1. Install Debian with ssh access (during advanced installation)
    a. Disable apt packages from CD
        i. **vim /etc/apt/sources.list**
            1. Comment out cdrom entries
2. If ssh access isn't configured during installation:
    a. **apt install openssh-server**
3. Install vim
    a. **apt install vim**
4. Configure default editor as vim
    a. Export EDITOR="/usr/bin/vim"
5. Add user to sudoers file using visudo
6. Install xrdp
    a. **apt install xfce4 xfce4-goodies xorg dbus-x11 x11-xserver-utils**
    b. **apt install xrdp**
7. Configure xrdp
    a. Stop xrdp
        i. **systemctl stop xrdp**
    b. Modify /etc/xrdp/startwm.sh
        i. Comment out:
            test -x /etc/X11/Xsession && exec /etc/X11/Xsession
            exec /bin/sh /etc/X11/Xsession
        ii. Add:
            startxfce4
    c. start xrdp
        i. **systemctl start xrdp**
8. Install locate
    a. **apt install locate**
9. set up alias
    a. Modify user & root's .bashrc file
        i. vim /root/.bashrc & **vim /<usr>/.bashrc**
            1. Uncomment alias lines and/or create new ones
    b. Re-read bashrc for both user and root
        i. source ~/.bashrc
10. Disable sleep & hibernation:
    a. sudo systemctl mask sleep.target suspend.target hibernate.target hybrid-sleep.target
11. Install and configure NFS
    a. apt install nfs-common
    b. apt-get install cifs-utils
    c. Configure mount point by adding the below entry to the fstab:
        i. vim /etc/fstab

    //*NAS_IP/MOUNTPOINT*/media/NAS cifs
    username=*customUsername,*password=*customPassword*,iocharset=utf8 0 0

    d. make directory /media/NAS

i. **mkdir /media/NAS**

e. Mount the share

i. **Mount -a**

12. Install Docker Engine on Debian

a. Uninstall all conflicting packages

i. **for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get remove $pkg; done**

b. Set up the repository

i. **sudo apt-get update**

ii. **sudo apt-get install ca-certificates curl gnupg**

c. Add Docker's official GPG key:

i. **sudo install -m 0755 -d /etc/apt/keyrings**

ii. **curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg**

iii. **sudo chmod a+r /etc/apt/keyrings/docker.gpg**

d. Set up the repository:

i. **echo **

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

**"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | **

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

e. Install Docker Engine:

i. **sudo apt-get update**

ii. **sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin**

iii. **apt install docker-compose**

f. Verify operation:

i. **sudo docker run hello-world**

1. if errors out due to version is unsupported error, obtain the latest docker-compose

a. **sudo apt-get remove docker-compose**

b. **curl -O -J -L https://github.com/docker/compose/releases/download/v2.11.2/docker-compose-linux-x86_64**

c. **chmod +x docker-compose-linux-x86_64**

d. **sudo cp ./docker-compose-linux-x86_64 /usr/bin/docker-compose**

2. Check the version

a. **docker-compose version**

13. Install coral drivers

a. Install Debian package repo:

i. **echo "deb https://packages.cloud.google.com/apt coral-edgetpu-stable main" | sudo tee /etc/apt/sources.list.d/coral-edgetpu.list**
ii. **curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -**
iii. **sudo apt-get update**

b. Install PCIe driver and Edge TPU runtime packages
i. **sudo apt-get install gasket-dkms libedgetpu1-std**

c. Add udev rules
i. **sudo sh -c "echo 'SUBSYSTEM==\"apex\", MODE=\"0660\", GROUP=\"apex\"' >> /etc/udev/rules.d/65-apex.rules"**
ii. **sudo groupadd apex**
iii. **sudo adduser $USER apex**

d. Reboot

14. Install the pycoral libraries
a. sudo apt-get install python3-pycoral
b. sudo apt install pip

15. Run a model on the Edge TPU
a. Download example from github
i. **mkdir coral && cd coral**
ii. **git clone https://github.com/google-coral/pycoral.git**
iii. **cd pycoral**

b. Download the model, labels, and bird photo
i. **bash examples/install_requirements.sh classify_image.py**

c. Run the image classifier with the bird photo
i. **python3 examples/classify_image.py **
**--model test_data/mobilenet_v2_1.0_224_inat_bird_quant_edgetpu.tflite **
**--labels test_data/inat_bird_labels.txt **
--input test_data/parrot.jpg

a. Verify the USB module is detected
d. lsusb | grep Google

16. Create config.yml file (see notepad++)

```
mqtt:
  enabled: false

    #database:
    #  path: /db/frigate.db

detectors:
  coral:
    type: edgetpu
    device: usb

birdseye:
  enabled: True
  mode: continuous
```

```
cameras:
  family:
    ffmpeg:
      inputs:
        - path:
rtsp://customUsername:CustomPassword@cameraIP/cam/realmonitor?channel=1&sub
type=0
          roles:
            - record
            - detect
      hwaccel_args: preset-vaapi
    detect:
      enabled: True
      width: 1920
      height: 1080
    objects:
      track:
        - person
        - dog
        - cat
        - car
        - motorcycle
        - bus
        - bird
    record:
      enabled: True
```

17. Create frigate.yml file

```
version: "3.9"
services:
  frigate:
    container_name: frigate
    privileged: true # this may not be necessary for all setups
    restart: unless-stopped
    image: ghcr.io/blakeblackshear/frigate:stable
    shm_size: "256mb" # update for your cameras based on calculation above
    devices:
      - /dev/bus/usb:/dev/bus/usb # passes the USB Coral, needs to be modified for other
versions
      - /dev/apex_0:/dev/apex_0 # passes a PCIe Coral, follow driver instructions here
https://coral.ai/docs/m2/get-started/#2a-on-linux
      - /dev/dri/renderD128 # for intel hwaccel, needs to be updated for your hardware
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - /home/frigate/config/config.yml:/config/config.yml
      - /home/frigate:/db:rw
#     - /home/frigate:/media/frigate
```

```
      - /media/NAS:/media/frigate:rw
      - type: tmpfs # Optional: 1GB of memory, reduces SSD/SD Card wear
        target: /tmp/cache
        tmpfs:
          size: 1000000000
    ports:
      - "5000:5000"
      - "1935:1935" # RTMP feeds
      - "8554:8554" # RTSP feeds
      - "8555:8555/tcp" #WebRTC over TCP
      - "8555:8555/udp" #WebRTC over UDP
    environment:
FRIGATE_RTSP_PASSWORD: "password"
```