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
$ apt-cache madison docker-ce
```

7) After running the last line, this should show in the terminal, pick the desired version and copy the version name



```
docker-ce | 5:18.09.1~3-0~ubuntu-xenial | https://download.docker.com/linux/ubuntu  xenial/stable amd64 Packages
docker-ce | 5:18.09.0~3-0~ubuntu-xenial | https://download.docker.com/linux/ubuntu  xenial/stable amd64 Packages
docker-ce | 18.06.1~ce~3-0~ubuntu       | https://download.docker.com/linux/ubuntu  xenial/stable amd64 Packages
docker-ce | 18.06.0~ce~3-0~ubuntu       | https://download.docker.com/linux/ubuntu  xenial/stable amd64 Packages
```

8) Next run the following command in terminal and replace the **<VERSION_STRING>** with the copied version name.

```
$ sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING> containerd.io
```

9) Run the following command to test docker installation, run the command and wait, this should take about 10 to 15 seconds.

```
$ sudo docker run hello-world
```

1.2. Docker Compose

1) Visit this link to choose the correct OS for docker compose install. This tutorial show the Linux OS.
2) Run the following command to install docker compose. The version can be changed to allow other version installation. (i.e 1.29.2 to other version)

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

$ sudo chmod +x /usr/local/bin/docker-compose

$ docker-compose --version
```

3) The last line will show the installed version, this will indicate whether the installation is successful or not.

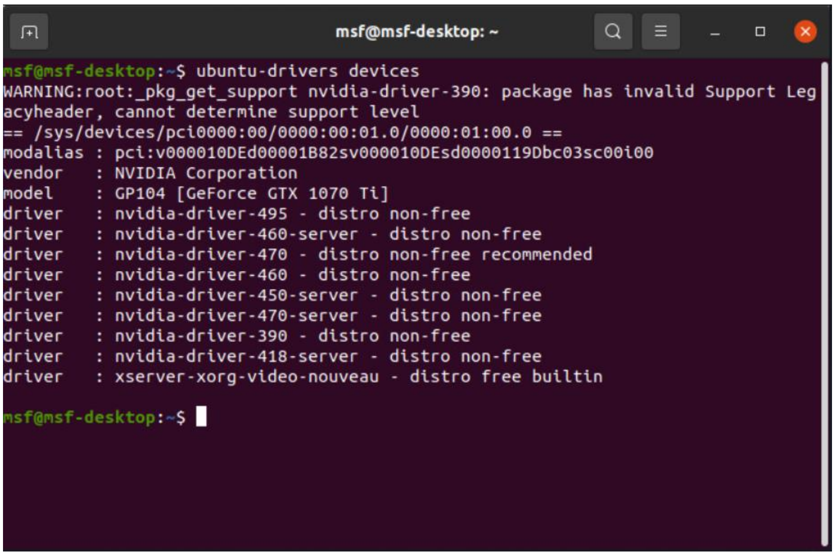
2. NVidia Driver Installation

2.1. Nvidia Driver Installation through Terminal

1) To install NVidia driver, consult this page for more information.
2) There are several ways for installing the drivers, but this tutorial will focus on installation through the terminal for Ubuntu Bionic.
3) First run the following command to check the available driver for install.

```
$ ubuntu-drivers devices
```

4) This will display all the available drivers to install



```
msf@msf-desktop:~$ ubuntu-drivers devices
WARNING:root:_pkg_get_support nvidia-driver-390: package has invalid Support Leg
acyheader, cannot determine support level
== /sys/devices/pci0000:00/0000:00:01.0/0000:01:00.0 ==
modalias : pci:v000010DEd00001B82sv000010DEsd0000119Dbc03sc00i00
vendor   : NVIDIA Corporation
model    : GP104 [GeForce GTX 1070 Ti]
driver   : nvidia-driver-495 - distro non-free
driver   : nvidia-driver-460-server - distro non-free
driver   : nvidia-driver-470 - distro non-free recommended
driver   : nvidia-driver-460 - distro non-free
driver   : nvidia-driver-450-server - distro non-free
driver   : nvidia-driver-470-server - distro non-free
driver   : nvidia-driver-390 - distro non-free
driver   : nvidia-driver-418-server - distro non-free
driver   : xserver-xorg-video-nouveau - distro free builtin

msf@msf-desktop:~$ 
```

5) If you are not sure which drivers to install, use the command for auto install

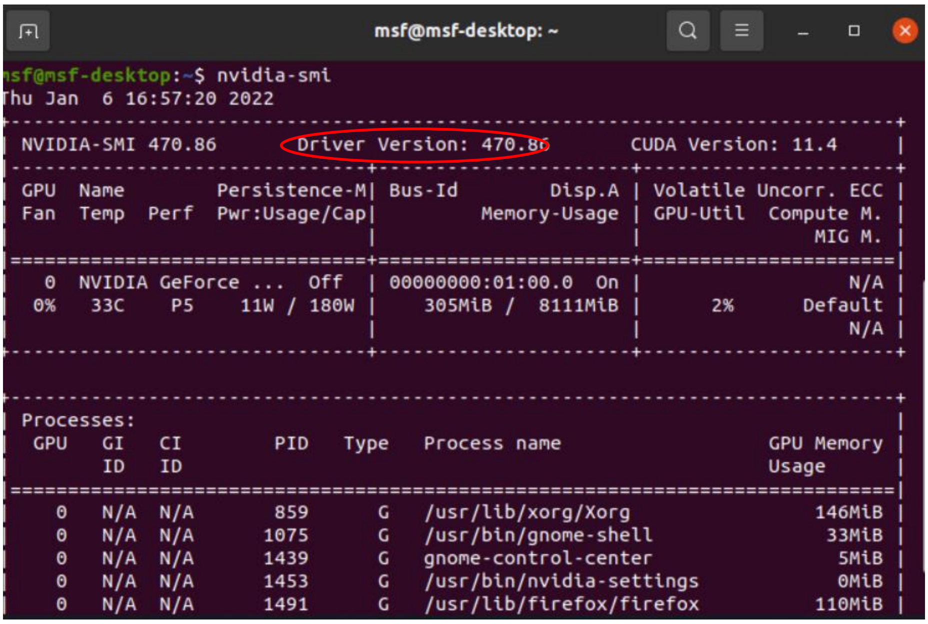
```
$ sudo ubuntu-drivers autoinstall
```

6) Otherwise, run the following command for the version to install by replacing the drivers shown in the previous command

WARNING: Do not install the third-party drivers unless it's the last choice as it can take very long to install and it may not even work

```
$ sudo apt install <driver version>
```

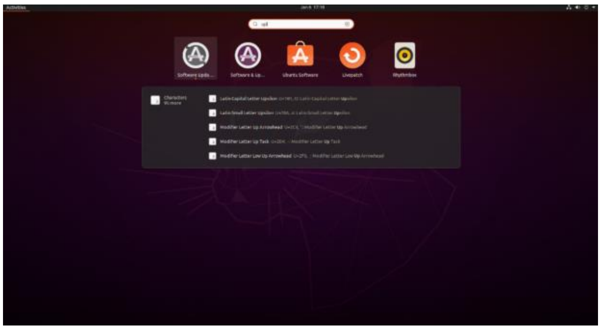
7) Once installation is done, reboot the PC and check the driver version using the command <nvidia-smi>

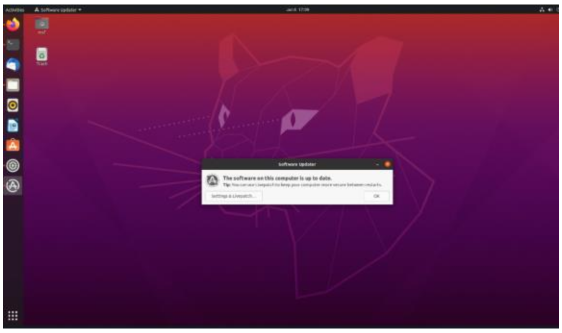


```
msf@msf-desktop:~$ nvidia-smi
Thu Jan  6 16:57:20 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.86       Driver Version: 470.86       CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ...  Off  | 00000000:01:00.0  On |                  N/A |
|  0%   33C    P5    11W / 180W |    305MiB /  8111MiB |      2%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A       859      G   /usr/lib/xorg/Xorg                146MiB |
|    0   N/A  N/A      1075      G   /usr/bin/gnome-shell               33MiB |
|    0   N/A  N/A      1439      G   gnome-control-center                5MiB |
|    0   N/A  N/A      1453      G   /usr/bin/nvidia-settings            0MiB |
|    0   N/A  N/A      1491      G   /usr/lib/firefox/firefox          110MiB |
```

2.2. Nvidia Driver Installation through GUI

1) Another method shown in the previous page is installation via the **Software Updater** and click settings.

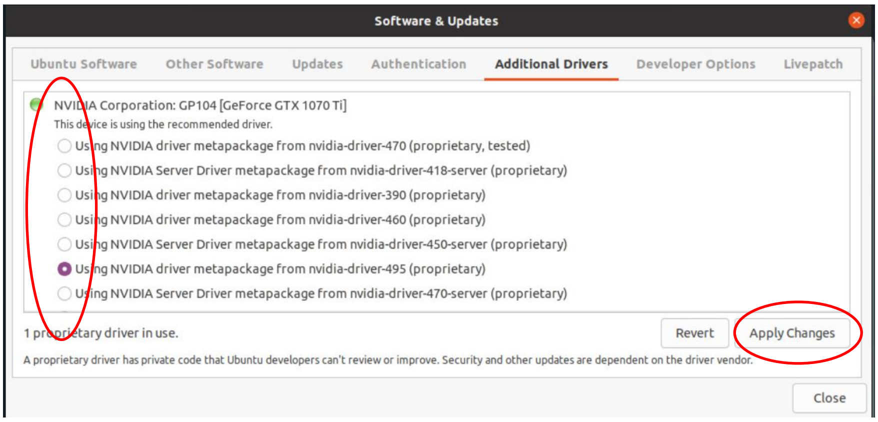




The software on this computer is up to date.

Tip: You can use Livepatch to keep your computer more secure between restarts.

2) Once open the settings, go to additional driver and choose the desired driver and apply the changes.
3) Once changes applied, reboot the system and check the driver version using the command <nvidia-smi>.



NVIDIA Corporation: GP104 [GeForce GTX 1070 Ti]
This device is using the recommended driver.

○ Using NVIDIA driver metapackage from nvidia-driver-470 (proprietary, tested)
○ Using NVIDIA Server Driver metapackage from nvidia-driver-418-server (proprietary)
○ Using NVIDIA driver metapackage from nvidia-driver-390 (proprietary)
○ Using NVIDIA driver metapackage from nvidia-driver-460 (proprietary)
○ Using NVIDIA Server Driver metapackage from nvidia-driver-450-server (proprietary)
● Using NVIDIA driver metapackage from nvidia-driver-495 (proprietary)
○ Using NVIDIA Server Driver metapackage from nvidia-driver-470-server (proprietary)

1 proprietary driver in use.

A proprietary driver has private code that Ubuntu developers can't review or improve. Security and other updates are dependent on the driver vendor.

2.3. Installation Debugging

1) Installation of Nvidia driver can be quite difficult on Linux, if installation is not successful, follow the following instruction for debugging.
2) First, open up **Software Updater** and make sure your PC is up to date.
3) Next, try running the following command to make sure everything is properly updated.

```
$ sudo apt update
$ sudo apt upgrade
```

4) If when installing, the error message shows 'Unable to install due to broken packages' try the following.
5) First, remove all Nvidia driver files by using the following command,

```
$ sudo apt-get --purge remove "*nvidia*"
```

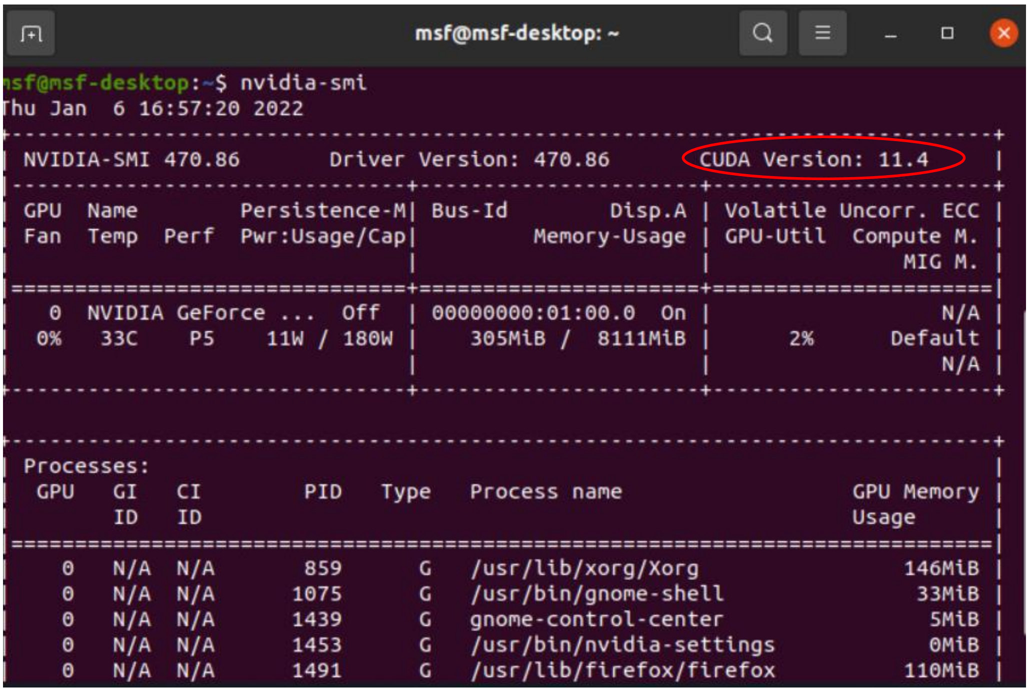
6) Next, install Nvidia runtime container to ensure docker will be able to detect the gpu, following this section.
7) If none of the suggestion above, try to reboot and re-install Linux OS.
8) Make sure to use this as last resort and always backup everything first.

3. Application Installation

3.1. Docker Pull

1) First enable the GPU support for docker so that the app can utilized and accelerator rather than just the base CPU.
2) Go to this link and start installing Nvidia Container Toolkit, make sure the necessary Nvidia driver has been installed first.
3) Run the following commands to install Nvidia Container Toolkit

```
$ distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \

   && curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - \

   && curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list |
sudo tee /etc/apt/sources.list.d/nvidia-docker.list

$ sudo apt-get update

$ sudo apt-get install -y nvidia-docker2

$ sudo systemctl restart docker

$ nvidia-smi  (for checking if cuda exist now)
```



```
sf@msf-desktop:~$ nvidia-smi
hu Jan  6 16:57:20 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.86       Driver Version: 470.86       CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ...  Off  | 00000000:01:00.0  On |                  N/A |
|  0%   33C    P5    11W / 180W |    305MiB /  8111MiB |      2%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A       859      G   /usr/lib/xorg/Xorg                146MiB |
|    0   N/A  N/A      1075      G   /usr/bin/gnome-shell               33MiB |
|    0   N/A  N/A      1439      G   gnome-control-center                5MiB |
|    0   N/A  N/A      1453      G   /usr/bin/nvidia-settings            0MiB |
|    0   N/A  N/A      1491      G   /usr/lib/firefox/firefox          110MiB |
```

4) The GPU should be enabled after this, next clone the repository from GitHub using the following command in the terminal.

```
$ git clone https://github.com/msf4-0/Integrated-Vision-Inspection-System.git
```

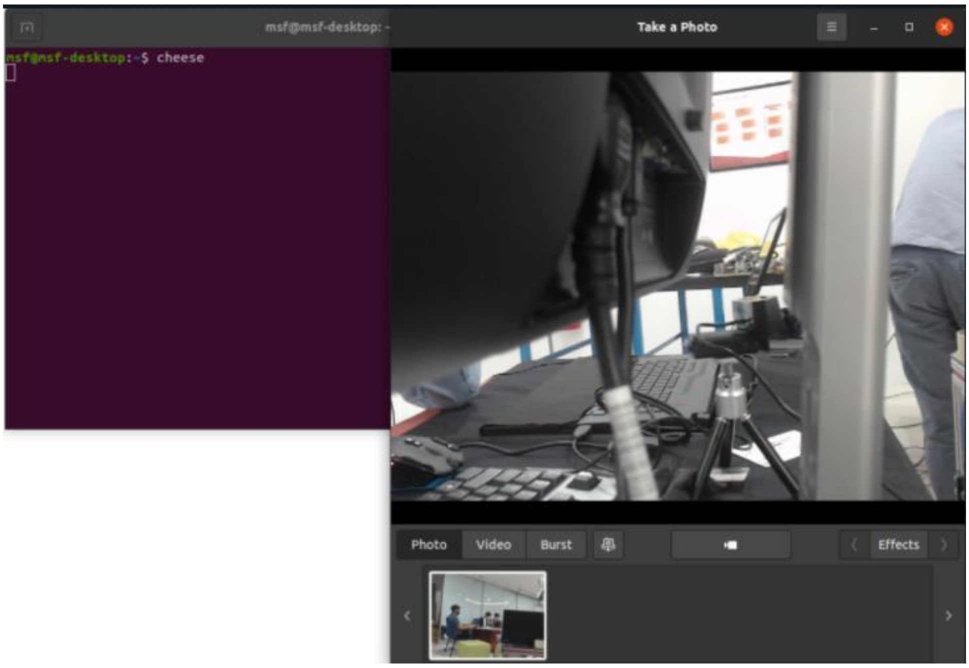
5) Next change the directory to the folder pulled from the repository.

```
$ cd Integrated-Vision-Inspection-System
```

6) Go to the file folder <Integrated-Vision-Inspection-System> and look for the file named <.env.sample> and rename it to <.env>

NOTE: The env.sample file will be hidden by default, click on the hamburger menu on top right of the file explorer to check 'show hidden files' to display it.

7) Next attach the USB cameras and open a new terminal using Ctrl-Alt-T and type <cheese> to test the camera.



```
msf@msf-desktop:~$ cheese
```

8) Now check the device name allocated by the PC using the following command.

```
$ sudo apt-get install v4l-utils
$ v4l2-ctl --list-devices
```

9) Once the devices names have been obtained, open the file folder <Integrated-Vision-Inspection-System> and look for the file named <docker-compose.yml> or <docker-compose.cpu.yml> depending on whether GPU is supported.

10) Look for devices under app and paste in the device name as such,

```
devices:
  - "/dev/camera_name:/dev/camera_name"
  - "/dev/camera_name2:/dev/camera_name2"
```

NOTE: Under app there will be 2 device section, be sure not to include the one with capabilities [gpu]

11) Next, pull the necessary images and run the app using the docker compose file pulled from the repository.

```
$ sudo docker-compose up --build
```

12) OR run the following command if GPU support has not been enabled.

```
$ sudo docker-compose up -f docker-compose.cpu.yml up --build
```

13) The above commands should always be run first after every time the docker compose file was modified.
14) Once the installation is complete, the command can be modified to below, for faster launch time.

```
$ sudo docker-compose up -d

$ sudo docker-compose up -f docker-compose.cpu.yml up -d
```

NOTE: It is best to leave out the -d so that we can see if any error messages exist.

15) The first installation may take up quite some time. Be sure not to close the terminal before the installation is done.

3.2. Application Launch

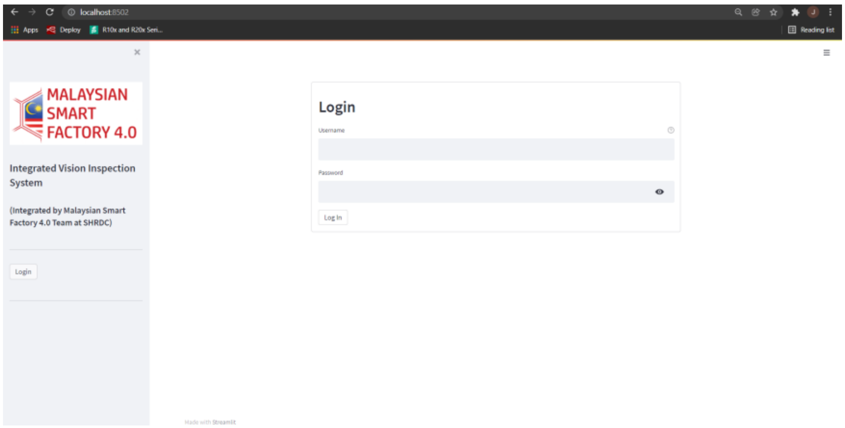
1) If the installation shows that capabilities[gpu] is not supported, follow the following steps to install Nvidia runtime container, this is necessary to expose the GPU driver to docker.
2) Refer to this link for more detailed info.
3) First, open up a text file and paste the following lines and save the file with the name < nvidia-container-runtime-script.sh>

```
curl -s -L https://nvidia.github.io/nvidia-container-runtime/gpgkey | \ sudo apt-
key add - distribution=$(. /etc/os-release;echo $ID$VERSION_ID) curl -s -L
https://nvidia.github.io/nvidia-container-runtime/$distribution/nvidia-container-
runtime.list | \ sudo tee /etc/apt/sources.list.d/nvidia-container-runtime.list
sudo apt-get update
```

4) Run the following commands to install Nvidia runtime container.

```
$ cat nvidia-container-runtime-script.sh

$ sh nvidia-container-runtime-script.sh

$ apt-get install nvidia-container-runtime
```

5) Once installation is done, launch the app again and docker will be able to use GPU now.
6) Once the docker installation is successful, open a web browser and navigate to the link <**localhost:8502**>
7) The app will be launch if the installation is successful.
8) The default username and password will be <admin>



Integrated Vision Inspection System

(Integrated by Malaysian Smart Factory 4.0 Team at SHRDC)

9) To stop the app, use the command <docker-compose down>