

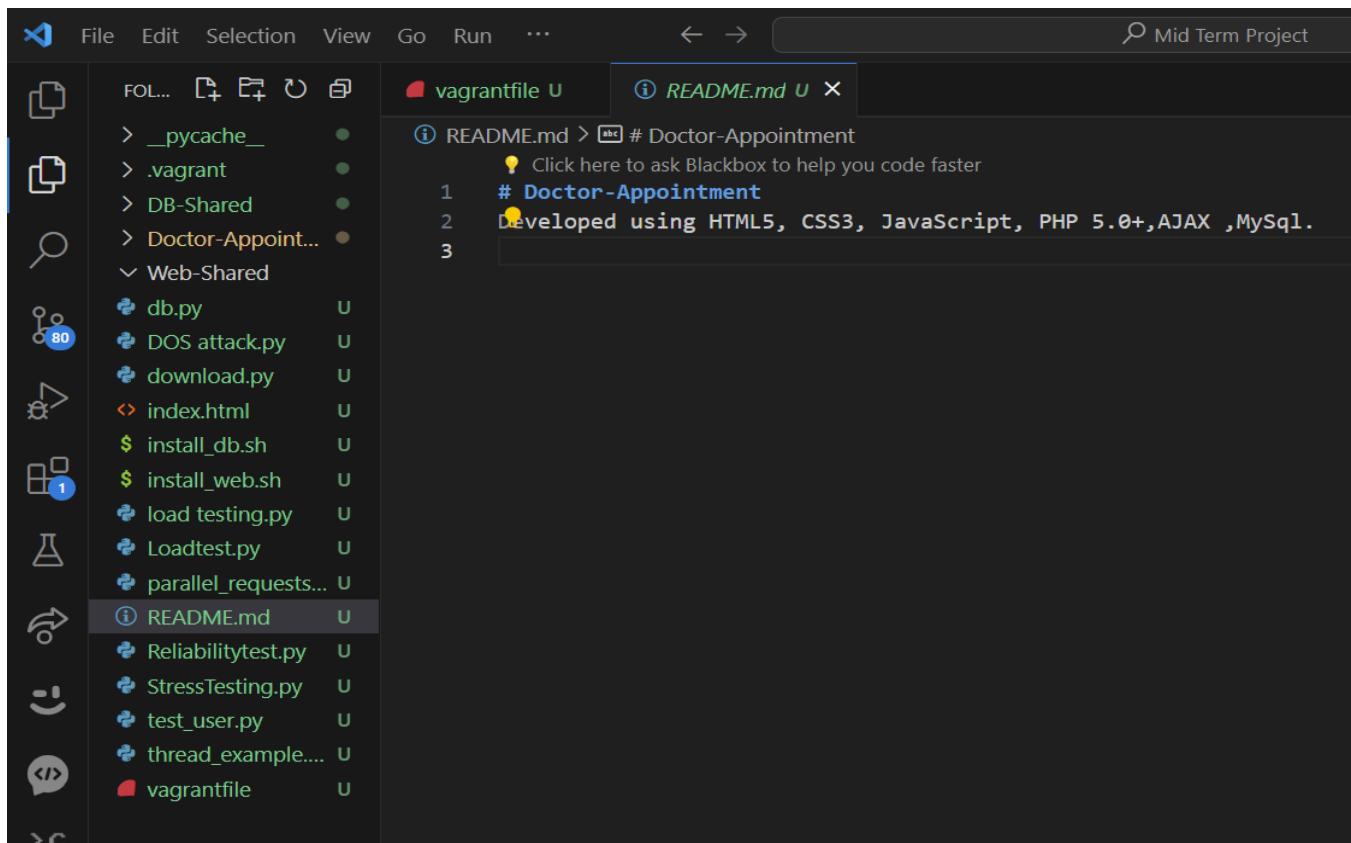
Final Project

NON-FUNCTIONAL TESTING

Student Name	Student ID
Shivani Varu	8941914
Anu Ajith	8809994
Nithin Varghese	8946846
Shikha Shah	8881100
Mohammed Rafique	8954785

1. Documentation Testing

Current README file:



The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with various icons representing file types and project structure. The main area shows a file named 'README.md' with the following content:

```
# Doctor-Appointment
Developed using HTML5, CSS3, JavaScript, PHP 5.0+,AJAX ,MySQL.
```

Doctor-Appointment

Developed using HTML5, CSS3, JavaScript, PHP 5.0+,AJAX ,MySQL

Mentioned and explained well:

Project Title and Description: Clearly stated what the project is about, and the technologies used.

Technologies Used: Listed the technologies utilized in the project.

Mentioned but could be explained better:

Features: Only mentioned "Book and manage doctor appointments." A more comprehensive feature list would be beneficial.

Installation: While it mentions installation steps, more detailed instructions and clarity would improve understanding.

Usage: Not mentioned, which is crucial for understanding how users can interact with the application.

Not mentioned but must be included:

Usage: Essential for understanding application functionality.

Contributing: Encourages community involvement and collaboration.

Troubleshooting: Helps users resolve issues independently.

Not mentioned but it would be nice to have them:

Testing: Information about testing methodologies and instructions for running tests.

Deployment: Guidance on deploying the application to different environments.

Security Considerations: Highlighting security best practices and considerations.

Proposed README File:

Project Title: Doctor-Appointment

Description: A web application for managing doctor appointments. Developed using HTML5, CSS3, JavaScript, PHP 5.0+, AJAX, and MySQL.

Features: Book and manage doctor appointments. Comprehensive feature list to be added.

Installation: Step-by-step instructions on setting up and running the project, including dependencies and database setup.

Usage: Explanation of how users can interact with the application, with screenshots or examples if possible.

Technologies Used: HTML5, CSS3, JavaScript, PHP 5.0+, AJAX, MySQL

Contributing: Guidelines for contributors, explaining how others can contribute to the project.

Troubleshooting: List of common issues users might face and their solutions.

License: Specify the project's license (e.g., MIT, GPL) and provide a link to the full license text.

Acknowledgments: Mention any sources, libraries, or individuals that contributed to the project.

Contact: Provide contact information for project maintainers or developers.

2. Installation script

Install_web.sh

```
$ install_web.sh
    Click here to ask Blackbox to help you code faster
1 # Update the package list to get the latest versions of the packages and their dependencies
2 sudo apt update
3
4 # Install apache2 and mysql-client packages
5 | sudo apt install -y apache2 mysql-client
6
7 # Stop the apache2 service
8 | sudo systemctl stop apache2
9
10 # Install necessary packages to add new repository over HTTPS
11 sudo apt install lsb-release ca-certificates apt-transport-https software-properties-common -y
12
13 # Add the repository for PHP 8.0
14 | sudo add-apt-repository ppa:ondrej/php
15
16 # Install PHP 8.0
17 | sudo apt install php8.0 -y
18
19 # Install additional PHP 8.0 modules required for web development
20 | sudo apt install -y php8.0-cli php8.0-common php8.0-mysql php8.0-zip php8.0-gd php8.0-mbstring php8.0-curl ph
21
22 # Start the apache2 service again
23 | sudo systemctl start apache2
```

Install_db.sh

```
$ install_db.sh
    Click here to ask Blackbox to help you code faster
1#!/bin/bash
2
3 # Update the package lists for upgrades and new package installations
4 sudo apt -y update
5 | # sudo debconf-set-selections <<< "mysql-server mysql-server/root_password password root"
6 | # sudo debconf-set-selections <<< "mysql-server mysql-server/root_password_again password root"
7
8 # Install MySQL server package
9 | sudo apt install -y mysql-server
10
11 # Create a new database called 'hospital'
12 | sudo mysql -u root -e "CREATE DATABASE hospital"
13
14 # Create a new MySQL user 'root' with access from the IP '192.168.33.20' and password 'root'
15 | sudo mysql -u root -e "CREATE USER 'root'@'192.168.33.20' IDENTIFIED BY 'root';"
16
17 # Grant the user 'root' all privileges on all databases and tables from the IP '192.168.33.20'
18 | sudo mysql -u root -e "GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES, RELOAD on *.*"
19
20 # Apply the new privileges immediately
21 | sudo mysql -u root -e "FLUSH PRIVILEGES;"
22
23 # Import the 'hospital' database schema from the provided SQL file
24 | sudo mysql -u root hospital < /vagrant-db/hospital.sql
25
26 # Allow MySQL to listen on all interfaces by changing the bind-address in the configuration file
27 | sudo sed -i 's/^bind-address.*$/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
28 | sudo sed -i 's/^mysqlx-bind-address/#mysqlx-bind-address/' /etc/mysql/mysql.conf.d/mysqld.cnf
29
30 # Restart the MySQL service to apply the configuration changes
31 | sudo systemctl restart mysql
```

Installation Script inside webserver box

```
vagrant@WebServer:/var/ + - X
GNU nano 4.8                               install_web.sh                         Modified
# Update the package list to get the latest versions of the packages and their dependencies
sudo apt update

# Install apache2 and mysql-client packages
sudo apt install -y apache2 mysql-client

# Stop the apache2 service
sudo systemctl stop apache2

# Install necessary packages to add new repository over HTTPS
sudo apt install lsb-release ca-certificates apt-transport-https software-properties-common -y

# Add the repository for PHP 8.0
sudo add-apt-repository ppa:ondrej/php

# Install PHP 8.0
sudo apt install php8.0 -y

# Install additional PHP 8.0 modules required for web development
sudo apt install -y php8.0-cli php8.0-common php8.0-mysql php8.0-zip php8.0-gd php8.0-mbstring php8.0-curl php8.0-xml

# Start the apache2 service again
sudo systemctl start apache2

^G Get Help   ^O Write Out   ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   M-U Undo
^X Exit       ^R Read File   ^\ Replace    ^U Paste Text  ^T To Spell   ^_ Go To Line M-E Redo
                                         M-A Mark Text  M-6 Copy Text
```

Installation Script inside dbserver box

```
vagrant@DBServer:~ + - X
GNU nano 4.8                               install_db.sh                         Modified
#!/bin/bash

# Update the package lists for upgrades and new package installations
sudo apt -y update
    # sudo debconf-set-selections <<< "mysql-server mysql-server/root_password password root"
    # sudo debconf-set-selections <<< "mysql-server mysql-server/root_password_again password root"

# Install MySQL server package
sudo apt install -y mysql-server

# Create a new database called 'hospital'
sudo mysql -u root -e "CREATE DATABASE hospital"

# Create a new MySQL user 'root' with access from the IP '192.168.33.20' and password 'root'
sudo mysql -u root -e "CREATE USER 'root'@'192.168.33.20' IDENTIFIED BY 'root';"

# Grant the user 'root' all privileges on all databases and tables from the IP '192.168.33.20'
sudo mysql -u root -e "GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES, INDEX, SUPER ON *.* TO 'root'@'192.168.33.20' WITH GRANT OPTION;" >

# Apply the new privileges immediately
sudo mysql -u root -e "FLUSH PRIVILEGES;"

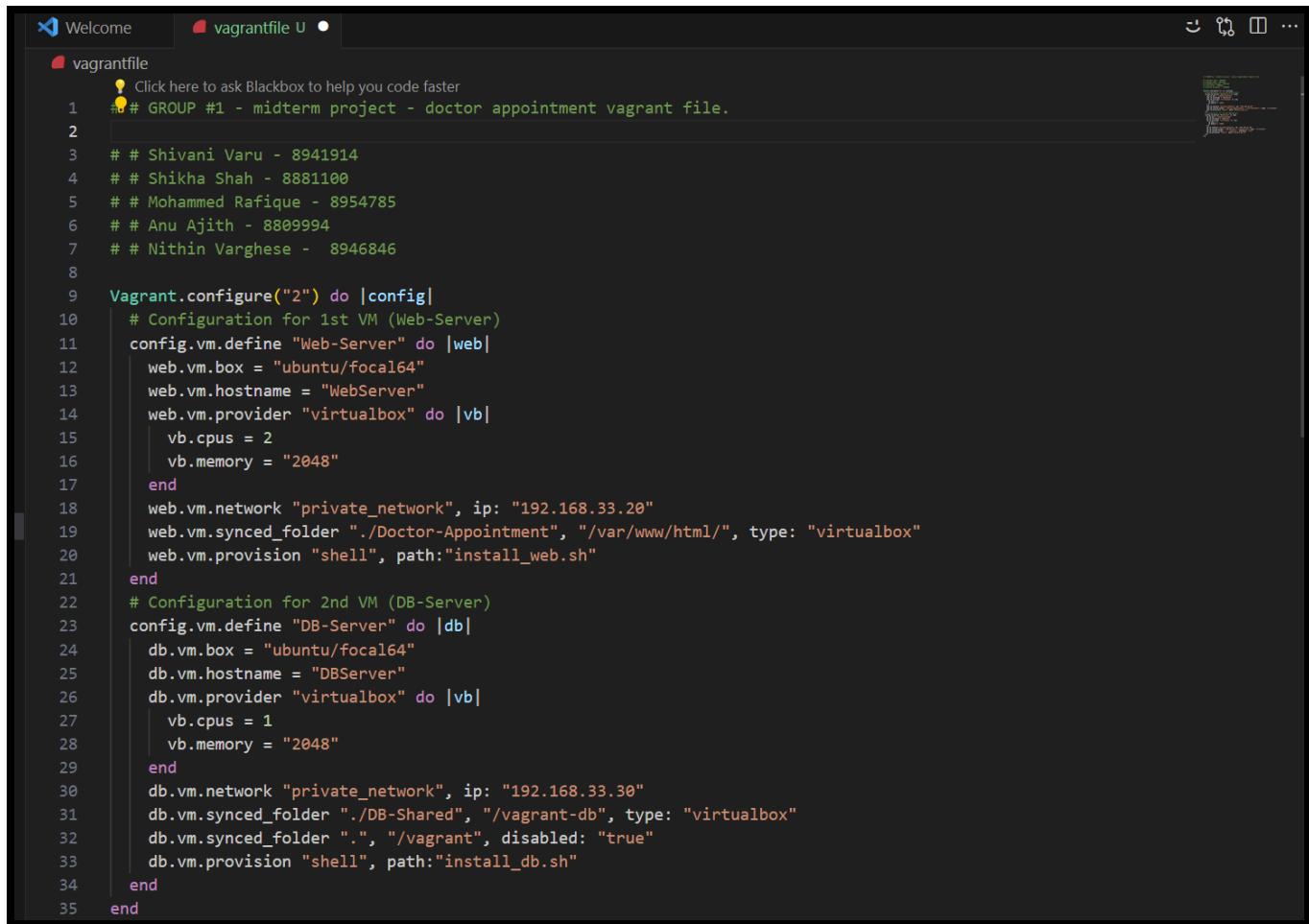
# Import the 'hospital' database schema from the provided SQL file
sudo mysql -u root hospital < /vagrant-db/hospital.sql

# Allow MySQL to listen on all interfaces by changing the bind-address in the configuration file
sudo sed -i 's/'bind-address.*'/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
sudo sed -i 's/^mysqlx-bind-address/#mysqlx-bind-address/' /etc/mysql/mysql.conf.d/mysqld.cnf

# Restart the MySQL service to apply the configuration changes
sudo systemctl restart mysql

^G Get Help   ^O Write Out   ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   M-U Undo
^X Exit       ^R Read File   ^\ Replace    ^U Paste Text  ^T To Spell   ^_ Go To Line M-E Redo
```

Updated Vagrant file for running installation script.



The screenshot shows a code editor window with a dark theme. The title bar says "Welcome" and "vagrantfile U". The main area contains the Vagrantfile code. The code defines two VMs: "Web-Server" and "DB-Server". The "Web-Server" VM uses "ubuntu/focal64" as its base image, has 2 CPUs, 2048 MB of memory, and is provisioned with "install_web.sh". It also has a private network interface with IP 192.168.33.20 and syncs the "/Doctor-Appointment" folder to "/var/www/html". The "DB-Server" VM uses "ubuntu/focal64" as its base image, has 1 CPU, 2048 MB of memory, and is provisioned with "install_db.sh". It also has a private network interface with IP 192.168.33.30 and syncs the "/DB-Shared" and "." folders to "/vagrant-db" and "/vagrant" respectively, with the latter being disabled.

```
1  # GROUP #1 - midterm project - doctor appointment vagrant file.
2
3  # # Shivani Varu - 8941914
4  # # Shikha Shah - 8881100
5  # # Mohammed Rafique - 8954785
6  # # Anu Ajith - 8809994
7  # # Nithin Varghese - 8946846
8
9  Vagrant.configure("2") do |config|
10    # Configuration for 1st VM (Web-Server)
11    config.vm.define "Web-Server" do |web|
12      web.vm.box = "ubuntu/focal64"
13      web.vm.hostname = "WebServer"
14      web.vm.provider "virtualbox" do |vb|
15        vb.cpus = 2
16        vb.memory = "2048"
17      end
18      web.vm.network "private_network", ip: "192.168.33.20"
19      web.vm.synced_folder "./Doctor-Appointment", "/var/www/html/", type: "virtualbox"
20      web.vm.provision "shell", path:"install_web.sh"
21    end
22    # Configuration for 2nd VM (DB-Server)
23    config.vm.define "DB-Server" do |db|
24      db.vm.box = "ubuntu/focal64"
25      db.vm.hostname = "DBServer"
26      db.vm.provider "virtualbox" do |vb|
27        vb.cpus = 1
28        vb.memory = "2048"
29      end
30      db.vm.network "private_network", ip: "192.168.33.30"
31      db.vm.synced_folder "./DB-Shared", "/vagrant-db", type: "virtualbox"
32      db.vm.synced_folder ".", "/vagrant", disabled: "true"
33      db.vm.provision "shell", path:"install_db.sh"
34    end
35  end
```

3.Efficiency Testing

- For stress testing we have used Apache benchmark tool. we have installed ab on virtual machine with command:

```
sudo apt-get install apache2-utils
```

- We did stress test by sending 1000 request with a concurrency of 10:

```
ab -n 1000 -c 100 http://192.168.33.20/Backend
```

```
vagrant@WebServer:~$ ab -n 1000 -c 100 http://192.168.33.20/Backend
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.33.20 (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests

Server Software:      Apache/2.4.41
Server Hostname:     192.168.33.20
Server Port:          80

Document Path:        /Backend
Document Length:      316 bytes

Concurrency Level:    100
Time taken for tests: 0.331 seconds
Complete requests:   1000
Failed requests:      0
Non-2xx responses:   1000
Total transferred:   545000 bytes
HTML transferred:    316000 bytes
Requests per second: 3019.45 [#/sec] (mean)
Time per request:    33.119 [ms] (mean)
Time per request:    0.331 [ms] (mean, across all concurrent requests)
Transfer rate:        1607.03 [Kbytes/sec] received
```

Concurrency level

Concurrency is used for number of multiple requests to make at a time.

```
LEMS OUTPUT TERMINAL PORTS SEARCH ERROR
▽ TERMINAL
Concurrency Level:      100
Time taken for tests:  0.331 seconds
Complete requests:     1000
Failed requests:       0
Non-2xx responses:    1000
Total transferred:    545000 bytes
HTML transferred:     316000 bytes
Requests per second:   3019.45 [#/sec] (mean)
Time per request:     33.119 [ms] (mean)
Time per request:     0.331 [ms] (mean, across all concurrent requests)
Transfer rate:        1607.03 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    1  1.7     1     6
Processing:     5   29  7.0    29    48
Waiting:        4   28  7.1    29    44
Total:         10   31  6.8    30    49

Percentage of the requests served within a certain time (ms)
  50%    30
  66%    32
  75%    34
  80%    35
  90%    41
  95%    43
  98%    44
  99%    45
100%    49 (longest request)
vagrant@WebServer:~$
```

Stress testing: While the stress test is running, we simultaneously monitored performance using the “top” command for monitoring CPU utilization.

```
Tasks: 116 total, 1 running, 115 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1971.4 total, 943.4 free, 163.8 used, 864.2 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 1637.4 avail Mem

Unknown command - try 'h' for help
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
501 root rt 0 345880 18272 8308 S 0.3 0.9 0:17.49 multipathd
1 root 20 0 169480 12908 8448 S 0.0 0.6 0:19.72 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.08 kthread
3 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_gp
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_par_gp
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:0H-kblockd
8 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 mm_percpu_wq
9 root 20 0 0 0 0 S 0.0 0.0 0:01.39 ksoftirqd/0
10 root 20 0 0 0 0 I 0.0 0.0 0:03.70 rcu_sched
11 root rt 0 0 0 0 S 0.0 0.0 0:00.70 migration/0
12 root -51 0 0 0 0 S 0.0 0.0 0:00.00 idle_inject/0
14 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/0
15 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/1
16 root -51 0 0 0 0 S 0.0 0.0 0:00.00 idle_inject/1
17 root rt 0 0 0 0 S 0.0 0.0 0:01.45 migration/1
18 root 20 0 0 0 0 S 0.0 0.0 0:04.77 ksoftirqd/1
20 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/1:0H-kblockd
21 root 20 0 0 0 0 S 0.0 0.0 0:00.08 kdevtmpfs
22 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 netns
23 root 20 0 0 0 0 S 0.0 0.0 0:00.00 rcu_tasks_kthre
24 root 20 0 0 0 0 S 0.0 0.0 0:00.01 kaudit
26 root 20 0 0 0 0 S 0.0 0.0 0:00.01 khungtaskd
27 root 20 0 0 0 0 S 0.0 0.0 0:00.00 oom_reaper
28 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 writeback
29 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kcompactd0
30 root 25 5 0 0 0 S 0.0 0.0 0:00.00 ksmd
31 root 39 19 0 0 0 S 0.0 0.0 0:00.00 khugepaged
77 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 integrityd
78 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kblockd
79 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 blkcg_punt_bio
80 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 tpm_dev_wq
78 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kblockd
```

Based on our analysis, the overall CPU utilization appears to be quite low. The highest CPU utilization is attributed to the multipath process with PID 501, which consumes approximately 0.3% of CPU resources. Even the system process (PID 1) is utilizing only a small amount of CPU resources. Most other processes show negligible CPU usage, indicated by 0.0% usage. From this perspective, our system does not seem to be under heavy CPU load. It suggests that the stress test may not be exerting significant pressure on the CPU, indicating that our system is capable of efficiently handling the workload.

Free -h command

```
vagrant@webServer:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:       1.9Gi       163Mi     943Mi      3.0Mi     864Mi       1.6Gi
Swap:          0B         0B         0B

vagrant@webServer:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:       1.9Gi       174Mi     930Mi      3.0Mi     865Mi       1.6Gi
Swap:          0B         0B         0B

vagrant@webServer:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:       1.9Gi       175Mi     928Mi      3.0Mi     866Mi       1.6Gi
Swap:          0B         0B         0B

vagrant@webServer:~$ ab -n 5000 -c 100 http://192.168.33.20/
[Output from ApacheBench 2.3 showing results for 5000 requests over 1.359 seconds with 5000 complete requests, 0 failed, and 0 non-2xx responses. Total transferred: 27258000 bytes, HTML transferred: 1580000 bytes, Requests per second: 3680.46 [#/sec] (mean), Time per request: 27.170 [ms] (mean), Time per request: 0.272 [ms] (mean, across all concurrent requests), Transfer rate: 1958.84 [Kbytes/sec] received]

Connection Times (ms)
                  min  mean[+/-sd] median   max
Connect:          0    0  0.6    0     6
Processing:      2   26  6.7   25   114
Waiting:         2   26  6.5   25   113
Total:           5   27  6.6   26   114

Percentage of the requests served within a certain time (ms)
  50%   26
  66%   29
  75%   30
  80%   32
  90%   35
  95%   38
  98%   43
  99%   45
 100%  114 (longest request)
```

Analysis of above Efficiency Testing:

We found that the stress test results with 5000 total requests and 100 concurrent requests at the Apache Benchmark (ab) concurrency level produced encouraging results. We successfully completed all 5000 requests without any errors, demonstrating the server's durability under demand.

Importantly, our server did very well during the stress test, averaging 3523.16 requests per second and responding to them in an average of 28.384 milliseconds. Our server can handle many requests with efficiency, as these numbers show.

processes consuming only 0.3% of CPU resources. The user CPU time (us) was continuously low, at 0.3%, suggesting that the CPU was not under any stress. This shows that the CPU resources in our system are not being used to their full potential, providing flexibility to handle heavier workloads, or scaling up resources as needed.

Memory use was also extremely low, with 920.9 MiB of free memory out of a total of 1971.4 MiB. We may conclude that our present memory allocation is sufficient for the workload analyzed because there is a lot of available memory, which suggests the stress test did not significantly affect our system's memory resources.

Given the modest CPU and memory consumption seen during the stress test, our current resource allocation for the VMs looks to be appropriate. However, we should frequently check resource utilization and scale up resources as needed to support potential expansion or higher workload demands.

4. Scalability and Performance Improvement

On Database server we have installed below command:

```
sudo apt update
```

```
sudo apt install -y apache2 php libapache2-mod-php
```

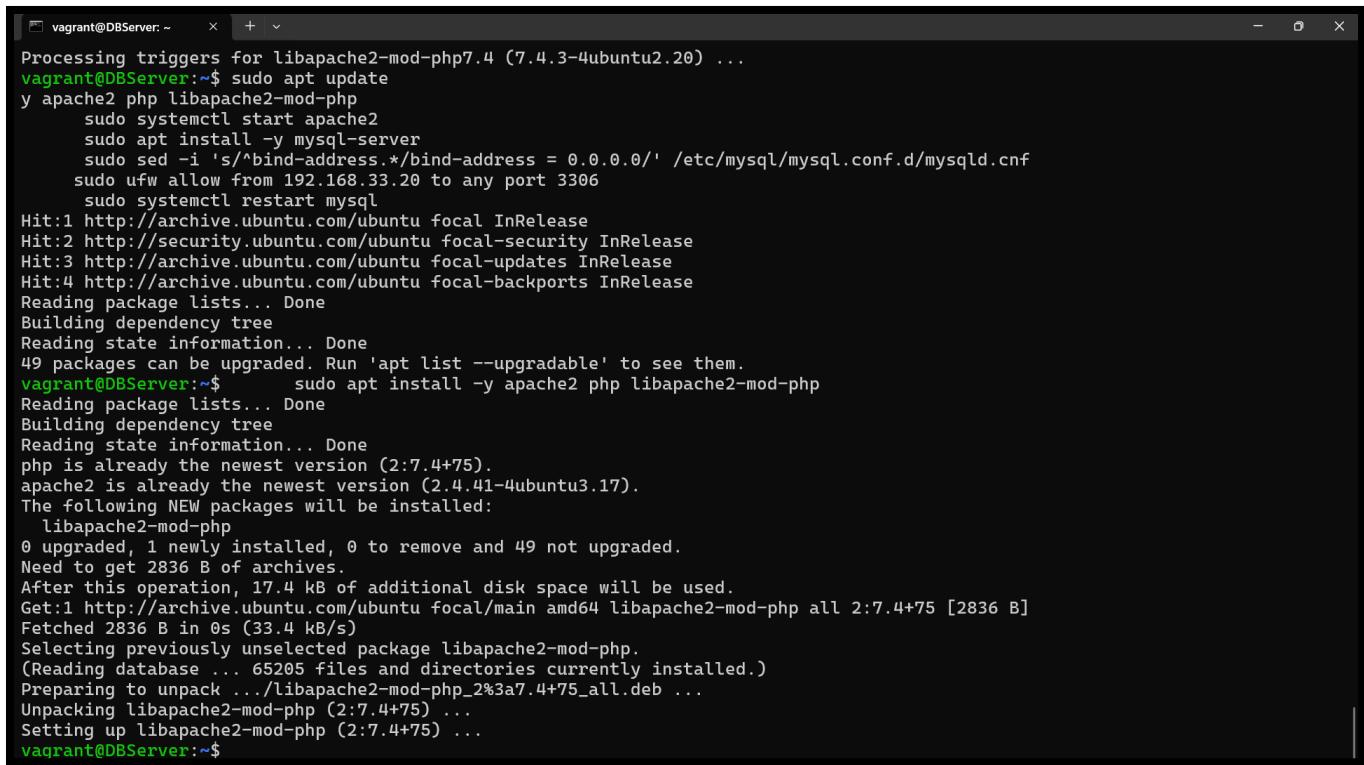
```
sudo systemctl start apache2
```

```
sudo apt install -y mysql-server
```

```
sudo sed -i 's/^bind-address.*$/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
sudo ufw allow from 192.168.33.20 to any port 3306
```

```
sudo systemctl restart mysql
```



```
vagrant@DBServer:~$ sudo apt update
y apache2 php libapache2-mod-php
  sudo systemctl start apache2
  sudo apt install -y mysql-server
  sudo sed -i 's/^bind-address.*$/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
  sudo ufw allow from 192.168.33.20 to any port 3306
  sudo systemctl restart mysql
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
49 packages can be upgraded. Run 'apt list --upgradable' to see them.
vagrant@DBServer:~$      sudo apt install -y apache2 php libapache2-mod-php
Reading package lists... Done
Building dependency tree
Reading state information... Done
php is already the newest version (2:7.4+75).
apache2 is already the newest version (2.4.41-4ubuntu3.17).
The following NEW packages will be installed:
  libapache2-mod-php
0 upgraded, 1 newly installed, 0 to remove and 49 not upgraded.
Need to get 2836 B of archives.
After this operation, 17.4 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/main amd64 libapache2-mod-php all 2:7.4+75 [2836 B]
Fetched 2836 B in 0s (33.4 kB/s)
Selecting previously unselected package libapache2-mod-php.
(Reading database ... 65205 files and directories currently installed.)
Preparing to unpack .../libapache2-mod-php_2%3a7.4+75_all.deb ...
Unpacking libapache2-mod-php (2:7.4+75) ...
Setting up libapache2-mod-php (2:7.4+75) ...
vagrant@DBServer:~$
```

On the DB box (DB-Server), we install and configure Apache2 with PHP module to serve web applications.

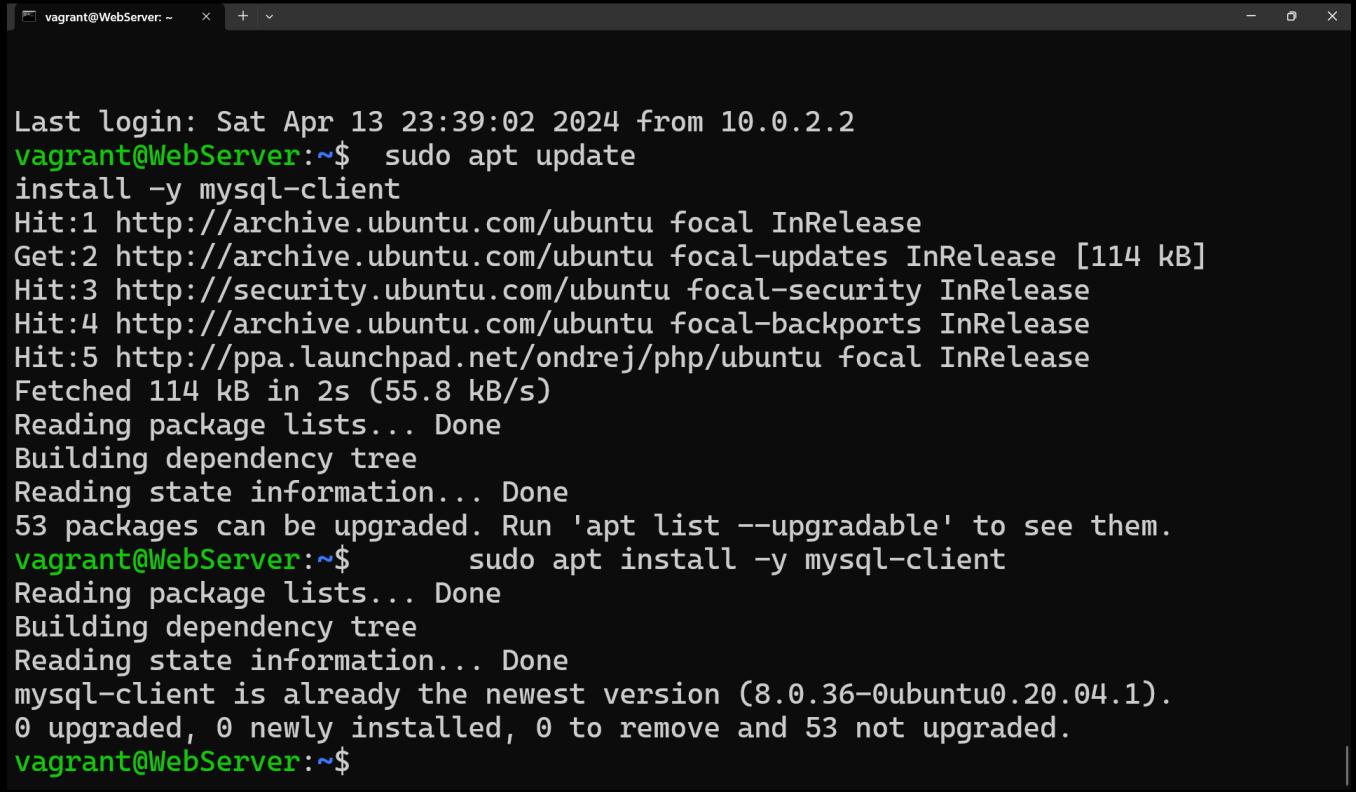
Additionally, we install MySQL server and configure it to listen on all interfaces to accept remote connections. We also open the MySQL port (3306) in the firewall to allow connections from the Web box (Webserver).

Install database server on web box

With the help of below command we updated the package list and install MySQL server:

```
sudo apt update
```

```
sudo apt install -y mysql-client
```



The screenshot shows a terminal window titled "vagrant@WebServer: ~". The terminal displays the following command-line session:

```
Last login: Sat Apr 13 23:39:02 2024 from 10.0.2.2
vagrant@WebServer:~$ sudo apt update
install -y mysql-client
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:3 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:5 http://ppa.launchpad.net/ondrej/php/ubuntu focal InRelease
Fetched 114 kB in 2s (55.8 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
53 packages can be upgraded. Run 'apt list --upgradable' to see them.
vagrant@WebServer:~$ sudo apt install -y mysql-client
Reading package lists... Done
Building dependency tree
Reading state information... Done
mysql-client is already the newest version (8.0.36-0ubuntu0.20.04.1).
0 upgraded, 0 newly installed, 0 to remove and 53 not upgraded.
vagrant@WebServer:~$
```

On the Web box (Webserver), we install the MySQL client to interact with the MySQL server installed on the DB box.