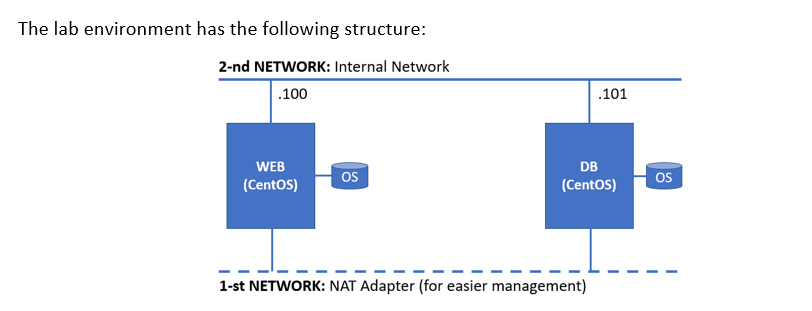
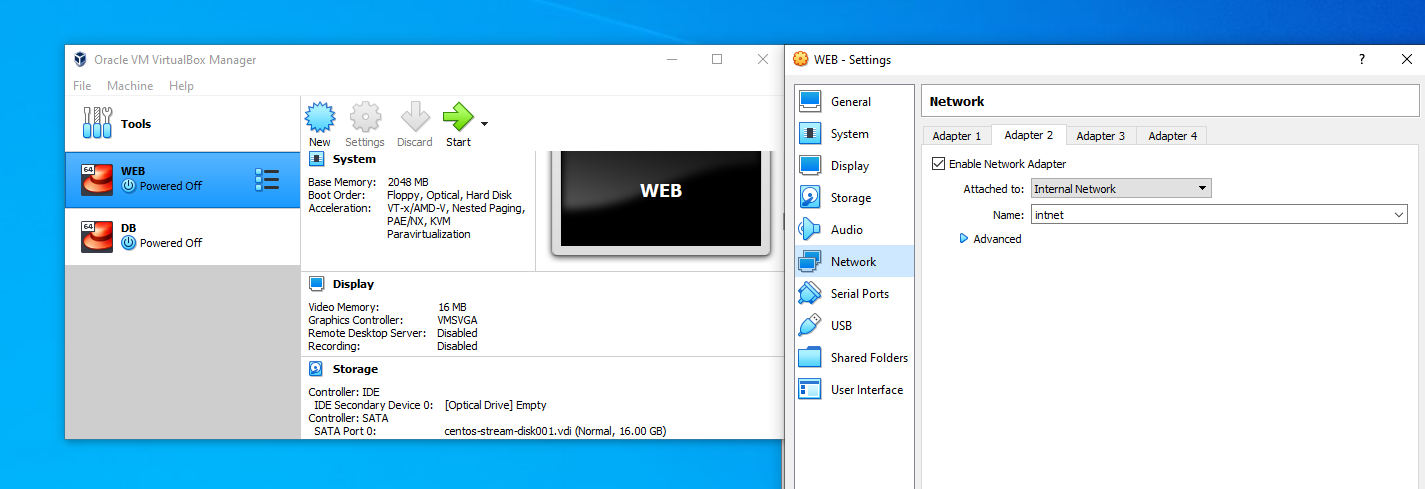
Practice



1. First step import from the application two Cent OS naming **Web and DB**

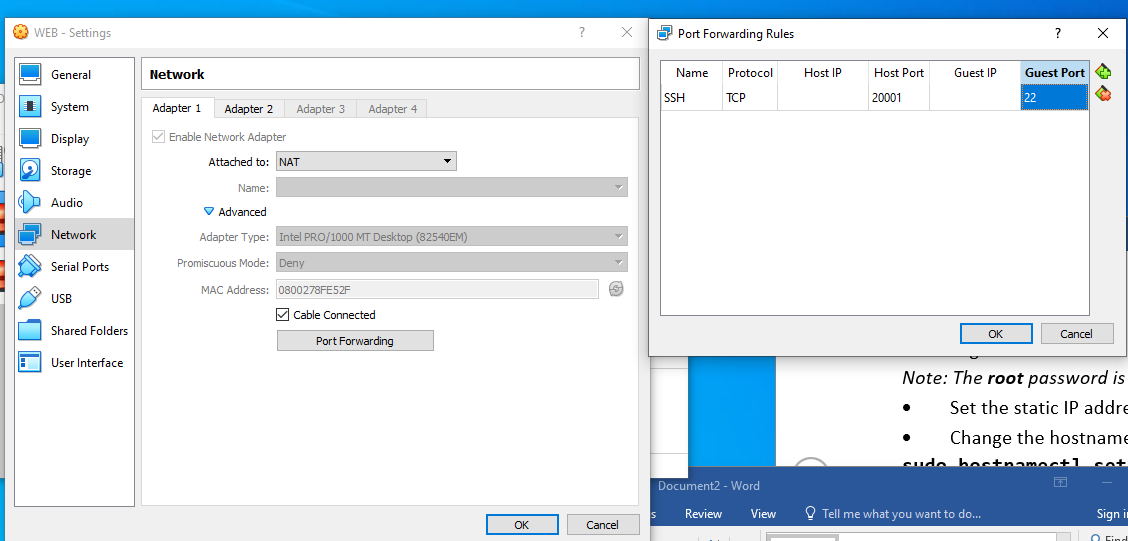
* To have a communication between them we need to enable second network before power on the machines
* 

**Power on the virtual machines**

2. Create SSH connection with both virtual machines /port forwarding for Web and DB

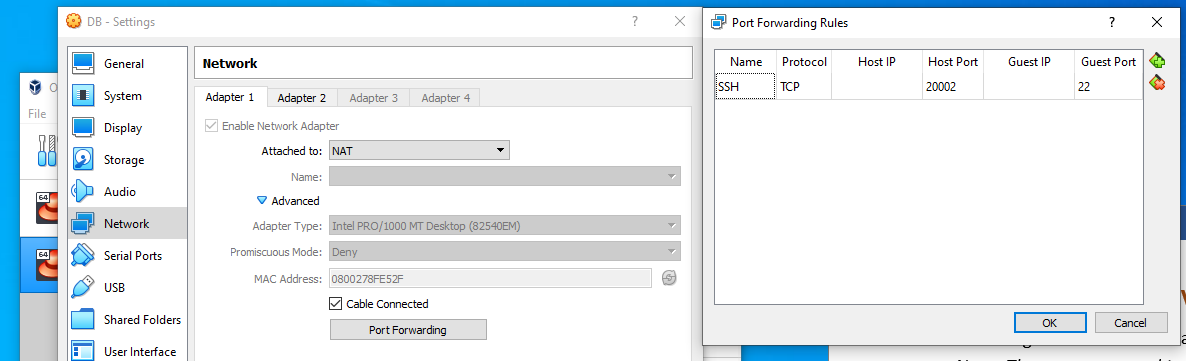
**WEB**

Add rules for SSH



DB

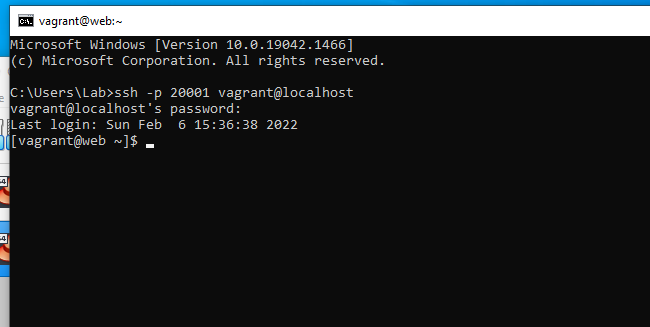
Add Rules for DB



Open CMD and connect to Web with following command

**ssh -p 20001 vagrant@localhost**

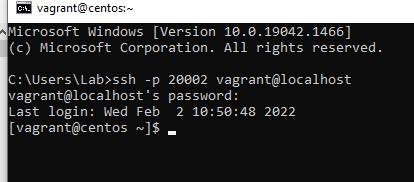
**Password = vagrant**

****

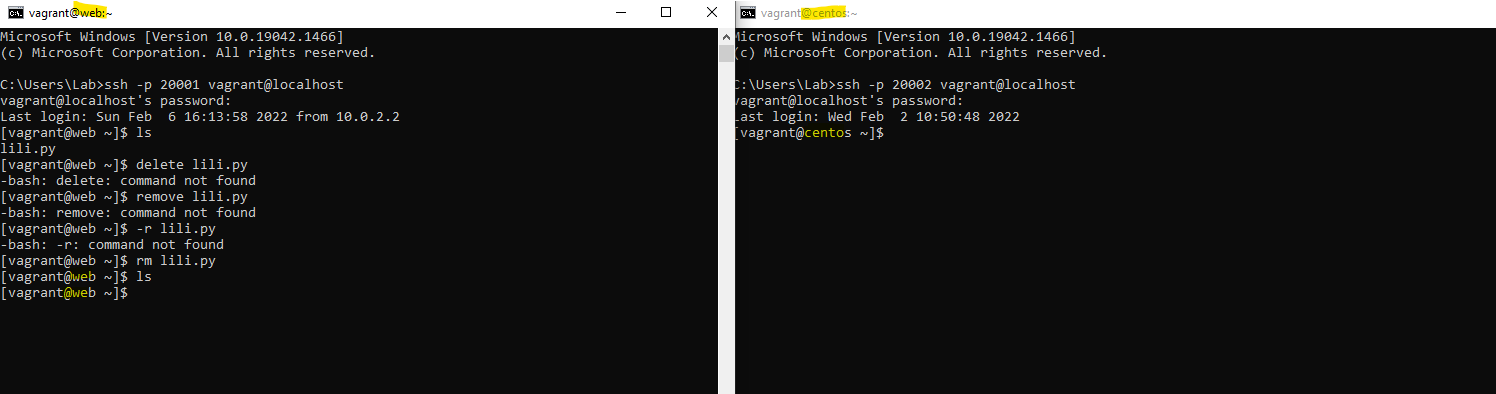
Open second command terminal and login in for DB

**ssh –p 20002 vagrant@localhost**

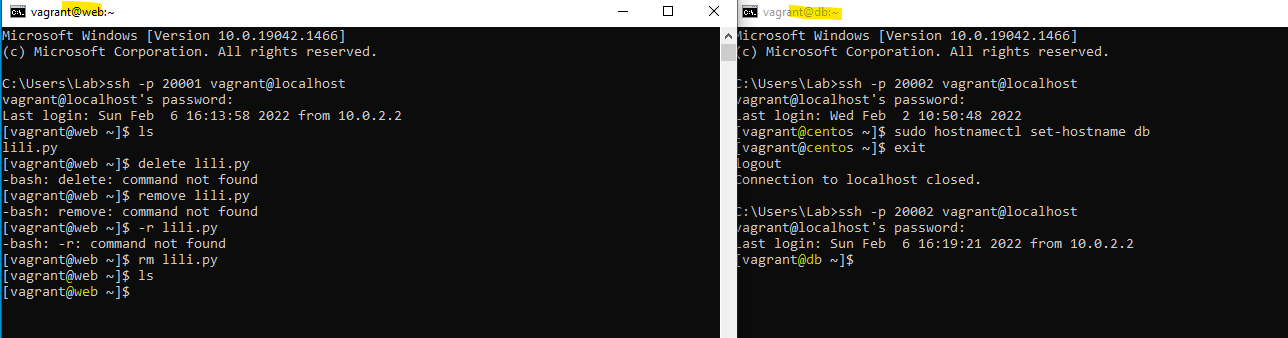
**Password = vagrant**

****

1. We need to Change the hostname to **web** or DB by executing
   * **sudo hostnamectl set-hostname (web/db)**

****

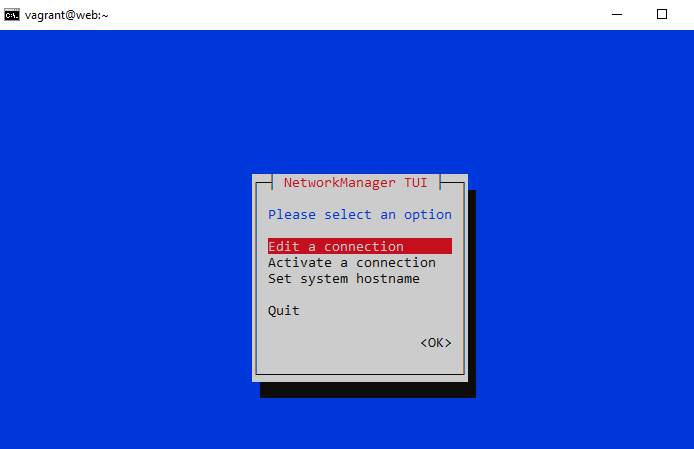
We need to re-login in order to see changes

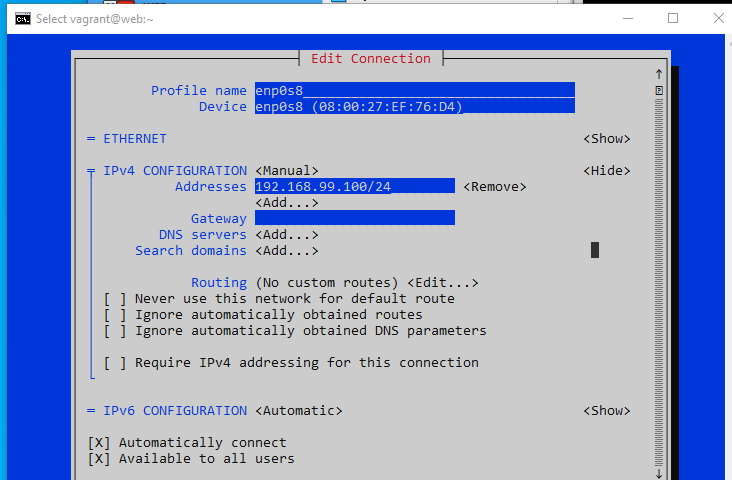


1. Next step to configure the IP address

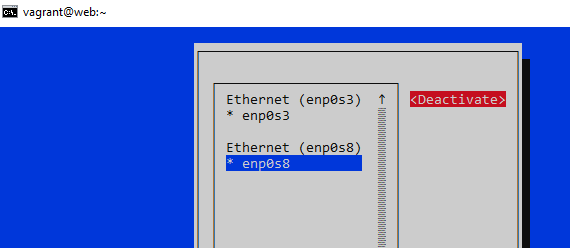
We will use the following command to lunch Network Manager TUI on WEB

**Sudo nmtui**

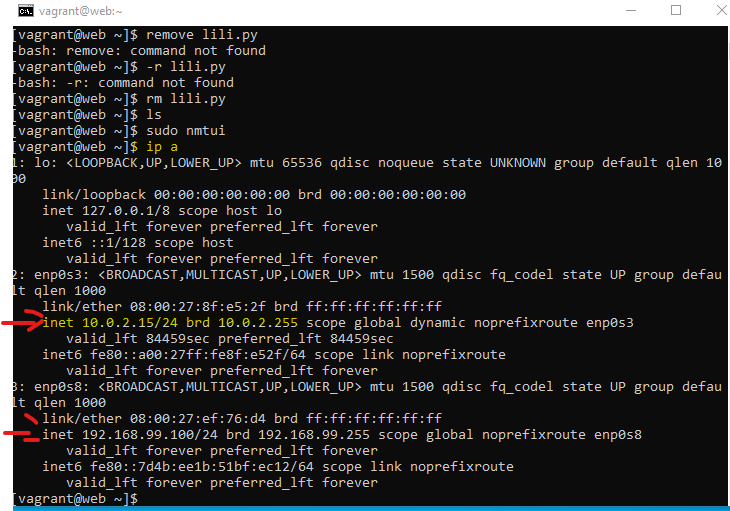
****

**We click on edit connection, rename the profile name and add IP address .  
  
**

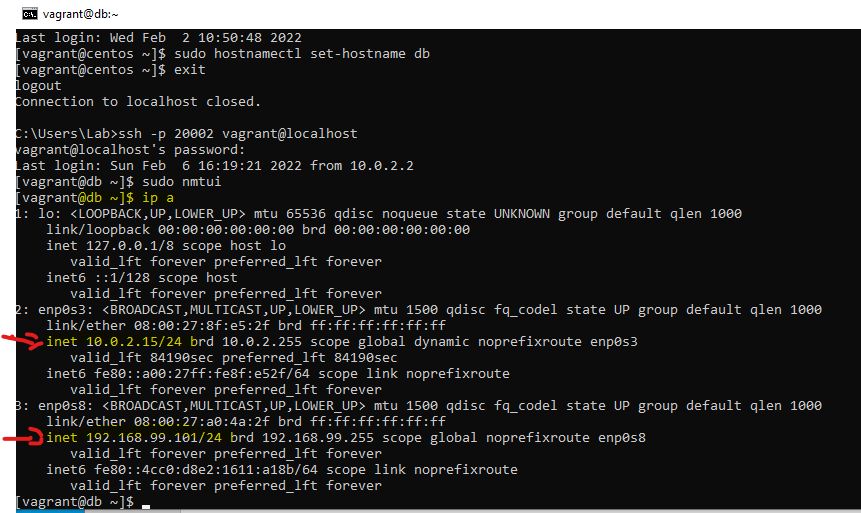
**After that we go to the activate connection and deactivate and activate the second connection.**

****

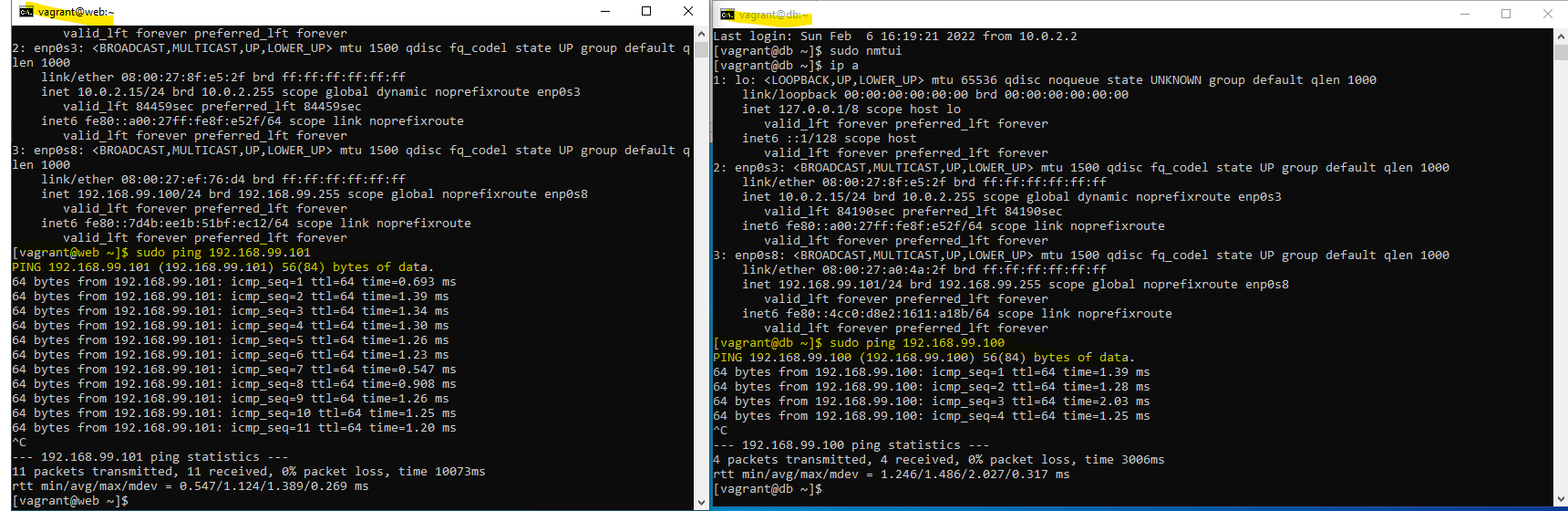
We can verify the **configuration** by typing command - **ip a**

****

We will configure the same step with DB instance we will use the ip add 192.168.99.101/24

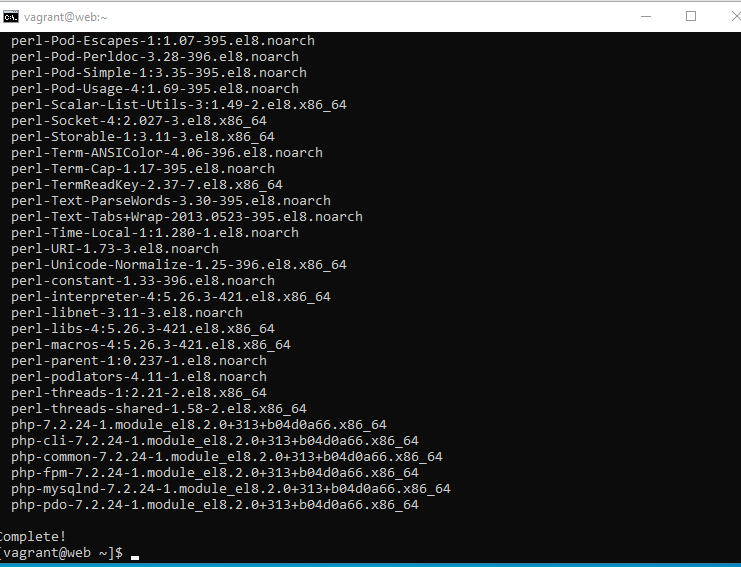


We can ping to machines to verify connection with command - **sudo ping** ip address



1. On WEB machines  
     
    we need to install PHP , Web server and Git

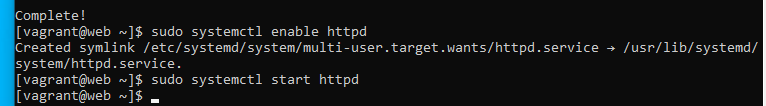
**sudo dnf install -y httpd php php-mysqlnd git**

****

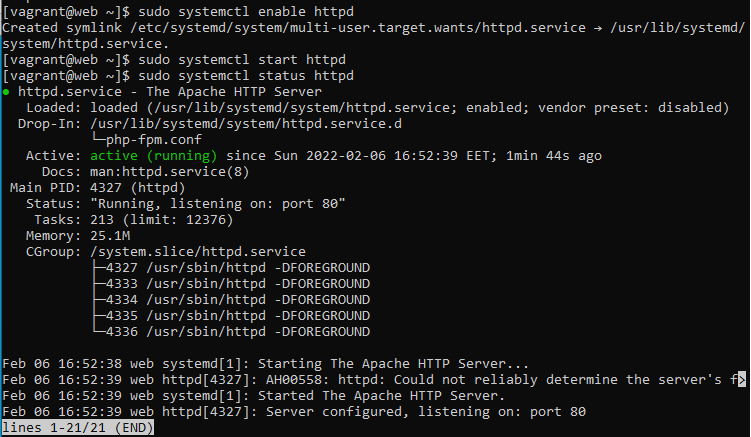
Configure **Apache** to start automatically on boot and start it

**sudo systemctl enable httpd**

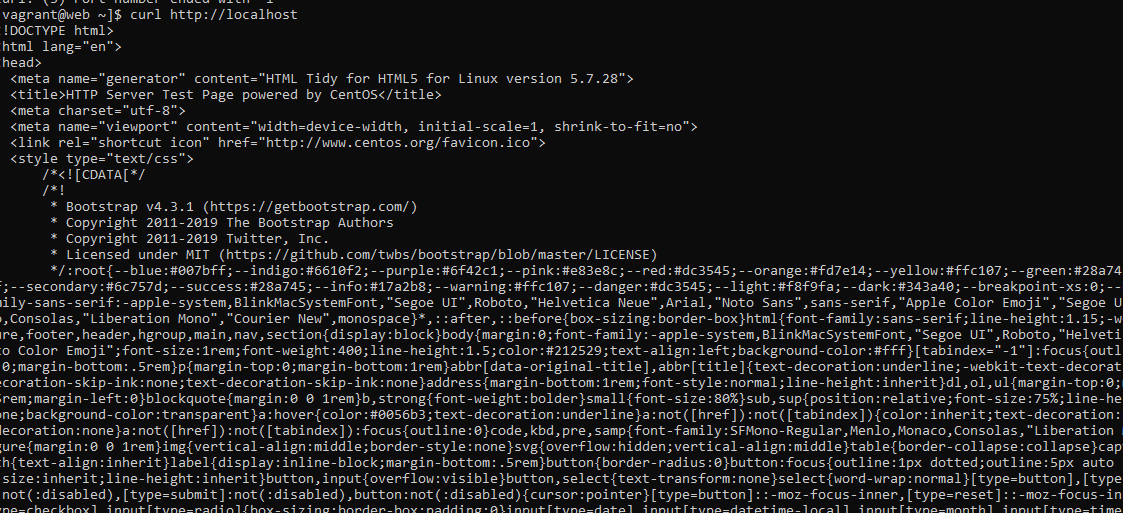
**sudo systemctl start httpd  
  
short cut command for both above is sudo systemctl enable –now httpd**



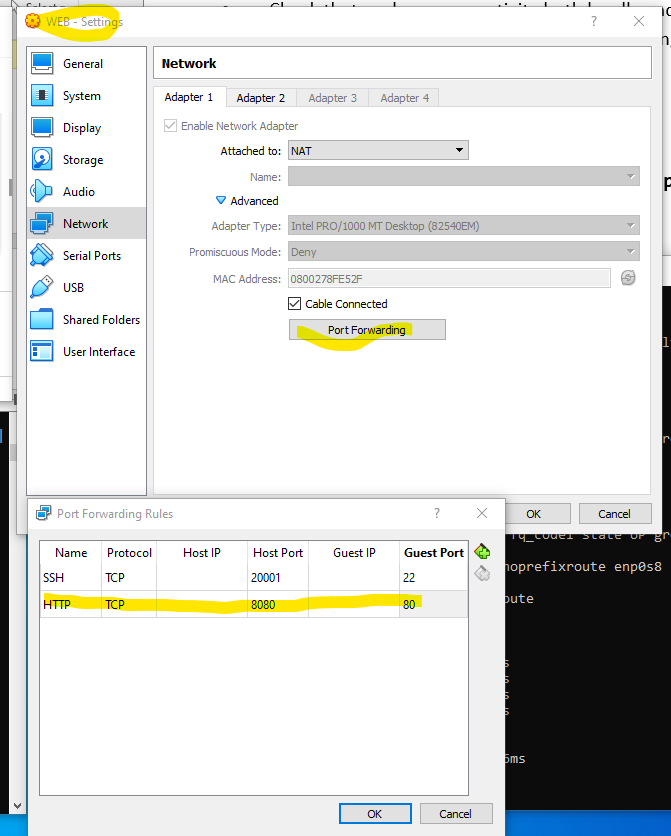
To check the status of the apache server is running: **sudo systemctl status httpd**



Check if we can reach in the server locally command **curl** [**http://localhost**](http://localhost)



We need to add rule for HTTP



Check that we have connectivity both locally and remotely\*

We can adjust the firewall in one of the following ways:

* + Stop and disable it

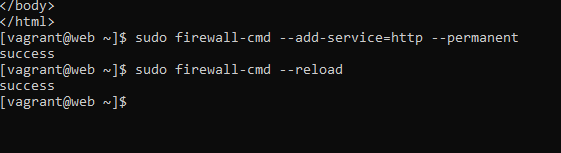
**sudo systemctl stop firewalld**

**sudo systemctl disable firewalld**

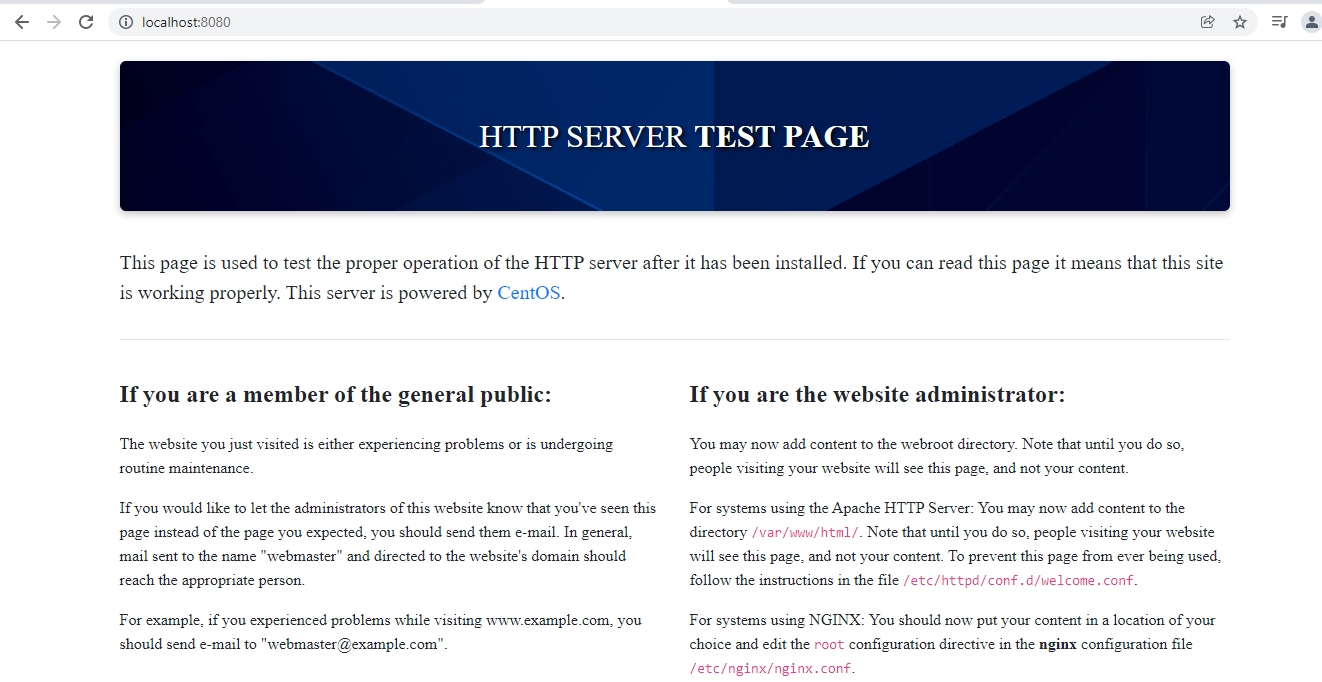
* + Or open the appropriate port or service

**sudo firewall-cmd --add-service=http --permanent**

**sudo firewall-cmd --reload**



Test server page should be appeared on localhost:8080



#### Deploy the application (on WEB VM)

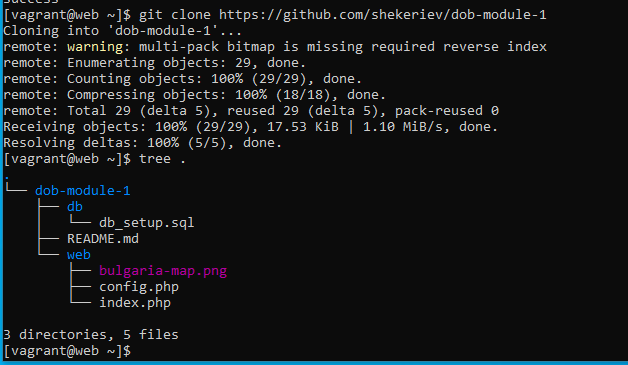
* Go to the home folder of the **vagrant** user
* Execute **git clone** [**https://github.com/shekeriev/dob-module-1**](https://github.com/shekeriev/dob-module-1)
* Copy all files from **dob-module-1/web** to **/var/www/html**

**sudo cp dob-module-1/web/\* /var/www/html/**

* Try to open the page on the host \*

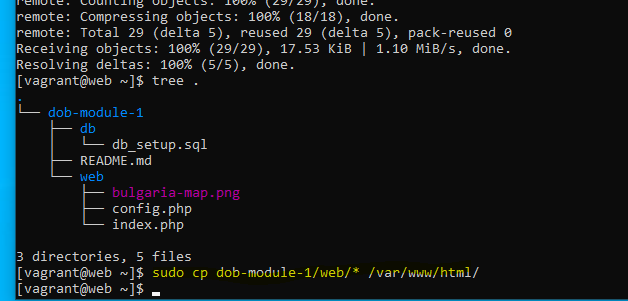
*\* Depending on the network mode of the virtual machines, you may need to set up a port forwarding rule. For example, port* ***80*** *on the* ***VM*** *to port* ***8080*** *on the host*

*Use the command* ***tree .*** *to check what we have.*

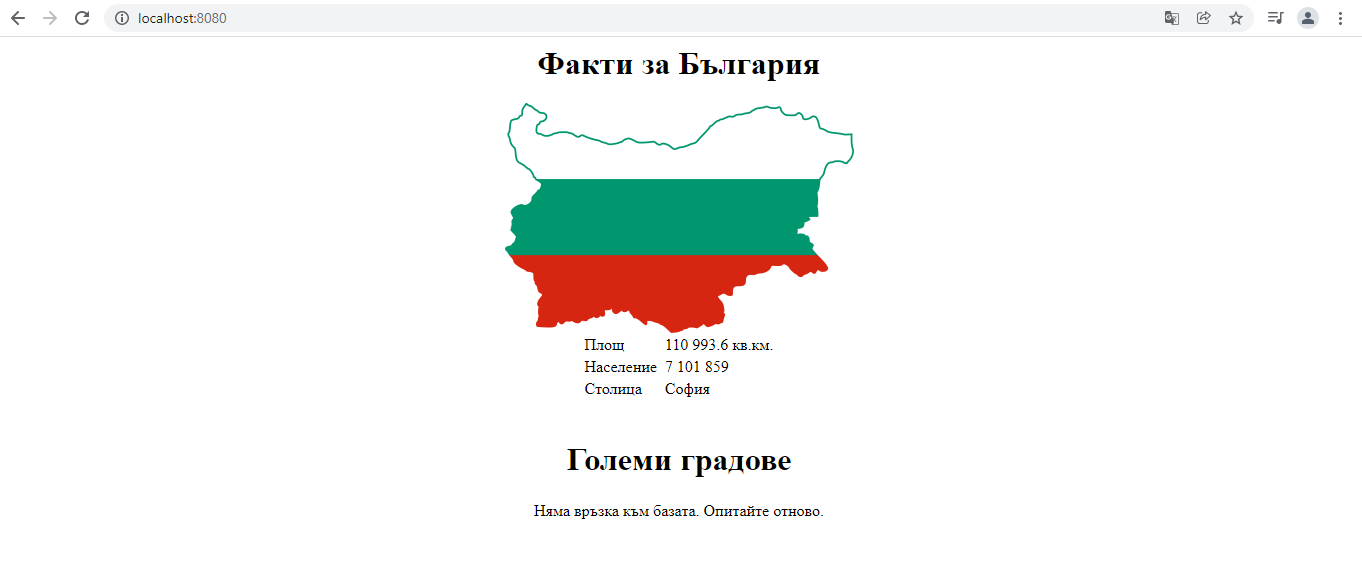
******

We need to copy web component and redirected it

**sudo cp dob-module-1/web/\* /var/www/html/**



After the refresh, the page should like this.



1. Deploy Maria DB on DB VM

Now we ca switch to the DB component to fix the database.

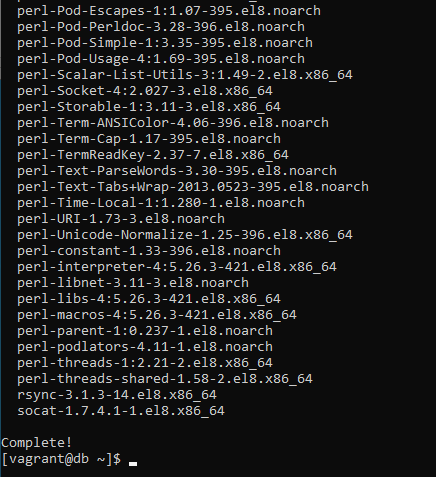
* If using **CentOS Stream**, we must add the **MariaDB** repository with

**curl -LsS https://downloads.mariadb.com/MariaDB/mariadb\_repo\_setup | sudo bash -s -- --mariadb-server-version=10.6 --skip-maxscale --skip-tools**

**

Install **MariaDB** client and server components

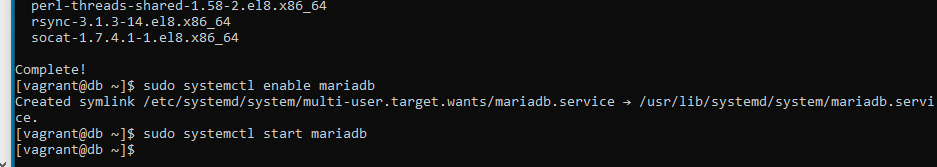
**sudo dnf install -y mariadb mariadb-server git**

****

Enable and start the service

**sudo systemctl enable mariadb**

**sudo systemctl start mariadb**

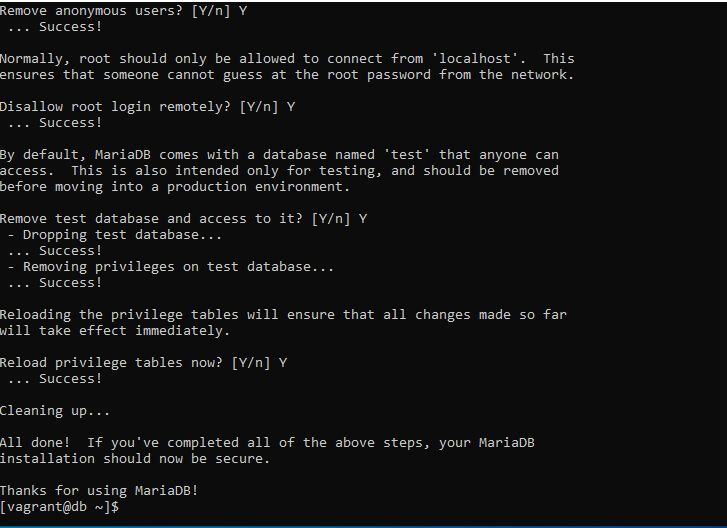
****

Do some initial configuration \*

**sudo myql\_secure\_installation**

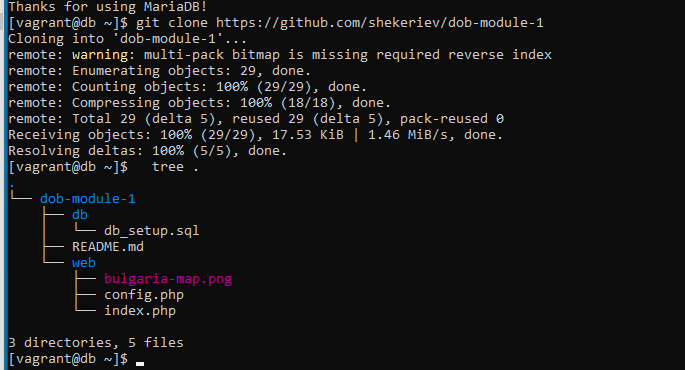
*Note: If installed via a dedicated repository, then the above command should become* ***sudo mariadb-secure-installation***

***Change root password: root***

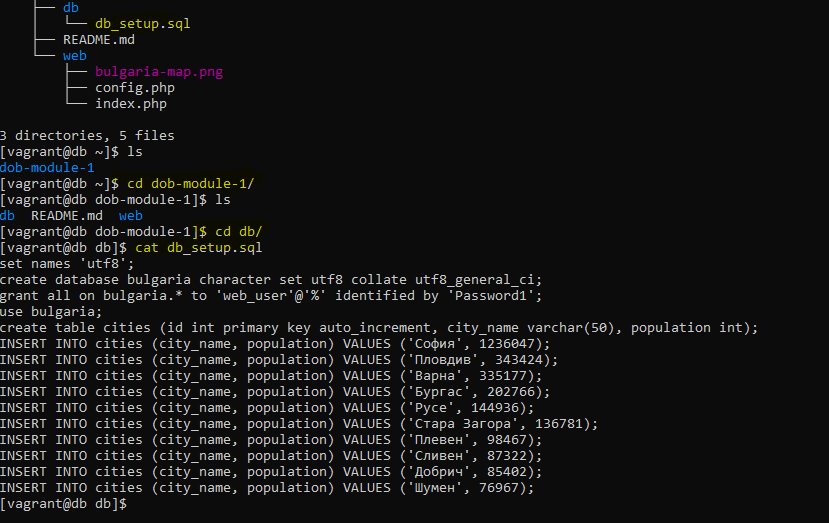
******

Same steps as WEB base

* Execute **git clone** [**https://github.com/shekeriev/dob-module-1**](https://github.com/shekeriev/dob-module-1)



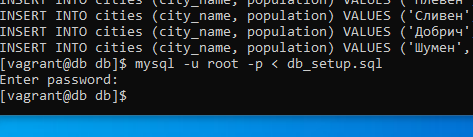
* Navigate to the **db** folder
* Examine the **SQL** script



Execute the script against the database

**mysql -u root -p < db\_setup.sql**

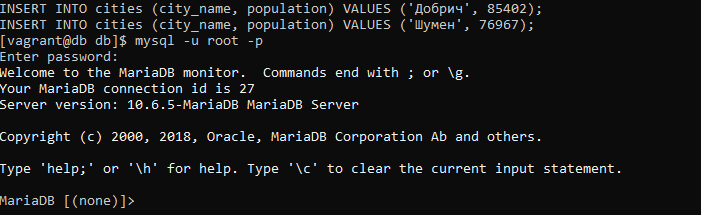
**for the database password = root**

****

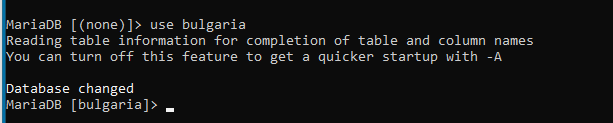
Log-on and check that the data is there

– type the command not copy and paste it .

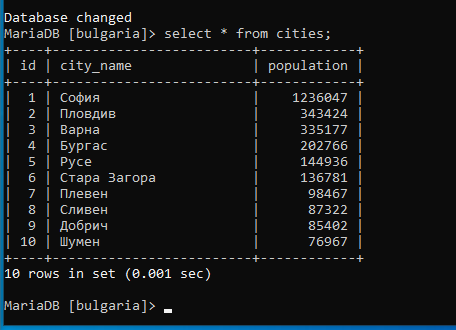
**mysql -u root –p**

****

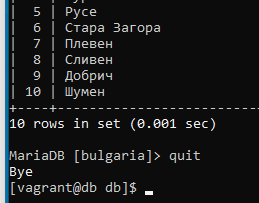
**use bulgaria;**

****

**Select \* from cities;**

****

**Quit**

****

**At this point, we successfully initialize DB component**

We can modify the firewall on DB to check if the issue is coming from there.

* Modify the firewall state:
  + Disable it:

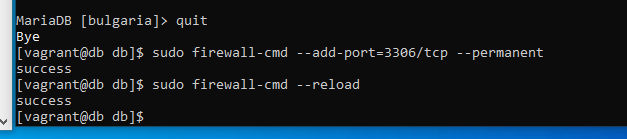
**sudo systemctl stop firewalld**

**sudo systemctl disable firewalld**

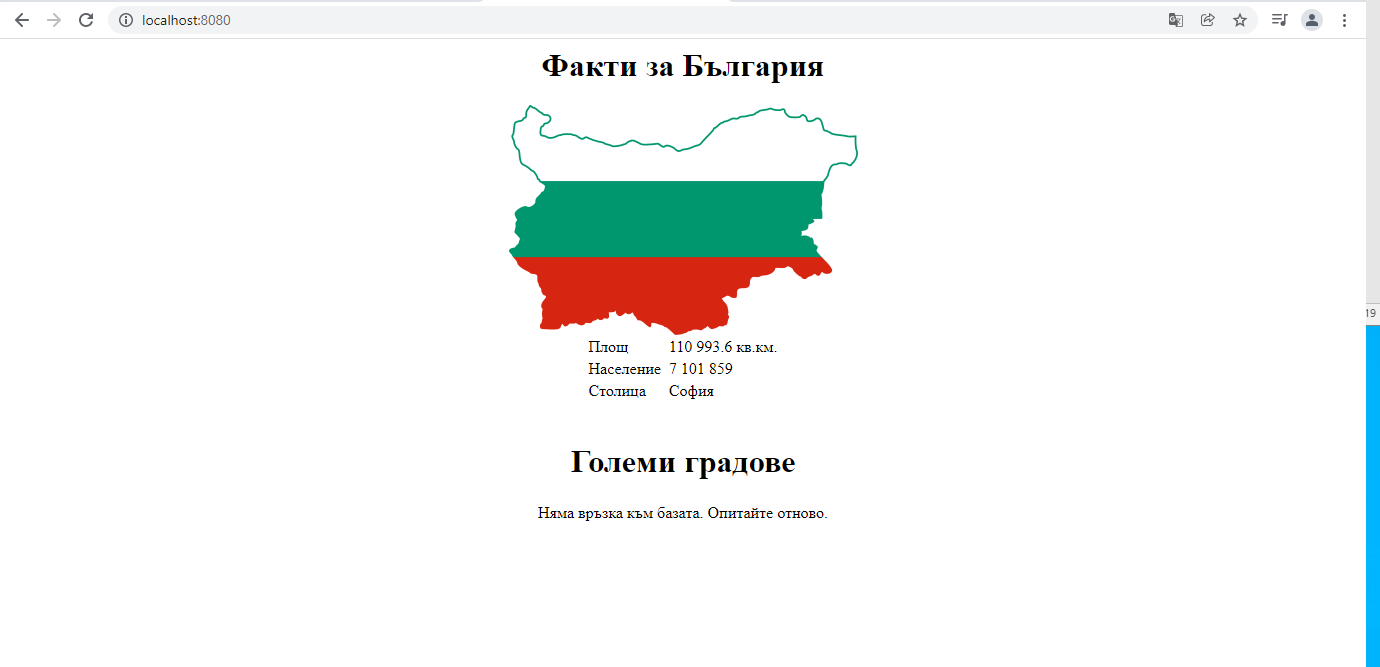
* + Or open the appropriate port:

**sudo firewall-cmd --add-port=3306/tcp --permanent**

**sudo firewall-cmd –reload**

****

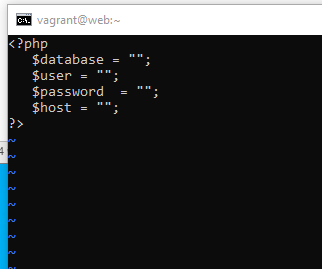
After the changes on firewall still database is not present on the web page.

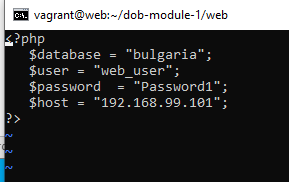
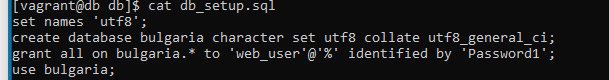


The issue may come from code itself we need to check config.php file from the WEB

sudo vi /var/www/html/config.php

Its empty and we need to fill it

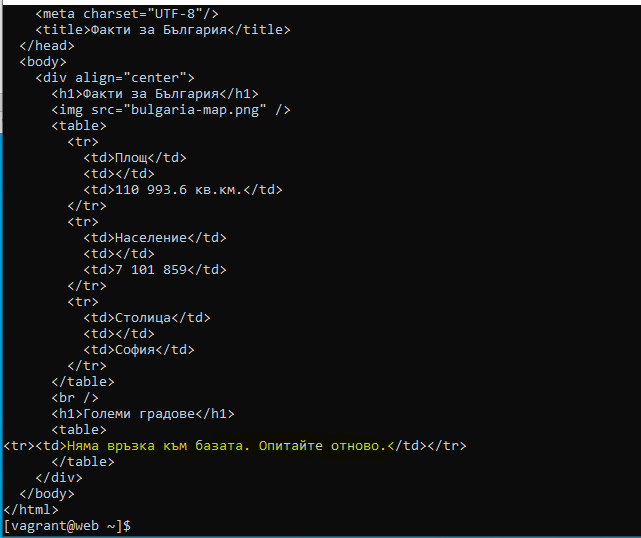


Still the database is not present

We are testing on WEB

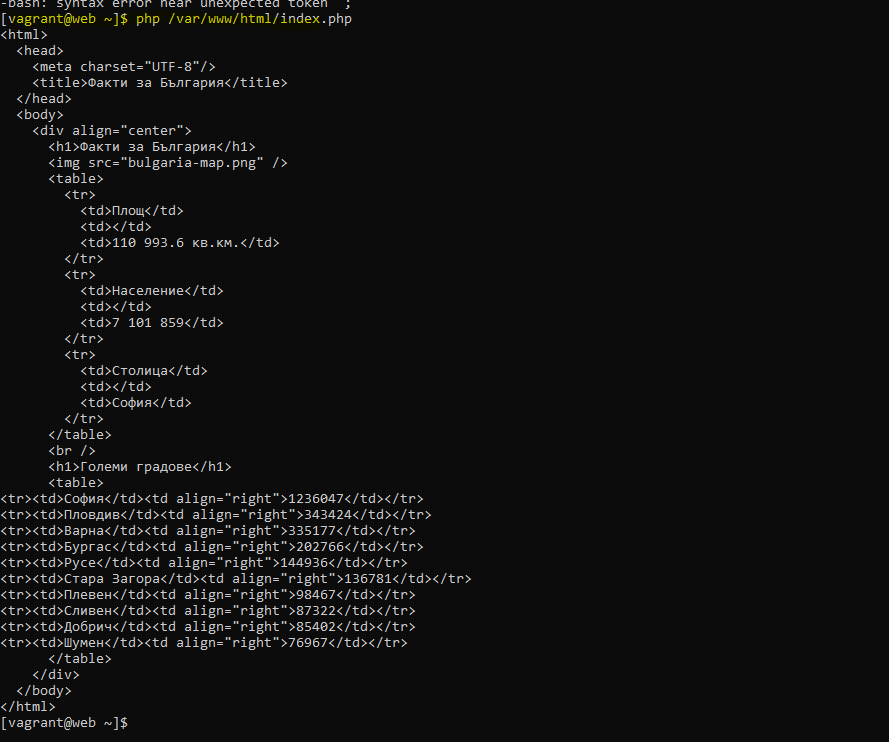
culr <http://localhost>



* Check the **config.php** script if there is a need to change connection parameters
* Open browser and check the final result. It should be working
* In case of connectivity error execute the following on the web server to check where is the issue

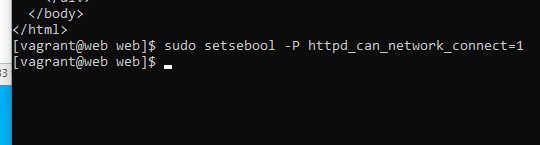
**curl** [**http://localhost**](http://localhost)

**php /var/www/html/index.php**

****

* Under **RedHat**-based distributions, we will see that in one of the attempts we can see the data and in the other – we cannot
* This is caused by the **SELinux** suite, and we can tackle the issue by executing the following

**sudo setsebool -P httpd\_can\_network\_connect=1**



Now is working

