For personal:  
  
sudo snap install code –classic

sudo snap install spotify

sudo apt install vlc

For setup:

sudo apt update

sudo apt upgrade

Install nvidia drivers

sudo ubuntu-drivers install

reboot

Install cuda: <https://developer.nvidia.com/cuda-downloads?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=22.04&target_type=deb_network>

wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86\_64/cuda-keyring\_1.1-1\_all.deb

sudo dpkg -i cuda-keyring\_1.1-1\_all.deb

sudo apt update

sudo apt-get -y install cuda-toolkit-12-4

Install open kernel flavour (legacy will provide errors):

sudo apt-get install -y nvidia-driver-550-open

sudo apt-get install -y cuda-drivers-550

Install cuDNN:  
<https://developer.nvidia.com/cudnn-downloads?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=22.04&target_type=deb_network>  
  
wget <https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/cuda-keyring_1.1-1_all.deb>  
sudo dpkg -i cuda-keyring\_1.1-1\_all.deb

sudo apt-get update

sudo apt-get -y install cudnn

sudo apt-get -y install cudnn-cuda-12

Ensures all cuda dependencies are present  
sudo apt install cuda

Install tensorflow:

Install tensorflow:<https://www.tensorflow.org/install/pip>

pip install --upgrade pip

pip install tensorflow[and-cuda]

This may be needed in bashsrc file:  
  
export CUDNN\_PATH="$HOME/.local/lib/python3.10/site-packages/nvidia/cudnn"

export LD\_LIBRARY\_PATH="$CUDNN\_PATH/lib":"/usr/local/cuda/lib64"

export PATH="$PATH":"/usr/local/cuda/bin"  
  
Testing in vscode means its not required:  
  
python3 -c "import tensorflow as tf; print(tf.config.list\_physical\_devices('GPU'))"

Install git

sudo apt install git

Configure git

git config --global user.email “email”

git config --global user.name “name”

Clone git

git clone: <https://github.com/ConnorSouthEngineering/FYP>

Install docker: <https://docs.docker.com/engine/install/ubuntu/>

Setup pre-requisites:

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update  
  
Install docker:  
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

Test:  
sudo docker run hello-world

Allow for non sudo:<https://docs.docker.com/engine/install/linux-postinstall/>

(should already exist but do first line anyway)

sudo groupadd docker

sudo usermod -aG docker $USER  
newgrp docker

Test without sudo:  
docker run hello-world

Open vscode with docker plugin - containers should now be visible, if not restart pc and try again

Install nvidia container toolkit:https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/latest/install-guide.html

Add repository:

curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \

&& curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \

sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' | \

sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list

sudo apt-get update

Install container toolkit:  
  
sudo apt-get install -y nvidia-container-toolkit

Configure docker:  
  
sudo nvidia-ctk runtime configure --runtime=docker

sudo systemctl restart docker

Configure rootless docker:

nvidia-ctk runtime configure --runtime=docker --config=$HOME/.config/docker/daemon.json

systemctl --user restart docker

sudo nvidia-ctk config --set nvidia-container-cli.no-cgroups --in-place

Allow for execution (without this NVML Unknown error occurs):

sudo gedit /etc/nvidia-container-runtime/config.toml

no-cgroups = false

Install NVM (node version manager):

wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.1/install.sh | bash

export NVM\_DIR="$([ -z "${XDG\_CONFIG\_HOME-}" ] && printf %s "${HOME}/.nvm" || printf %s "${XDG\_CONFIG\_HOME}/nvm")"

[ -s "$NVM\_DIR/nvm.sh" ] && \. "$NVM\_DIR/nvm.sh"

Install node version:  
  
nvm install 16

Install angular 16:

nvm use 16

npm install @angular/cli@16 - -global

Add NG to bashrc

sudo gedit ~/.bashrc

Append with following line:  
  
alias ng="/home/connor/.nvm/versions/node/v16.20.2/lib/node\_modules/@angular/cli/bin/ng.js"

Install project dependencies from package.json  
  
First directories must change to user ownership  
cd FYP  
sudo chown -R $USER:$(id -gn $USER) .  
  
Then navigate to each directory (OVision and VisionLinkAPI) and run   
  
npm install   
  
Fix issues with

npm audit fix

Create custom docker network for master-branch to sit on:  
docker network create master-bridge

launch with docker compose up  
  
Record ips via inspecting each container (these shouldnt change)

create config file of format:  
{"allowedIps":["::ffff:192.168.0.160","::ffff:172.19.0.0","::ffff:172.19.0.1","::ffff:172.19.0.2","::ffff:172.19.0.3","::ffff:172.19.0.4"]}  
  
All IPs sourced from inspection should be provided with prefix of ::ffff as its implicitly converts all ipv4s to ipv6 format for some reason   
  
Create network partition for model-repo on nodes:  
  
sudo apt-get update

sudo apt-get install nfs-kernel-server

Generate key for node  
  
docker compose up  
  
Wait for a minute or two  
  
localhost:3000/nodes/key/generate

NVIDIA JETSON SETUP

sudo apt update

download vscode:

https://update.code.visualstudio.com/1.84.2/linux-deb-arm64/stable

sudo apt install python3-pip

pip3 install setuptools

pip3 install –update pip

sudo pip3 install -U jetson-stats

pip3 install asyncio

pip3 install aiocsv

pip3 install aiofiles