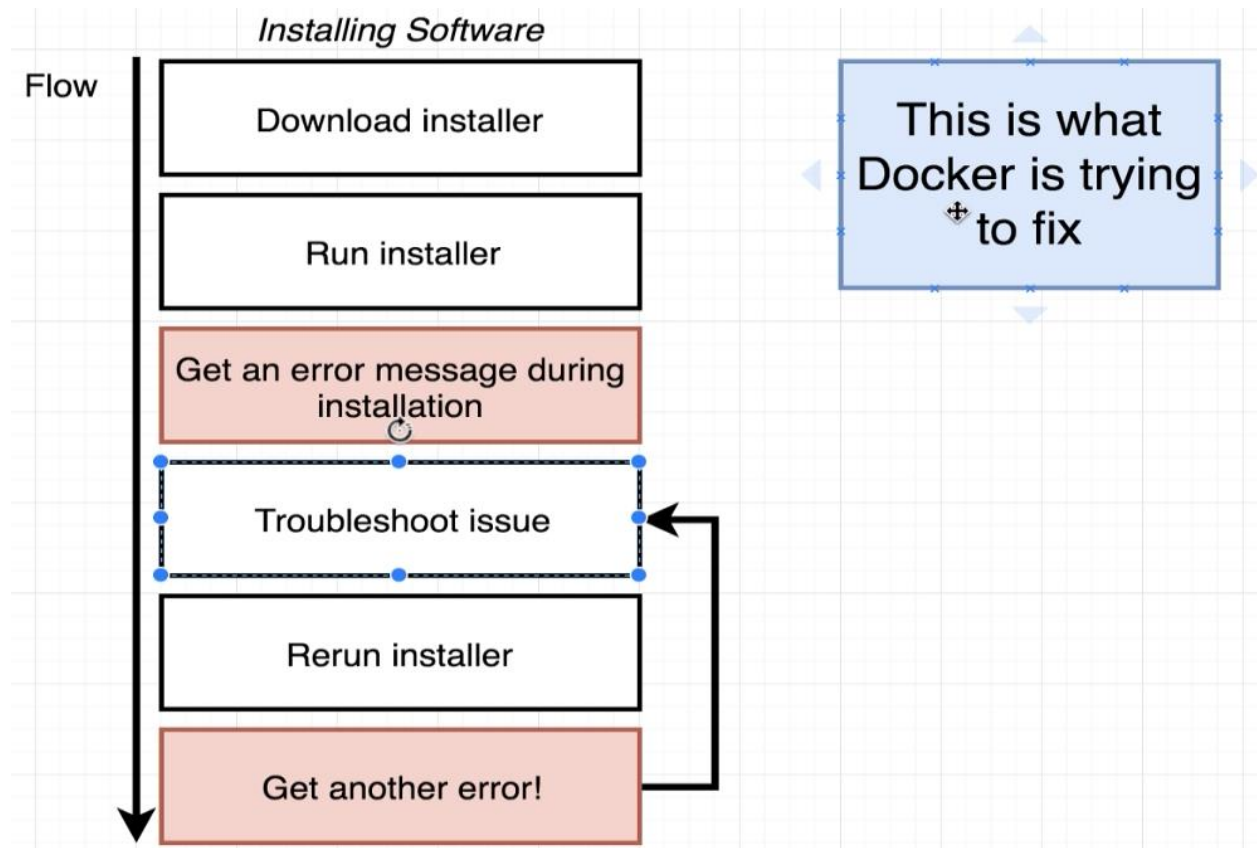


# Docker a Kubernetes

Build, test, a deploy Docker applications s Kubernetes a zároveň sa naučiť pracovné postupy vývoja v produkčnom štýle.

Docker teoretická časť -

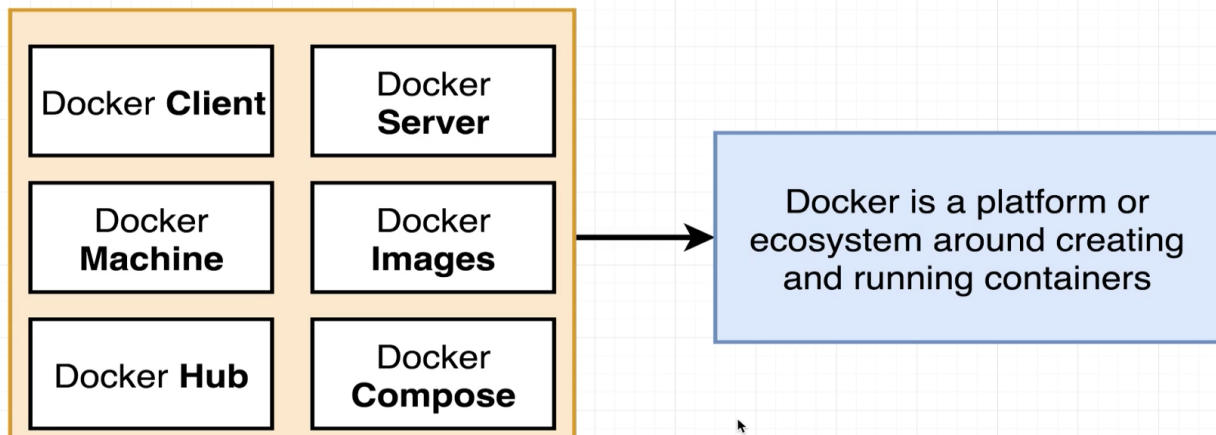
Prečo používať docker ?



Docker sa snaží čo najľahšie a najrychlejšie nainštalovať nejaký software na počítač či už osobný alebo akykoľvek iný (webservice a cloud base).

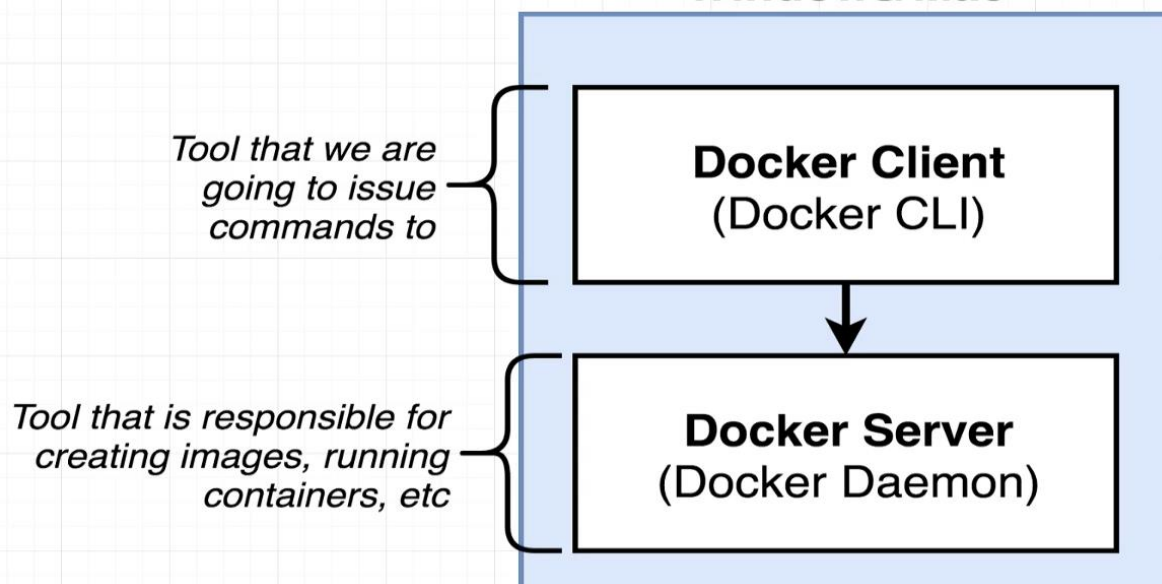
Co je docker ?

## Docker Ecosystem

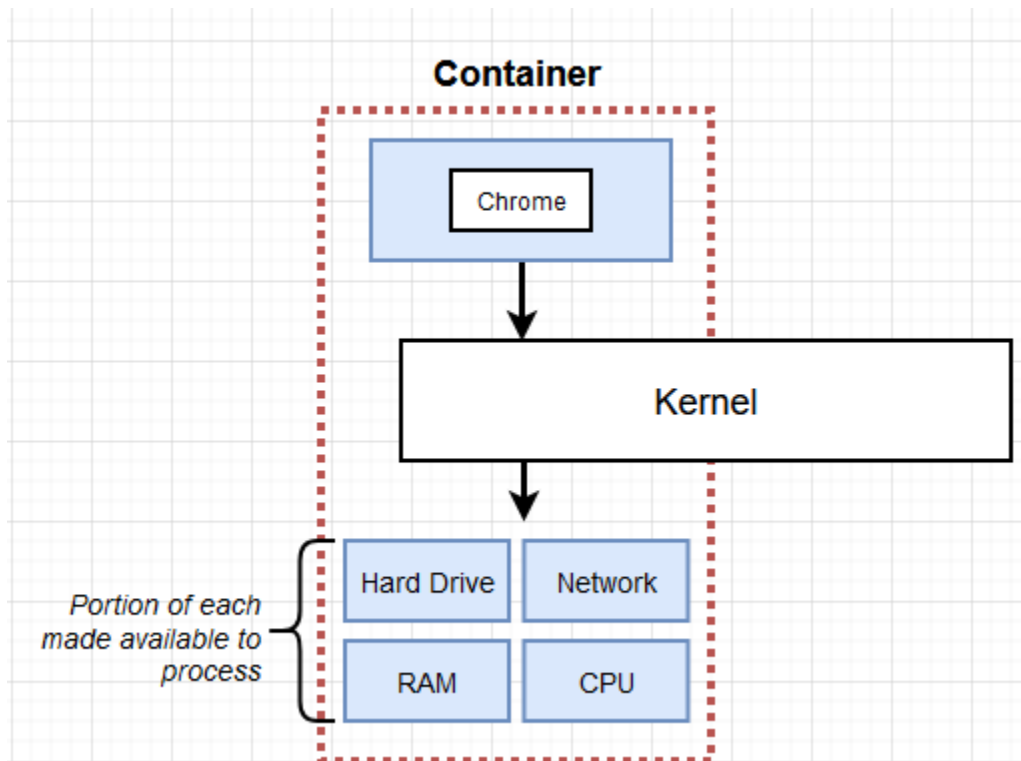


Container je instancia imagu ktora spusta program predom zadefinovaneho image templatu , Docker CLI -> docker hub -> stiahne image -> a docker na zaklade toho spusti a vytvori container ktory berie template z toho imagu

## Docker for Windows/Mac

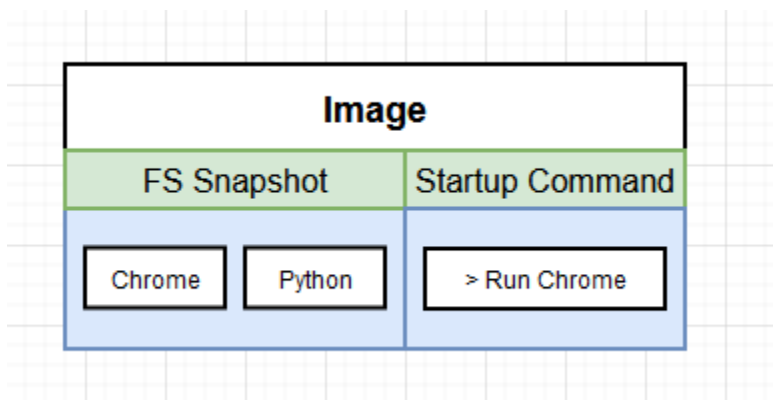


Co je container ?



**container** (kontajner) izolované prostredie, ktoré umožňuje spustenie aplikácie alebo služby s všetkými potrebnými závislosťami a konfiguráciou, bez ohľadu na to, na akom operačnom systéme alebo hardvéri sa spúšťa.

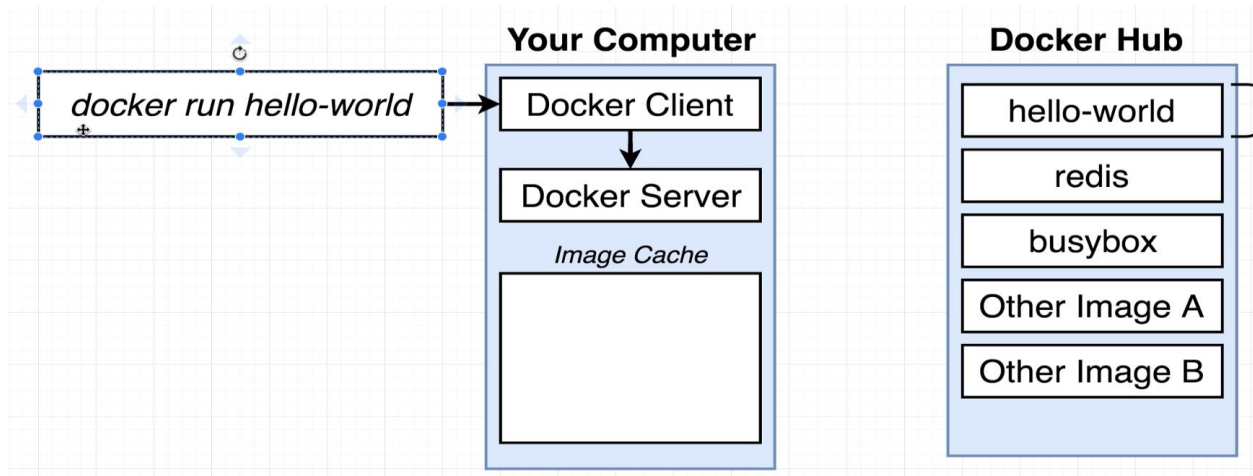
Ako jeden image môže spustiť viac containerov ?



Docker prakticka časť

## Hello-world

[https://hub.docker.com/\\_/hello-world](https://hub.docker.com/_/hello-world)



Tým že sme tento image nepoužívali tak nevedle najst ten image v cache tak ho stiahol pomocou docker servre z docker hub (dokumentacia dockerhub hello-world)

Ak použijeme prikaz

```
user@DESKTOP-F5J917O:~$ docker run busybox ls
```

bin

dev

etc

home

lib

lib64

proc

root

sys

tmp

usr

Var

Tak všetko sa vykona v containery

Toto však nemožeme spraviť pri hello-world lebo v imagine hello-world file systeme tento príkaz nie je preto ho nepozná a nevie ho spustiť.

```
user@DESKTOP-F5J917O:~$ docker run hello-word echo hi there
```

Unable to find image 'hello-word:latest' locally

docker: Error response from daemon: pull access denied for hello-word, repository does not exist or may require 'docker login'.

See 'docker run --help'.

Rozdiel medzi spustiním a vytvorením

```
user@DESKTOP-F5J917O:~$ docker create hello-world
```

```
e8409ce15f415540b05d22b6959528dbe47e65b964158d923d3fee6c6dd84f6e
```

```
user@DESKTOP-F5J917O:~$ docker start -a
```

```
e8409ce15f415540b05d22b6959528dbe47e65b964158d923d3fee6c6dd84f6e
```

### Hlavné rozdiely:

Príkaz	Čo robí	Výstup
docker create	Vytvorí kontajner, ale <b>nevykoná ho</b> .	Vráti ID kontajnera, ale neukáže výstup.
docker start	Spustí <b>už vytvorený</b> kontajner.	Ak použiješ -a, pripojí sa na výstup kontajnera.
docker run	<b>Vytvorí a spustí</b> nový kontajner zobrazený v príkaze.	Spustí kontajner a pripojí sa na jeho výstup, až kým sa kontajner neskončí.

### Kedy použiť ktorý príkaz?

- `docker run` je najjednoduchší a najpoužívanejší príkaz, keď chceš rýchlo spustiť kontajner na základe obrazu, pretože v sebe kombinuje vytvorenie aj spustenie kontajnera.
- `docker create` a `docker start` môžeš použiť v situáciách, kde chceš oddeliť proces vytvárania kontajnera od jeho spustenia, čo môže byť užitočné napríklad v prípade, že chceš najprv upraviť konfigurácie kontajnera, alebo potrebujete spustiť kontajner viackrát s rôznymi parametrami.

### Docker stop a docker kill

	Bezpečne		
	zastaví	<code>SIGTERM</code> (pre pokus o	10 sekúnd (predvolené),
<code>docker</code>	kontajner,	bezpečné ukončenie),	po ktorých sa pošle
<code>stop</code>	umožní aplikácii	následne <code>SIGKILL</code> ak sa	<code>SIGKILL</code> ak kontajner
	správne ukončiť	kontajner neukončí včas.	stále beží.
	činnosť.		
<code>docker</code>	Okamžite zabije kontajner, nečaká na	<code>SIGKILL</code> (alebo iný	Okamžité
<code>kill</code>	ukončenie procesov, bez ohľadu na to,	signál, ak je	zastavenie
	či aplikácia je pripravená.	špecifikovaný).	kontajnera.

Ak sa container po 10 sekundach ak zadame prikaz `docker stop` nestopne -> vykona sa `docker kill` automaticky

### Vytvorenie Docker image

Postup :

Dockerfile -> docker client -> docker server -> použiteľny image

Vytvorenie dockerfilu:

Specifikovať základný image -> spustenie príkazov na podporné service -> špecifický command na spustenie containera

Vytvorenie docker image, ktorý nám spustí redis server.

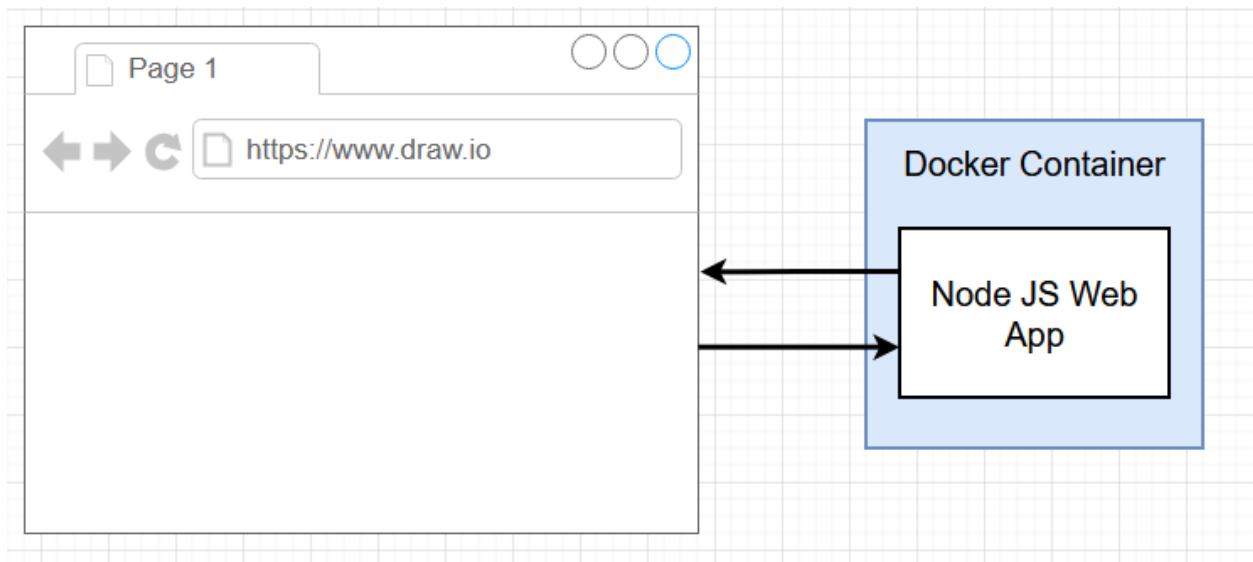
Informácie v docker\_linux\_CLI\_commands.txt

V sekcii “DOCKERFILE”

Zhrnutie :

V tomto docker file sme prv nainštalovali alpine linux verziu z docker hubu , to je základny blok pre náš container a nasledne sme nainštlovali redis (je to linux comand ktory sme mohli dat lebo používame alpine linux) , a posledny command spusti redis pri štarte containera.

### Application – tiny Node.JS web app



Stručne kroky vytvorenia -

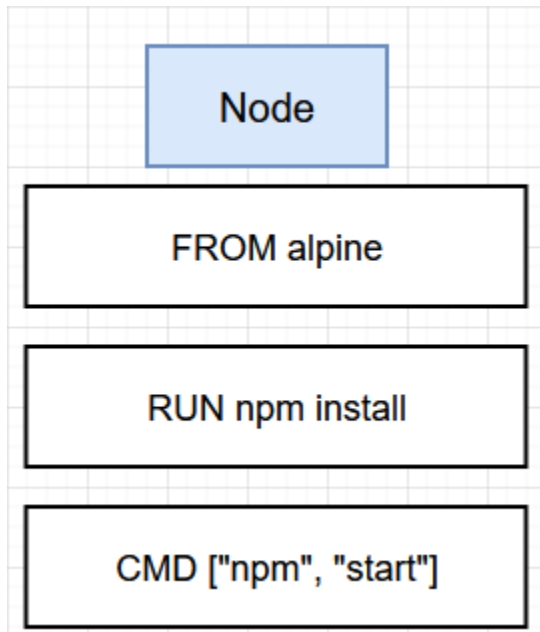
Vytvorenie node.js web app -> vytvorenie dockerfilu -> build image z dockerfilu -> run image ako container -> prepojenie web app s prehliadačom

#### Vysvetlenie kodu

Index.js ->

Tento kód je príklad základnej aplikácie postavenej na Express.js, čo je webový framework pre Node.js, ktorý zjednodušuje tvorbu serverov a webových aplikácií, vytvára jednoduchý webový server pomocou Express.js, ktorý odpovedá na GET požiadavky na URL / (hlavnú stránku) s textom 'Hi'. Server počúva na porte 8080.

### Vytvorenie docker imagu



npm (Node Package Manager) je nástroj pre správu balíkov (knižníc) v prostredí Node.js. Umožňuje vývojárom jednoducho inštalovať, aktualizovať a spravovať knižnice a nástroje, ktoré sú potrebné pre ich projekty.

V error handlingu sme sa stretli s chybou – oprava je tá že miesto alpine použijeme iný image

```
Dockerfile > ...  
1 # specify base image  
2 FROM node:14-alpine  
3
```

### Error handling

Pri docker build sme dostali



```

user@DESKTOP-F5J9170:~/projects/nodejs_web_application$ docker build .
[+] Building 3.9s (6/6) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile             0.0s
=> => transferring dockerfile: 149B                             0.0s
=> [internal] load metadata for docker.io/library/alpine:latest 2.4s
=> [auth] library/alpine:pull token for registry-1.docker.io    0.0s
=> [internal] load .dockerignore                               0.1s
=> => transferring context: 2B                                   0.0s
=> [1/2] FROM docker.io/library/alpine:latest@sha256:21dc6063fd678b478f57c0e13f47560d0ea4eeba26dfc947b2a4f81f686 0.9s
=> => resolve docker.io/library/alpine:latest@sha256:21dc6063fd678b478f57c0e13f47560d0ea4eeba26dfc947b2a4f81f686 0.0s
=> => sha256:38a8310d387e375e0ec6fabe047a9149e8eb214073db9f461fee6251fd936a75 3.64MB / 3.64MB 0.6s
=> => extracting sha256:38a8310d387e375e0ec6fabe047a9149e8eb214073db9f461fee6251fd936a75 0.1s
=> ERROR [2/2] RUN npm Install                                0.4s
-----
> [2/2] RUN npm Install:
0.326 /bin/sh: npm: not found
-----
Dockerfile:5
-----
 3 |
 4 |     #Install depenendecies
 5 |     >>> RUN npm Install
 6 |
 7 |     #default command
-----
ERROR: failed to solve: process "/bin/sh -c npm Install" did not complete successfully: exit code: 127

```

Alpine je dost maly image na to aby spustilo npm install , preto miesto alpine použijeme iny image - ("node:14-alpine")

```
=> ERROR [2/2] RUN npm Install 0.9s
-----
> [2/2] RUN npm Install:
0.417
0.417 Usage: npm <command>
0.417
0.417 where <command> is one of:
0.417   access, adduser, audit, bin, bugs, c, cache, ci, cit,
0.417   clean-install, clean-install-test, completion, config,
0.417   create, ddp, dedupe, deprecate, dist-tag, docs, doctor,
0.417   edit, explore, fund, get, help, help-search, hook, i, init,
0.417   install, install-ci-test, install-test, it, link, list, ln,
0.417   login, logout, ls, org, outdated, owner, pack, ping, prefix,
0.417   profile, prune, publish, rb, rebuild, repo, restart, root,
0.417   run, run-script, s, se, search, set, shrinkwrap, star,
0.417   stars, start, stop, t, team, test, token, tst, un,
0.417   uninstall, unpublish, unstar, up, update, v, version, view,
0.417   whoami
0.417
0.417 npm <command> -h   quick help on <command>
0.417 npm -l             display full usage info
0.417 npm help <term>    search for help on <term>
0.417 npm help npm       involved overview
0.417
0.417 Specify configs in the ini-formatted file:
0.417   /root/.npmrc
0.417 or on the command line via: npm <command> --key value
0.417 Config info can be viewed via: npm help config
0.417
0.417 npm@6.14.18 /usr/local/lib/node_modules/npm
0.419
0.419 Did you mean one of these?
0.419   install
0.419   uninstall
0.419   unstar
-----
Dockerfile:5
-----
3 |
4 |   #Install depenendecies
5 | >>> RUN npm Install
6 |
7 |   #default command
-----
ERROR: failed to solve: process "/bin/sh -c npm Install" did not complete successfully: exit code: 1
```

Pridanie "COPY ./ ./" - to znamená že všetky súbory ktoré sa nachádzajú v adresári sa skopirovali do vnútra containera