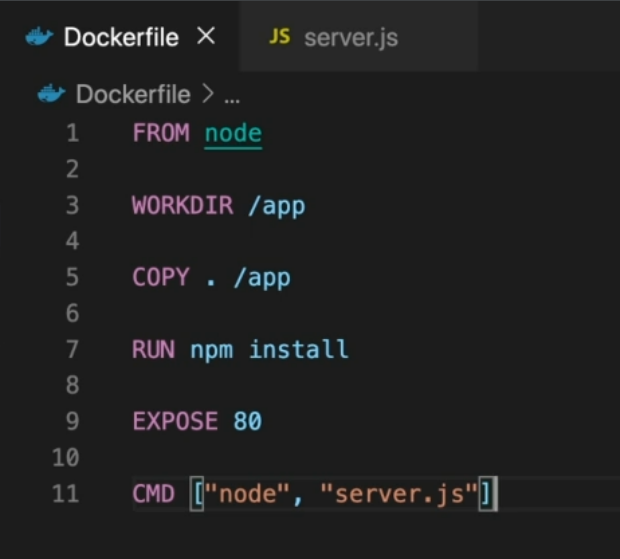
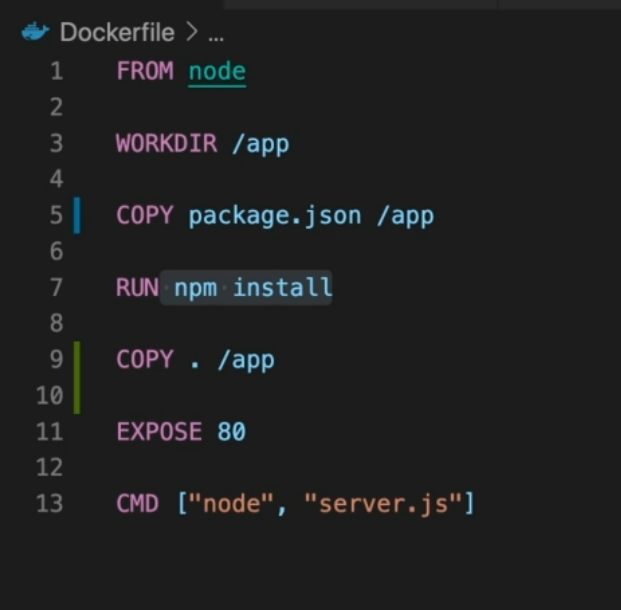
**Understanding Image Layers**

Whenever you build an image, Docker caches every instruction result, and when you then rebuild an image, it will use these ***cached*** results if there is no need to run an instruction again. And this is called a layer-based architecture. Every instruction represents a layer in your Dockerfile. And an image is simply built up from multiple layers based on these different instructions. In addition, an image is read only, which means once an instruction has been executed and once the image is built, the image is locked in and code in there can't change unless you rebuild the image, which technically means you create a new image.



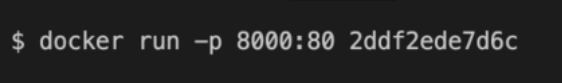
Now it also means that at the moment, whenever we change anything in our code, we also run npm install again, even though we as a developer know that this is unnecessary.

And here we have our first tiny bit of optimization potential for this Dockerfile.



This will run the container in attached mode by default, means the process will run in the foreground where we will be able to see the events in the terminal itself.

With that, we would pick up this package.json file, copy that into the app folder, then run npm install, and then copy over our other code. With this, we would ensure that this layer, the npm install layer comes before we copy our source code.



If nothing changed about our code and you just want to restart the container, you can do this by running *docker start.* It will start the container in a different mode, where the process will run in the background.

****

In the ***-d*** argument the docker will run the container in the detached mode, where we will not be able to see the event or interact with the container.

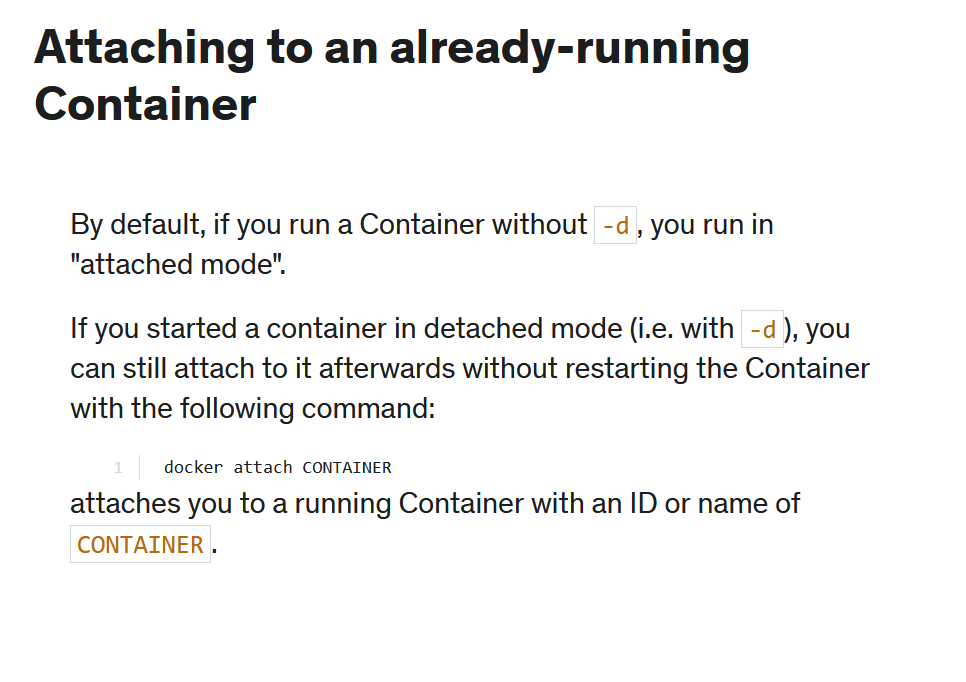
****

***attach*** will attach the container with the current terminal and you will be able to see the logs again.

****

****

***docker logs*** will fetch the logs for the given container name

****