## Sample structure of Dockerfile

`FROM ubuntu:18.04`<br/>

`COPY . /app`<br/>

`ENV PATH "$PATH:/app"`<br/>

`RUN make /app`<br/>

`WORKDIR /data` <br>

### 1. Set the base image by using FROM<br>

`FROM <base image name>` <br/>

ex: FROM ubuntu:18.04

Specifies which base image to use. It sets the base environment of the docker image, such an OS, a framework or an application to be used for our application. Base images such as Ubuntu, Node.js, Java, etc. are available from the docker hub. (https://hub.docker.com/)

### 2. Use COPY to import the application files

`COPY <source directory> <destination directory inside docker image>` <br/>

ex: COPY . /opt/toppic

Use COPY command to specify a local directory that contains the files needed to run the application, and the files inside it are copied into the docker image. If the directory does not exist in docker image, it is going to be created.

### 3. Set toppic path as environmental variable ENV

`ENV PATH "$PATH:<application path>"`<br/>

ex: ENV PATH "$PATH:/opt/toppic/bin"

Add the toppic suite application path to the environmental variable so that the users can just enter the program name to run it.

### 4. Install libraries using RUN

`RUN <command>` <br/>

ex: RUN apt-get install zlib1g-dev -y\

libboost-filesystem-dev -y\

It is faster to install multiple libraries in one RUN command, rather than doing it separately. Use \ to separate each libraries.

### 5. Set the current directory (work directory) for startup

`WORKDIR <dir>` <br/>

ex: WORKDIR /data

Set the directory inside docker image that is going to be the current directory when the program runs.