Steps followed to build the Yocto & docker

1. Clone and sync the required Linux Distro repo

```
$ repo init -u https://source.codeaurora.org/external/imx/imx-manifest -b imx-linux-hardknott -m imx-5.10.35-2.0.0_docker.xml
```

```
$ repo sync
```

2. Replace busybox .bb files

```
$ git clone git://git.yoctoproject.org/poky -b hardknott
```

```
$ rm -rf  L5.10.35_2.0.0/sources/poky/meta/recipes-core/busybox
```

```
$ cp -r poky/meta/recipes-core/busybox  L5.10.35_2.0.0/sources/poky/meta/recipes-core
```

Because sources/meta-virtualization/recipes-core/busybox/ busybox-initrd_1.33.1.bb require busybox-1.33.1.bb, but yocto L5.10.35_2.0.0 branch is busybox-1.33.1.bb, if not replaced the files, will meet below error.

ERROR: ParseError at /data1/Harish/imx-yocto-bsp-docker/sources/meta-virtualization/recipes-core/busybox/busybox-initrd_1.33.1.bb:3: Could not include required file recipes-core/busybox/busybox_1.33.1.bb

3. Get the build config files with the required board model and DISTRO version

```
$ DISTRO=fsl-imx-wayland MACHINE=imx8mpevk source imx-setup-release.sh -b docker_image
```

4. Configure the Yocto layers to enable virtualization by updating the bblayers.conf file

```
$ echo "BBLAYERS += \" \${BSPDIR}/sources/meta-virtualization \"" >> conf/bblayers.conf
```

5. Add docker to the image in conf/local.conf

```
$ DISTRO_FEATURES_append = " virtualization"
$ IMAGE_INSTALL_append = " docker"
```

6. Build the Image using bitbake command.

```
$ bitbake <image-name>
```

Some image recipes:
imx-image-core - core image with basic graphics and no multimedia
imx-image-multimedia - image with multimedia and graphics
imx-image-full - image with multimedia and machine learning and Qt

You can also refer to the ReadMe to build L5.10.52_2.1.0_docker

https://source.codeaurora.org/external/imx/imx-manifest/tree/README-docker?h=imx-linux-hardknott

Getting Started with Docker

i.MX Board Terminal

1. Start pulling the basic GNU/Linux Ubuntu image

```
$ docker pull ubuntu:21.04
```

2. Launch a basic Docker image

```
$ docker run -it --privileged=true -v /usr/lib/:/usr/local/lib/aarch64-linux-gnu/ --network=host  ubuntu:21.04
```

3. Close the Docker container and get its id

```
$ exit
```

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
ed4de41c0e3c	ubuntu:21.04	"bash"	Less than a second ago	Exited (255) 1 second ago		

4. Copy the *tensorflow-lite bin* to Docker container

```
$ docker cp /usr/bin/tensorflow-lite-2.4.1/examples/   <my_container_id>:/root
```

5. Install python3 in docker

```
$ mkdir bin
```

```
$ cp /usr/bin/python3*  bin/
```

```
$ docker cp bin/   ed4de41c0e3c:/usr
```

6. To start the Container again, run the following commands:

```
$ docker start <my_container_id>
$ docker attach <my_container_id>
```

7.Set environment variables in docker

```
$ ln -s /usr/local/lib/aarch64-linux-gnu/python3.9/  /usr/local/lib/python3.9
```

```
$ export PYTHONHOME=/usr/local
```

```
$ LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib/aarch64-linux-gnu
```

8. Run the label_image

```
$ cd /root/example
```

```
$ python3 label_image.py
```

```
root@imx8mpevk:~/examples# python3 label_image.py
INFO: Created TensorFlow Lite delegate for NNAPI.
Applied NNAPI delegate.
Warm-up time: 6353.8 ms

Inference time: 3.2 ms

0.870588: military uniform
0.031373: Windsor tie
0.011765: mortarboard
0.007843: bow tie
0.007843: bulletproof vest
```