Vagrant Day 2

```
config.vm.synced_folder "app", "/home/vagrant/app"
```

- this line of code says it would like to add the file "app" from my machine to the vm
- we destroyed our vm from yesterday, started up a new one and used a zip file downloaded from Shahruk

the app file did not appear so I used the

For Mac Users running a new Vagrant file, an error can occur with VM saying the user created doesn't match with
current user running Vagrant. To resolve this error, these are steps to resolve it:

- Sometimes, it causes a conflict and it uses the one of the old .vagrant file to start up the machine. To avoid this
  conflict and start from a completely new vagrant machine.

```
rm -r .vagrant
```

This should remove the .vagrant file which deletes the conflict from a previous vagrant machine. In other words,
resolving the error.

```
remove -r .vagrant
```

to remove the .vagrant file as it kept trying to route the

Communication with Developer is vital to understand the requirements

Comms is key to successful projects

- communication between dev-ops-testers-QA and devops

Questions to ask when placed on a proj

- What language is used to build this app?
- What framework is being used?
    - MVC, React (frontend framework), Node, Nginx, Ruby
    - These need to be installed beforehand
- Are there any dependencies to be installed together?
    - dependencies are things like modules that allow requests to work
    - dependencies can be a package, software, module
- What will the app look like?

Installing Bundler

```
sudo -i
```

- lets us run commands as admin
- then we should be in root
- we then did:

```
gem install bundler
```

- then exit root
- then password and it should be working

```
rake spec
```

- runs a series of tests which failed
- we then need to install the things that are missing
- run this in the rests file

```
so sudo apt-get update
sudo apt-get install nginx
sudo apt-get upgrade
systemctl status nginx
```

- then we ran rake spec again, and we had fewer failures
- we then needed to install nodejs (follow the above sudo commands)
- after installing nodejs, we ran rake spec again, resulting in even fewer failures

EXERCISE: install pm2

- I googled the instructions for installing pm2 on my VM
- I found out I had to run it as an admin so I used sudo -i
- next, I ran the command:

```
apt install npm
```

because npm is necessary for pm2 to be installed

next command was:

```
npm install -g pm2
```

the result looks like this:

I then went to the spec tests dir and ran rake spec which showed just 1 error remaining

Next things to install

```
sudo apt-get install python-software-properties
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
sudo apt-get install -y nodejs
```

next step

```
sudo npm install pm2 -g
rake spec
```

- this resulted in 0 failures!

Next step

```
npm install
```

- npm is a package manager for node, its like flask, requests
- its like importing a module in python

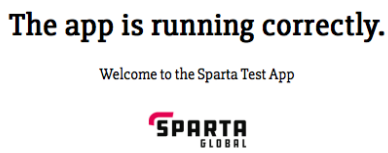
Next command

```
node app.js
```

- this message was returned: Your app is ready and listening on port 3000

NEXT

- I pasted this into my browser: http://development.local:3000/ and the below was the result




```
nginx_installation_script.sh
```

sh is an extension of shell script

```
chmod +x nginx_installation_script.sh
```

- changes it to an executable file

```
./nginx_installation_script.sh
```

- above code executes the code inside the nginx_installation file

We then entered the nginx_installation file with nano and entered:

```
#!/bin/bash

sudo apt-get update
sudo apt-get install nginx
sudo apt-get upgrade
```

```
sudo apt-get remove nginx
```

- the above code was a test to remove the nginx install, and then I did the ./nginx_installation_script.sh to do the automated reinstall

> 💡 This is particularly powerful as it automates the installation process from a few lines of code, highlights how DevOps is used as it saves time and money.

Synching

- any changes we make in the app folder from our OS will be reflected in our machine
    - the same applies vice a versa

NEW COMMANDS

```
top
ps
chmod +x file_name.sh
cat filename
```

- top command shows all the programmes running on the system
- ps command give the programID
- chmod makes it an executable file
- cat filename command displays the file content on your terminal (computer aided translation)

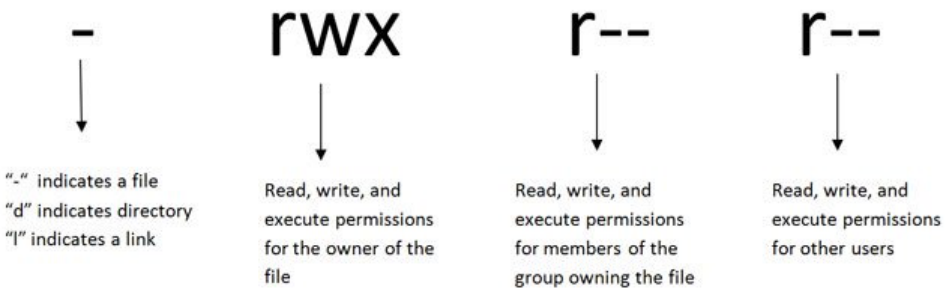
Exercise: "find the Linux permission codes to assign permissions to files"

To change directory permissions in Linux, use the following:

- **chmod +rwx filename** to add permissions.
- **chmod -rwx directoryname** to remove permissions.
- **chmod +x filename** to allow executable permissions.
- **chmod -wx filename** to take out write and executable permissions.

Note that “r” is for read, “w” is for write, and “x” is for execute.

This only changes the permissions for the owner of the file.



Changing permissions using Numeric code

You may need to know how to change permissions in numeric code in Linux, so to do this you use numbers instead of “r”, “w”, or “x”.

- **0 = No Permission**
- **1 = Execute**
- **2 = Write**
- **4 = Read**

Basically, you add up the numbers depending on the level of permission you want to give.

- **0 = ---**
- **1 = --x**
- **2 = -w-**
- **3 = -wx**
- **4 = r-**
- **5 = r-x**

- **6 = rw-**
- **7 = rwx**
- **chmod 777 foldername** will give read, write, and execute permissions for everyone.
- **chmod 700 foldername** will give read, write, and execute permissions for the user only.
- **chmod 327 foldername** will give write and execute (3) permission for the user, w (2) for the group, and read, write, and execute for the users.

Provisioning

config.vm.provision "shell", path: "environment/provision.sh"

First `vagrant destroy` Then `vagrant up`

`config.vm.provision` - Configures provisioners (refers to process of setting up IT infrastructure) on the machine, so that software can be automatically installed and configured when the machine is created. `config.vm.synced_folder` - Configures synced folders on the machine, so that folders on your host machine can be synced to and from the guest machine.

```
config.vm.synced_folder ".", "/home/vagrant/app"
```

- like the command from earlier today, but the "." tells it to do everything from the root