



Containerized CRUD Application with Automation Scripts

By: Salma Salah

Project's Description

This project is a containerized CRUD (Create, Read, Update, Delete) application that includes a frontend and backend, which interacts with a MongoDB database. The application is built using Docker, Docker-Compose, and is deployed on an AWS EC2 instance running Ubuntu. Also, a daily backup of the MongoDB database is automated using a cron job. Use Monitoring tools to track the application's performance.

Part 01

Deployment

Steps

- I. Creating CRUD App
- II. Confirming that application is working locally
- III. Dockerize the application in three containers; the first for the application front end , the second for API and third for the database
- IV. Running the application using Docker-Compose
- V. Pushing the Application into git repo
- VI. Clon application into Main Server on AWS
- VII. Write scripts to automate building, running, and stopping the Docker containers
- VIII. Write scripts to transfer code to target server and build/run the Docker image on the VM
- IX. Write scripts to ensure successful deployment.
- X. Implement an Automated Daily Task by Automatically back up the application's data once per day.

1- Docker file for frontend

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "NODE-API-MONGO-APP". The "frontend" folder is expanded, showing "public", "src", ".dockerignore", ".env.development", ".gitignore", "Dockerfile", "package-lock.json", "package.json", "README.md", ".dockerignore", ".gitignore", "docker-compose.yml", and "README.md".
- Editor:** The "Dockerfile M" tab is active, displaying the following Dockerfile content:

```
FROM node:14-alpine
WORKDIR /app
# add '/app/node_modules/.bin' to $PATH
ENV PATH /app/node_modules/.bin:$PATH
# install application dependencies
COPY package*.json .
# RUN npm install react-scripts --force
RUN npm install
# copy app files
COPY . .
#Expose port 3000
EXPOSE 3000
CMD ["npm", "start"]
```

- Terminal:** The terminal tab is active, showing the command "PS F:\Node-API-Mongo-App\Node-API-Mongo-App>".
- Bottom Status Bar:** Shows the current file is "main*", with 0 changes, and the status "Ln 22, Col 1 Spaces: 4 UTF-8 LF Dockerfile". It also displays system icons for battery, signal, and time (11:52 AM, 10/12/2024).

2- Docker file for Backend

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure for "NODE-API-MONGO-APP" with a "backend" folder containing "models", "routes", "Dockerfile", "package-lock.json", "package.json", "server.js", "frontend", ".dockerignore", ".gitignore", "docker-compose.yml", and "README.md".
- Search Bar:** At the top center, it says "Node-API-Mongo-App".
- Tab Bar:** Shows ".env.development", "Dockerfile frontend M", "docker-compose.yml", and "Dockerfile backend M X".
- Dockerfile Content:** The "Dockerfile backend" tab is selected and displays the following code:

```
1 FROM node:10-alpine
2
3 WORKDIR /usr/src/app
4
5 COPY package*.json .
6
7 RUN npm install
8
9 COPY .
10
11 EXPOSE 3001
12
13 CMD ["node", "server.js"]
```
- Terminal:** At the bottom, the terminal shows "PS F:\Node-API-Mongo-App\Node-API-Mongo-App>".
- Bottom Status Bar:** Shows "Ln 15, Col 1 Spaces: 4 UTF-8 LF Dockerfile" and a clock indicating "11:53 AM 10/12/2024".

3- Docker-Compose file

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "NODE-API-MONGO-APP". It includes folders for "backend" and "frontend", files like ".env.development", "Dockerfile", "package-lock.json", "package.json", "server.js", ".dockerignore", ".gitignore", and "README.md".
- Search Bar (Top):** Displays "Node-API-Mongo-App".
- Code Editor (Center):** The active file is "docker-compose.yml". The code defines a multi-container application with three services: "web", "api", and "mongo". The "web" service depends on "api" and runs on port 3000. The "api" service depends on "mongo" and runs on port 3001. The "mongo" service uses the "mongo" image, restarts always, and maps the "/data/db" volume. It also sets environment variables for MongoDB root user and password. A note at the bottom suggests uncommenting lines for local database access.
- Bottom Status Bar:** Shows file paths (main*, docker-compose.yml), line and column counts (Ln 41, Col 1), and encoding (UTF-8). It also displays system status icons for battery, signal, and network.
- Bottom Taskbar:** Includes icons for File Explorer, Search, Task Manager, File, Home, Recent Items, and the current tab.

4- Running app

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the title bar "Node-API-Mongo-App". The Explorer sidebar on the left displays the project structure:

- backend
- routes
- JS todos.js
- models
- JS todo.js
- index.js
- Dockerfile
- {} package-lock.json
- {} package.json
- JS server.js
- frontend
- public
- src
- .dockerignore
- \$.env.development
- .gitignore
- Dockerfile
- {} package-lock.json
- {} package.json
- README.md
- .dockerignore
- .gitignore
- docker-compose.yml
- README.md

The "todos.js" file is currently selected in the code editor. The code implements an Express API for managing todos:

```
const express = require("express");
const router = express.Router();
const Todo = require("../models/todo");

// GET all todos
router.get("/", async (req, res) => {
  const todos = await Todo.find({ is_complete: false });
  res.send(todos);
});

// GET todo based on ID
router.get("/:id", async (req, res) => {
```

Below the code editor, the "TERMINAL" tab is active, showing the output of a "docker ps" command:

```
=> => writing image sha256:3862fb1ebcf82f90ecceb933933c0c7ac78bf063b619080 0.0s
=> => naming to docker.io/library/node-api-mongo-app-web 0.0s
[+] Running 5/5
✓ Network node-api-mongo-app_network-backend Created 0.3s
✓ Volume "node-api-mongo-app_mongodb_data" Created 0.0s
✓ Container node-api-mongo-app-mongo-1 Started 2.4s
✓ Container node-api-mongo-app-api-1 Started 3.9s
✓ Container node-api-mongo-app-web-1 Started 4.9s
PS F:\Node-API-Mongo-App\Node-API-Mongo-App> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
699d599185c6 node-api-mongo-app-web "docker-entrypoint.s..." 3 minutes ago Up 3 minutes 0.0.0.0:3000->3000/tcp node-api-mongo-app-web-1
c7e80193ee06 node-api-mongo-app-api "docker-entrypoint.s..." 3 minutes ago Up 3 minutes 0.0.0.0:3001->3001/tcp node-api-mongo-app-api-1
bb2d4ca4c382 mongo "docker-entrypoint.s..." 3 minutes ago Up 3 minutes 27017/tcp node-api-mongo-app-mongo-1
PS F:\Node-API-Mongo-App\Node-API-Mongo-App>
```

The status bar at the bottom shows the following information:

- Ln 1, Col 1
- Spaces: 2
- UTF-8
- LF
- JavaScript
- 23°C
- 2:30 AM
- 10/7/2024
- PRE

5- Application after running the three containers

The screenshot shows the Docker Desktop interface with the following details:

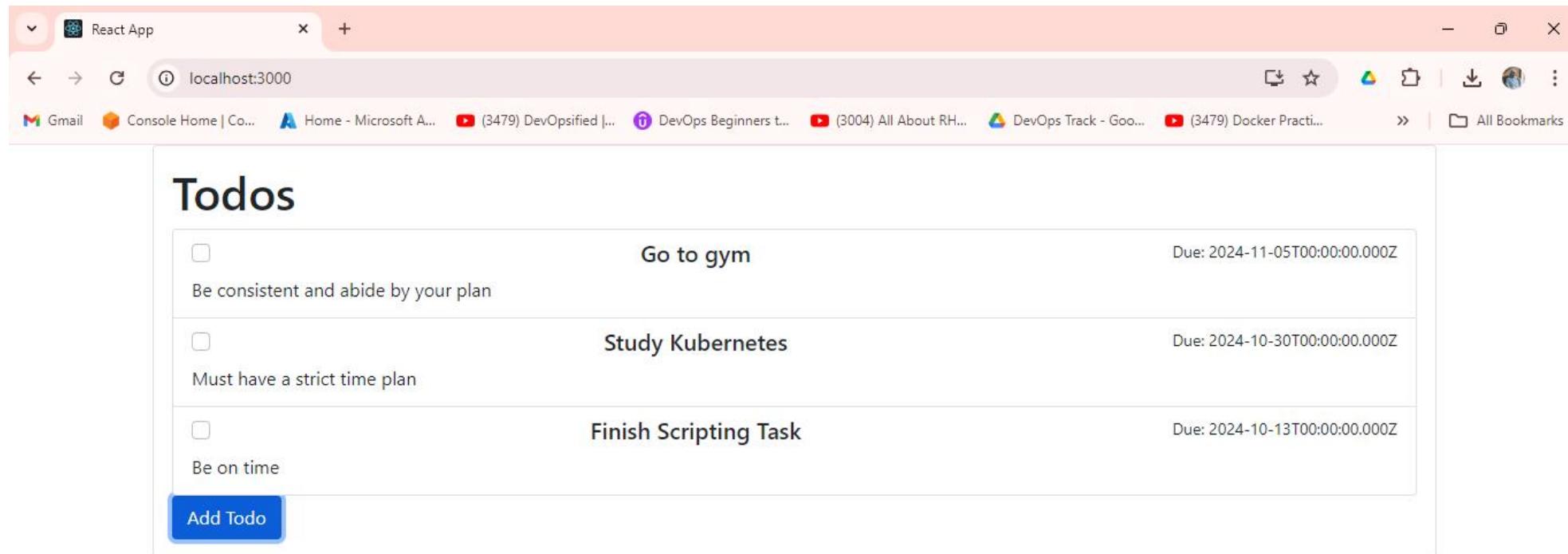
- Containers Tab:** Selected in the sidebar.
- Search Bar:** Shows "Search for images, containers, volumes, ext..." with a "Ctrl+K" hotkey.
- Metrics:** Container CPU usage: 1.09% / 800% (8 CPUs available) and Container memory usage: 285.33MB / 3.63GB. A "Show charts" link is present.
- Filter:** "Only show running containers" is selected.
- Table:** Displays the following container information:

Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
node-api-mongo-app		Running (3/3)		1.09%	5 minutes ago	[More]
api-1 c7e80193ee06	node-api-m	Running	3001:3001	0%	5 minutes ago	[More]
web-1 699d599185c6	node-api-m	Running	3000:3000	0%	5 minutes ago	[More]
mongo-1 bb2d4ca4c382	mongo	Running		1.09%	5 minutes ago	[More]

At the bottom, it says "Showing 4 items".

System Tray: Shows "Engine running", "23°C صاف غالباً", "RAM 2.27 GB CPU 0.25%", "New version available", "3 notifications", "2:29 AM 10/7/2024", and "ENG".

6- Frontend Api & DB working



7- Logging into db (todos)

The screenshot shows a VS Code interface with the title bar "Node-API-Mongo-App". The Explorer sidebar on the left shows a project structure for "NODE-API-MONGO-APP" with "backend" and "frontend" branches. The "backend" branch contains "models", "routes", "Dockerfile", "package-lock.json", "package.json", "server.js", ".dockerignore", and ".gitignore". The "frontend" branch contains "README.md". The "TERMINAL" tab is active, displaying the following MongoDB shell session:

```
2024-10-06T23:24:59.496+00:00: vm.max_map_count is too low
-----
test> show dbs
admin 40.00 KiB
config 92.00 KiB
local 40.00 KiB
todos 72.00 KiB
test> use todos
switched to db todos
todos> db.items.find().pretty()

todos> db.getCollectionInfos()
[
  {
    name: 'todos',
    type: 'collection',
    options: {},
    info: {
      readOnly: false,
      uuid: UUID('509c055d-cc63-4b1f-9f4a-442871f64848')
    },
    todos> db.products.find
```

The status bar at the bottom shows "Ln 39, Col 6 Spaces: 2 UTF-8 LF Compose". The system tray at the very bottom includes icons for weather (22°C), search, file explorer, file manager, phone, browser, and VS Code.

8- Adding more entries to our Todos

A screenshot of a web browser window titled "React App" displaying a todo list application at "localhost:3000". The page has a header "Todos" and a list of four tasks:

- Go to gym Due: 2024-11-05T00:00:00.000Z
Be consistent and abide by your plan
- Study Kubernetes Due: 2024-10-30T00:00:00.000Z
Must have a strict time plan
- Finish Scripting Task Due: 2024-10-13T00:00:00.000Z
Be on time
- Off-Line Soft Skills Due: 2024-10-13T00:00:00.000Z
Be on-time at 9:00 AM

A blue "Add Todo" button is located at the bottom left of the list.



9- Connecting to mongodb using MongoDB Compass (gui)

The screenshot shows the MongoDB Compass interface connected to the 'todos' database. The left sidebar lists connections, with 'localhost:27017' selected. Under 'todos', three documents are listed:

- `_id: ObjectId('67031ebe99a347448ed10811')`
title : "Study Kubernetes"
description : "Must have a strict time plan"
is_complete : false
due_date : 2024-10-30T00:00:00.000+00:00
__v : 0
- `_id: ObjectId('67031eea99a3474fe8d10812')`
title : "Finish Scripting Task"
description : "Be on time"
is_complete : false
due_date : 2024-10-13T00:00:00.000+00:00
__v : 0
- `_id: ObjectId('670325ece3ccc50d261e5f12')`
title : "Off-Line Soft Skills"
description : "Be on-time at 9:00 AM"
is_complete : false
due_date : 2024-10-13T00:00:00.000+00:00
__v : 0

At the bottom, the taskbar shows system icons and the date/time: 3:19 AM, 10/7/2024.

10- Create local and remote git repositories and push all files to remote repo

The screenshot shows a GitHub repository page for 'Node-API-Mongo-App'. The repository is public and has 1 branch and 0 tags. The main commit is by 'salmasalam024' titled 'BackUp file after exexuting backup script now data on database will b...' with a timestamp of 'b2661e1 · 17 hours ago' and 8 commits. Below it is a commit for 'BackUp-Dir/mongodump_20241011001034/to...'. The repository also contains folders for 'backend' and 'frontend', and files like '.dockerignore', '.gitignore', 'Automate-Daily-Backups.sh', and 'Copy_Automate_Deployment.sh'. The repository has 0 stars and 1 watching. The right sidebar provides an 'About' summary of the project: 'This project is a containerized CRUD (Create, Read, Update, Delete) application that includes a frontend and backend, which interacts with a MongoDB database. The application is built using Docker, Docker Compose, and is deployed on an AWS EC2 instance running Ubuntu. Also a daily backup of the MongoDB database is automated using a cron job.' The bottom of the screen shows a Windows taskbar with various icons and a system tray.

github.com/salmasalam024/Node-API-Mongo-App

Gmail Console Home | Co... Home - Microsoft A... (3479) DevOpsified |... DevOps Beginners t... (3004) All About RH... DevOps Track - Goo... Your Repositories All Bookmarks

salmasalam024 / Node-API-Mongo-App

Type / to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Node-API-Mongo-App Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags

Go to file Add file Code

salmasalam024 BackUp file after exexuting backup script now data on database will b... b2661e1 · 17 hours ago 8 Commits

BackUp-Dir/mongodump_20241011001034/to... BackUp file after exexuting backup script now data on datab... 17 hours ago

backend script to backup application data daily 2 days ago

frontend Bash scripts for Copying Deploying Testing and Stopping th... 3 days ago

.dockerignore Initial Commit 5 days ago

.gitignore Initial Commit 5 days ago

Automate-Daily-Backups.sh script to backup application data daily 2 days ago

Copy_Automate_Deployment.sh Bash scripts for Copying Deploying Testing and Stopping th... 3 days ago

About

This project is a containerized CRUD (Create, Read, Update, Delete) application that includes a frontend and backend, which interacts with a MongoDB database. The application is built using Docker, Docker Compose, and is deployed on an AWS EC2 instance running Ubuntu. Also a daily backup of the MongoDB database is automated using a cron job.

Readme

Activity

0 stars

1 watching

23°C صافي غالباً

Search

4:16 AM 10/13/2024

11- Creating Main Server on AWS (Ubuntu) and open required ports

The screenshot shows the AWS Management Console interface for the EC2 service. The main content area displays the 'Instance summary' for an instance named 'i-0fe9dc40a1294a815 (Main-Server)'. The instance is currently running. Key details shown include:

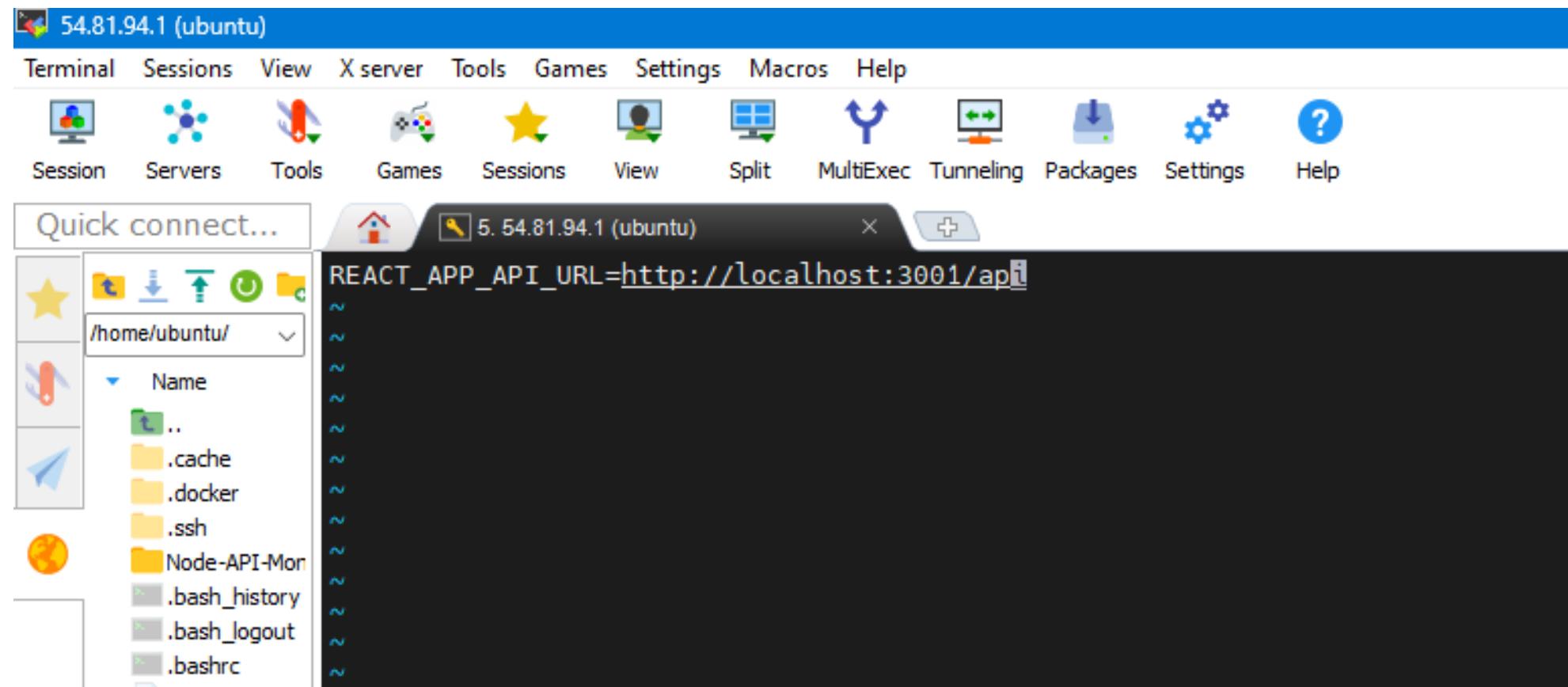
- Public IPv4 address:** 54.162.98.141 (with a link to 'open address')
- Private IPv4 addresses:** 172.31.32.116
- Public IPv4 DNS:** ec2-54-162-98-141.compute-1.amazonaws.com (with a link to 'open address')
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-172-31-32-116.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-07e368fb8901161b4 (with a link)
- Subnet ID:** subnet-0b61a6f56cdd003f5 (with a link)
- Instance ARN:** arn:aws:ec2:us-east-1:821594020462:instance/i-0fe9dc40a1294a815

The left sidebar shows the navigation menu for the EC2 service, including 'Instances' (selected), 'Images', and 'Elastic Block Store'.

12- Install Docker and Docker-Compose

```
100 11.6M 100 11.6M 0 0 31.2M 0 --:--:-- --:--:-- --:--:-- 31.2M
ubuntu@ip-172-31-32-116:~$ sudo chmod +x /usr/local/bin/docker-compose
ubuntu@ip-172-31-32-116:~$ docker-compose --version
docker-compose version 1.28.5, build c4eb3a1f
ubuntu@ip-172-31-32-116:~$ history
 1 sudo apt update
 2 sudo apt install apt-transport-https ca-certificates curl software-proper
ties-common
 3 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key ad
d -
 4 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/lin
ux/ubuntu focal stable"
 5 sudo apt update
 6 sudo apt install docker-ce
 7 sudo systemctl status docker
 8 sudo usermod -aG docker ${USER}
 9 groups
10 sudo curl -L "https://github.com/docker/compose/releases/download/1.28.5/
docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
11 sudo chmod +x /usr/local/bin/docker-compose
12 docker-compose --version
13 history
ubuntu@ip-172-31-32-116:~$
ubuntu@ip-172-31-32-116:~$
ubuntu@ip-172-31-32-116:~$
ubuntu@ip-172-31-32-116:~$
ubuntu@ip-172-31-32-116:~$
```

13- Note that at first the application was deployed on host machine so the API URL was <http://localhost:3001/api> but after deploying on the cloud vm (localhost) must be replaced by the public ip address of ec2 instance



13- API URL after modification with public ip address of ec2 (main server)

54.81.94.1 (ubuntu)

Terminal Sessions View X server Tools Games Settings Macros Help

Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages X server Exit

Quick connect... 5. 54.81.94.1 (ubuntu) +

Name

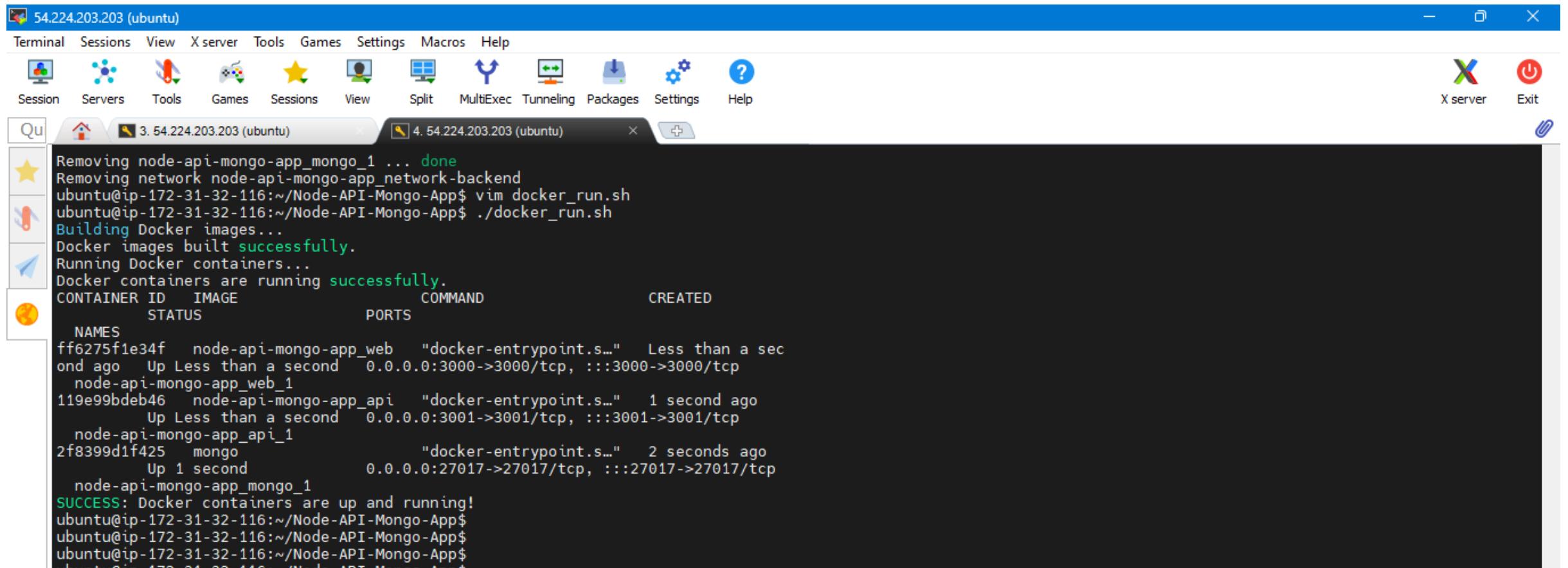
- ..
- .cache
- .docker
- .ssh
- Node-API-H
- .bash_hist
- .bash_logc
- .bashrc
- .gitconfig
- .profile
- .sudo_as_i
- .viminfo

REACT_APP_API_URL=http://54.81.94.1:3001/api

:wq

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

14- Running application from main server on aws



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "3. 54.224.203.203 (ubuntu)". The terminal content shows the following steps to run a Docker application:

```
Removing node-api-mongo-app_mongo_1 ... done
Removing network node-api-mongo-app_network-backend
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim docker_run.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ./docker_run.sh
Building Docker images...
Docker images built successfully.
Running Docker containers...
Docker containers are running successfully.
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
 NAMES
ff6275f1e34f      node-api-mongo-app_web   "docker-entrypoint.s..."   Less than a sec
ond ago           Up Less than a second  0.0.0.0:3000->3000/tcp, :::3000->3000/tcp
    node-api-mongo-app_web_1
119e99bdeb46      node-api-mongo-app_api   "docker-entrypoint.s..."   1 second ago
                Up Less than a second  0.0.0.0:3001->3001/tcp, :::3001->3001/tcp
    node-api-mongo-app_api_1
2f8399d1f425      mongo                 "docker-entrypoint.s..."   2 seconds ago
                Up 1 second            0.0.0.0:27017->27017/tcp, :::27017->27017/tcp
    node-api-mongo-app_mongo_1
SUCCESS: Docker containers are up and running!
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

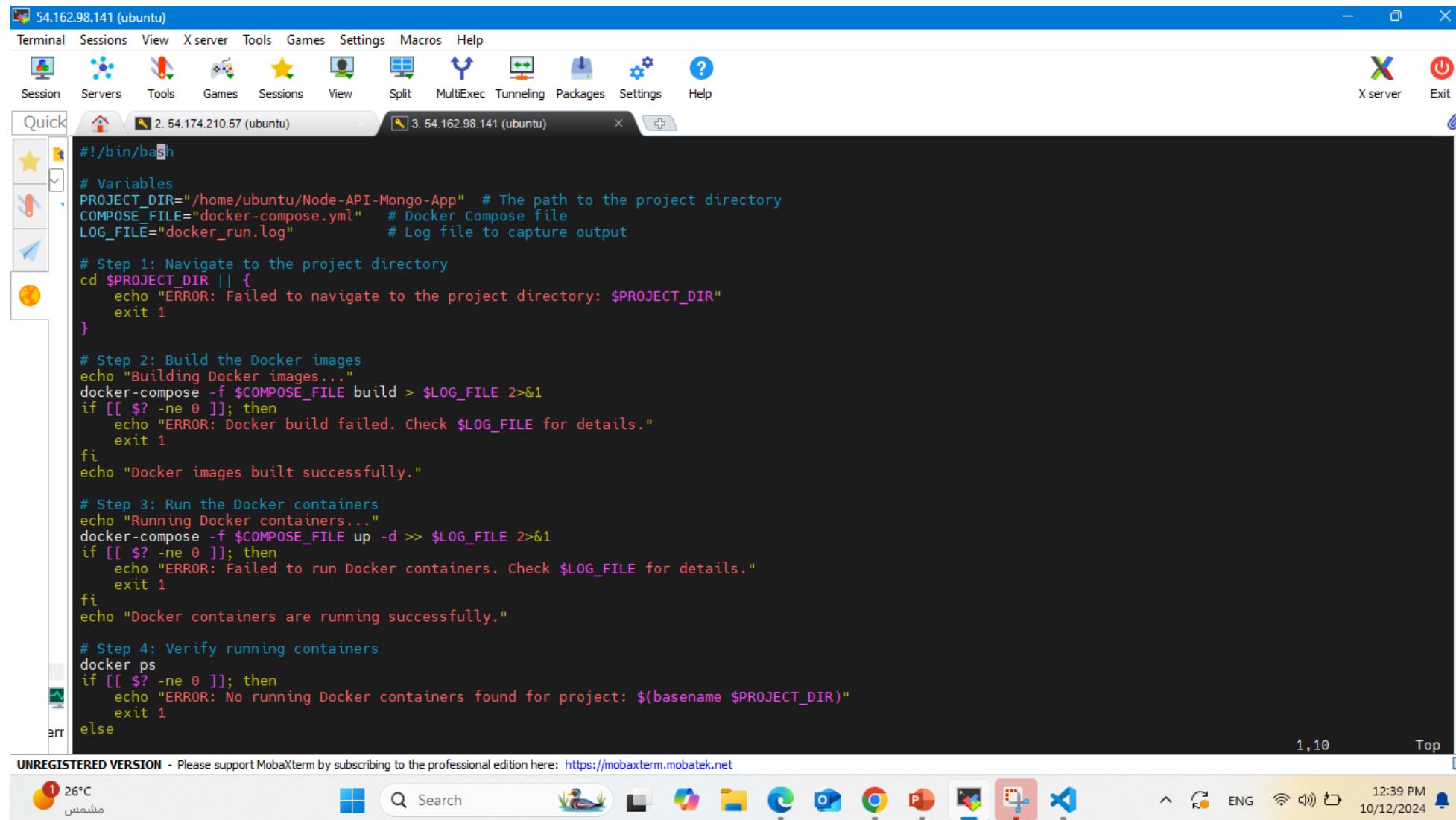
14- Application after deployment in main server on aws

The screenshot shows a web browser window with the following details:

- Title Bar:** React App
- Address Bar:** Not secure 54.81.94.1:3000
- Toolbar:** Back, Forward, Stop, Refresh, Home - Microsoft A..., YouTube, DevOpsified, DevOps Beginners t..., YouTube, All About RH..., DevOps Track - Goo..., Your Repositories, More, All Bookmarks.
- Content Area:**
 - Todos** heading
 - Problem Statement**:
 -
 - Due: 2024-10-10T00:00:00.000Z
 - Text: Here is an issue I faced that first code was on my local machine , After pushing it to github and cloning on aws ec2 as the main-server; Only Frontend was working but not sending API calls to Backend
 - Solution**:
 -
 - Due: 2024-10-10T00:00:00.000Z
 - Text: After troubleshooting, the issue was found that the React app API URL was set to http://localhost:3001/api and that was working on local machine and after running app on aws it has to be the Public IP address of the VM not localhost
- Buttons:** Add Todo



15- Writing Write scripts to automate building and running Docker containers



The screenshot shows a MobaXterm window titled "54.162.98.141 (ubuntu)". The terminal contains a shell script for automating Docker container management. The script includes steps for navigating to the project directory, building Docker images, running Docker containers, and verifying their status. It uses Docker Compose to handle the build and run processes.

```
#!/bin/bash

# Variables
PROJECT_DIR="/home/ubuntu/Node-API-Mongo-App" # The path to the project directory
COMPOSE_FILE="docker-compose.yml" # Docker Compose file
LOG_FILE="docker_run.log" # Log file to capture output

# Step 1: Navigate to the project directory
cd $PROJECT_DIR || {
    echo "ERROR: Failed to navigate to the project directory: $PROJECT_DIR"
    exit 1
}

# Step 2: Build the Docker images
echo "Building Docker images..."
docker-compose -f $COMPOSE_FILE build > $LOG_FILE 2>&1
if [[ $? -ne 0 ]]; then
    echo "ERROR: Docker build failed. Check $LOG_FILE for details."
    exit 1
fi
echo "Docker images built successfully."

# Step 3: Run the Docker containers
echo "Running Docker containers..."
docker-compose -f $COMPOSE_FILE up -d >> $LOG_FILE 2>&1
if [[ $? -ne 0 ]]; then
    echo "ERROR: Failed to run Docker containers. Check $LOG_FILE for details."
    exit 1
fi
echo "Docker containers are running successfully."

# Step 4: Verify running containers
docker ps
if [[ $? -ne 0 ]]; then
    echo "ERROR: No running Docker containers found for project: $(basename $PROJECT_DIR)"
    exit 1
else
    echo "Docker containers are running successfully."
fi
```

At the bottom of the terminal, a message reads: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>". The system tray at the bottom right shows the date and time as "10/12/2024 12:39 PM".

16- Writing a script to automate stopping the Docker containers

The screenshot shows a MobaXterm window with multiple sessions open. The current session is titled '54.162.98.141 (ubuntu)'. The terminal window displays a bash script for stopping Docker containers. The script sets variables for the project directory, Docker Compose file, and log file, then navigates to the project directory and uses docker-compose to stop the containers, logging the output to the specified log file.

```
#!/bin/bash

# Variables
PROJECT_DIR="/home/ubuntu/Node-API-Mongo-App" # Path to the project directory
COMPOSE_FILE="docker-compose.yml" # Docker Compose file
LOG_FILE="docker_stop.log" # Log file to capture output

# Step 1: Navigate to the project directory
cd $PROJECT_DIR || {
    echo "ERROR: Failed to navigate to the project directory: $PROJECT_DIR"
    exit 1
}

# Step 2: Stop Docker containers
echo "Stopping Docker containers..."
docker-compose -f $COMPOSE_FILE down > $LOG_FILE 2>&1
if [[ $? -ne 0 ]]; then
    echo "ERROR: Failed to stop Docker containers. Check $LOG_FILE for details."
    exit 1
fi

echo "Docker containers stopped and resources cleaned up successfully."
```

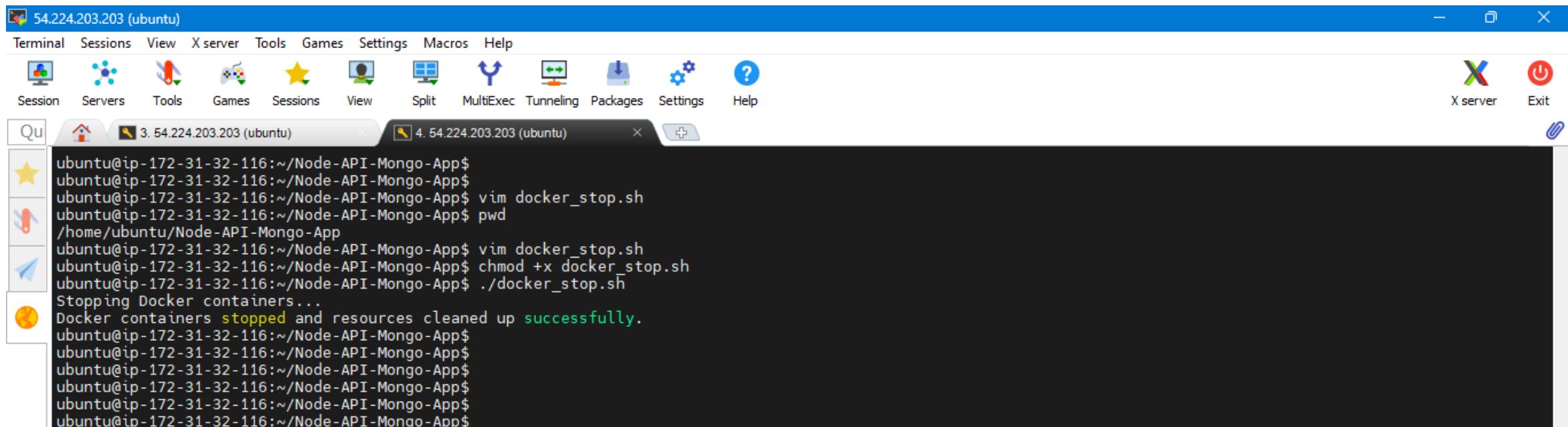
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

26°C مشرقي

Search

12:44 PM 10/12/2024

17- Give the running and stopping scripts execute permissions and testing their execution



The screenshot shows the Quassel IRC interface with a terminal session titled "4. 54.224.203.203 (ubuntu)". The terminal window contains the following command-line session:

```
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim docker_stop.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ chmod +x docker_stop.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ./docker_stop.sh
Stopping Docker containers...
Docker containers stopped and resources cleaned up successfully.
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

18- Creating ssh keys to authenticate with Github repo to push the new scripts

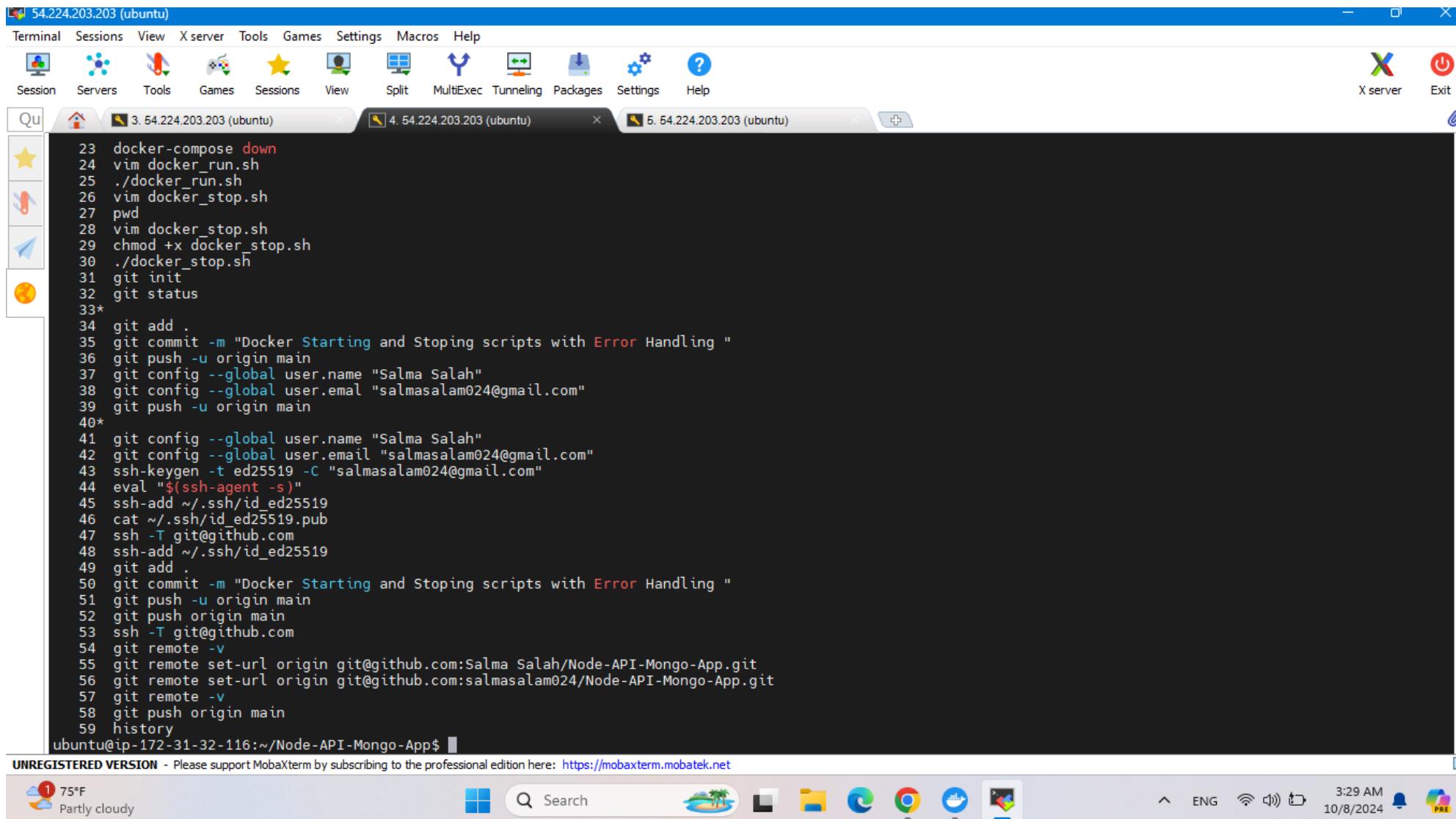
The screenshot shows a MobaXterm window with three tabs open, all connected to the same Ubuntu host (54.224.203.203). The central tab is active and displays the terminal session where the user is generating an SSH key:

```
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ssh-keygen -t ed25519 -C "salmasalam024@gmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:8ewkedaRxzzDogEBZL0ycWEqco8yuo9b2iJ4/6jdK7M salmasalam024@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
| .B |
| + = |
| . o + o.. |
| o + = =. |
| o . o S.= |
| . o B. ...o |
| o . . o....++ |
| +*..oo .o ..o |
| *+=oE=+. . |
+---[SHA256]----+
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ eval "$(ssh-agent -s)"
Agent pid 7600
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/ubuntu/.ssh/id_ed25519 (salmasalam024@gmail.com)
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIEMpfFQzDgclRglTB1CNT18JBisDirT32ZG/F1zRWCK salmasalam024@gmail.com
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ssh -T git@github.com
The authenticity of host 'github.com (140.82.114.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMsvHdkr4UvC0qu.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Hi salmasalam024! You've successfully authenticated, but GitHub does not provide shell access.
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/ubuntu/.ssh/id_ed25519 (salmasalam024@gmail.com)
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

At the bottom of the terminal, a message from MobaXterm encourages users to support the software by subscribing to the professional edition.

System tray icons at the bottom include: 1 notification (Partly cloudy), Search, File Explorer, VS Code, Google Chrome, Task View, Task Manager, 3:17 AM, ENG, WiFi, 10/8/2024, and PRE.

19- Successfully connecting to github repo and push script



The screenshot shows a MobaXterm window titled "54.224.203.203 (ubuntu)". The terminal session contains the following command history:

```
23 docker-compose down
24 vim docker_run.sh
25 ./docker_run.sh
26 vim docker_stop.sh
27 pwd
28 vim docker_stop.sh
29 chmod +x docker_stop.sh
30 ./docker_stop.sh
31 git init
32 git status
33*
34 git add .
35 git commit -m "Docker Starting and Stoping scripts with Error Handling"
36 git push -u origin main
37 git config --global user.name "Salma Salah"
38 git config --global user.emal "salmasalam024@gmail.com"
39 git push -u origin main
40*
41 git config --global user.name "Salma Salah"
42 git config --global user.email "salmasalam024@gmail.com"
43 ssh-keygen -t ed25519 -C "salmasalam024@gmail.com"
44 eval "$(ssh-agent -s)"
45 ssh-add ~/.ssh/id_ed25519
46 cat ~/.ssh/id_ed25519.pub
47 ssh -T git@github.com
48 ssh-add ~/.ssh/id_ed25519
49 git add .
50 git commit -m "Docker Starting and Stoping scripts with Error Handling"
51 git push -u origin main
52 git push origin main
53 ssh -T git@github.com
54 git remote -v
55 git remote set-url origin git@github.com:Salma Salah/Node-API-Mongo-App.git
56 git remote set-url origin git@github.com:salmasalam024/Node-API-Mongo-App.git
57 git remote -v
58 git push origin main
59 history
```

The command "git push -u origin main" was run at line 36, and "git push origin main" was run again at line 52. The command "git remote -v" was run at line 54, and "git remote -v" was run again at line 57.

At the bottom of the terminal, the prompt "ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App\$" is visible. The status bar at the bottom of the window displays "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

20- Creating Destination Server on AWS (Ubuntu) and open required ports

The screenshot shows the AWS EC2 Instances details page for an instance named "Destination_Server".

Instance summary for i-05f4ccfa52c8ef2c1 (Destination_Server)

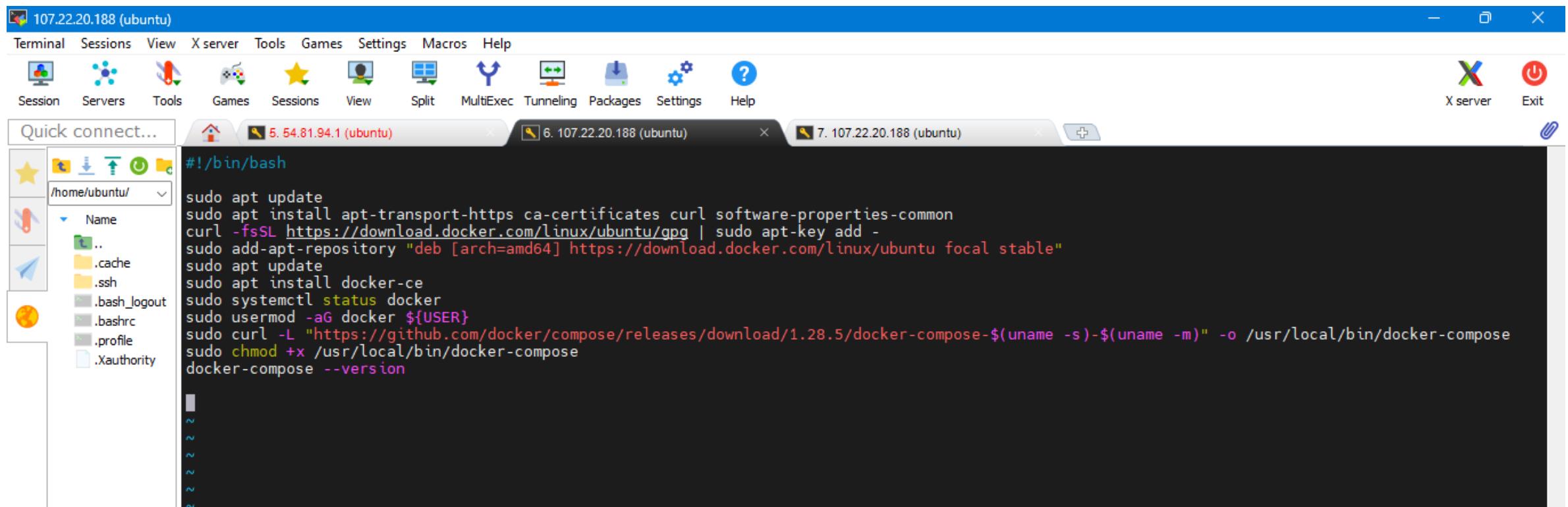
Attribute	Value
Instance ID	i-05f4ccfa52c8ef2c1 (Destination_Server)
Public IPv4 address	54.174.210.57 open address
Private IPv4 addresses	172.31.40.204
IPv6 address	-
Instance state	Running
Public IPv4 DNS	ec2-54-174-210-57.compute-1.amazonaws.com open address
Hostname type	IP name: ip-172-31-40-204.ec2.internal
Private IP DNS name (IPv4 only)	ip-172-31-40-204.ec2.internal
Answer private resource DNS name	IPv4 (A)
Instance type	t2.micro
Elastic IP addresses	It is taking a bit longer than usual to fetch your data
Auto-assigned IP address	It is taking a bit longer than usual to fetch your data
VPC ID	vpc-07e368fb8901161b4
AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations.
IAM Role	-
Subnet ID	subnet-0b61a6f56cd003f5
Auto Scaling Group name	-
IMDSv2	Required
Instance ARN	-

Actions: C, Connect, Instance state, Actions

EC2 Dashboard, **EC2 Global View**, **Events**, **Instances**: Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, **Images**: AMIs, AMI Catalog, **Elastic Block Store**: Volumes, Snapshots

CloudShell, Feedback, © 2024, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, Cookie preferences, 1:06 PM, ENG, 10/12/2024, 29°C, مشمس

21- Writing bash script to automate installation if docker and docker-compose on destination server

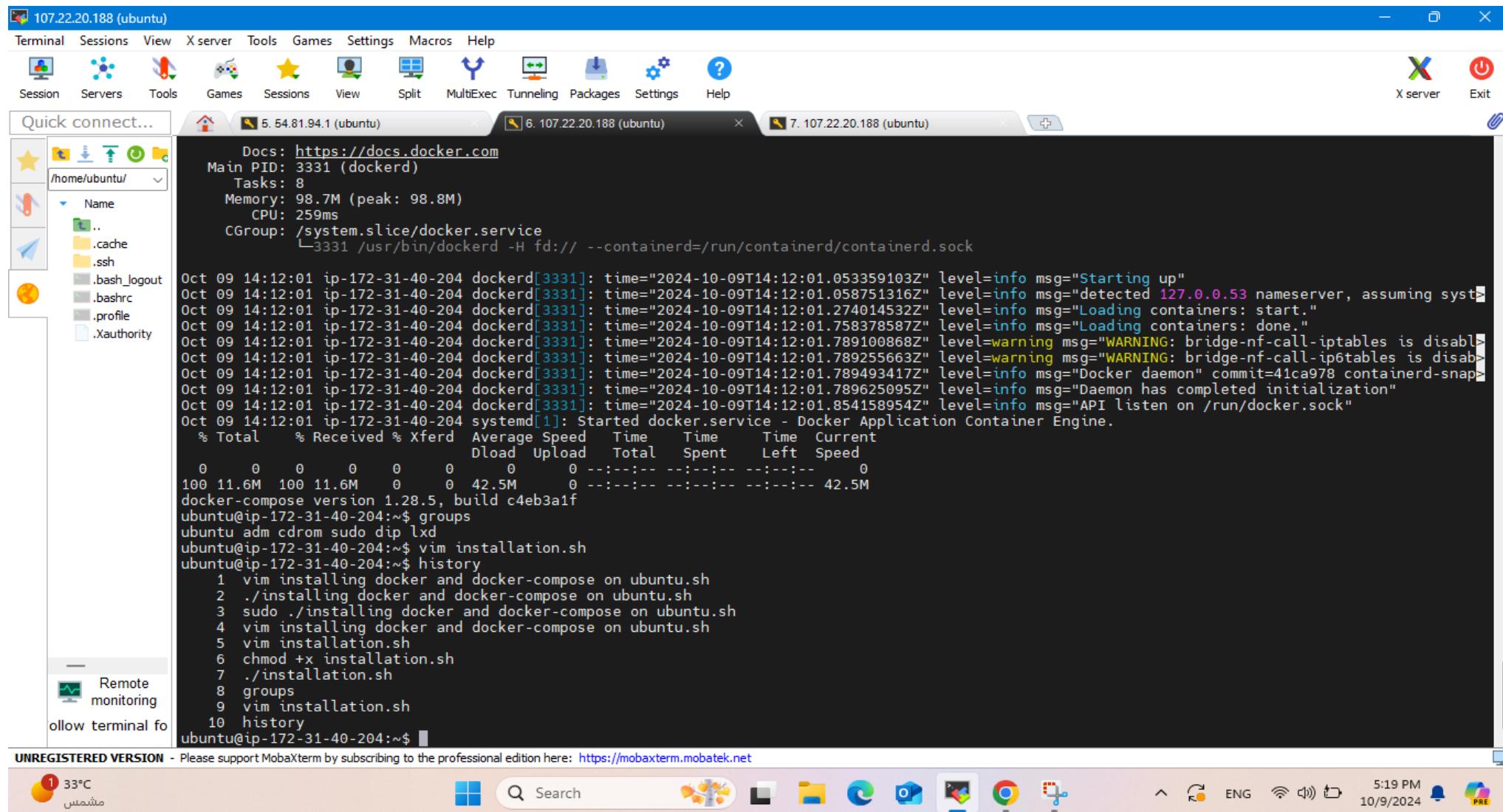


The screenshot shows a desktop interface with multiple windows. The main window is a terminal titled "107.22.20.188 (ubuntu)". The terminal contains the following bash script:

```
#!/bin/bash

sudo apt update
sudo apt install apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
sudo apt update
sudo apt install docker-ce
sudo systemctl status docker
sudo usermod -aG docker ${USER}
sudo curl -L "https://github.com/docker/compose/releases/download/1.28.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
docker-compose --version
```

22- Successfully installing docker and docker-compose using script on destination server



The screenshot shows a MobaXterm window with three tabs open:

- Tab 5: 54.81.94.1 (ubuntu)
- Tab 6: 6. 107.22.20.188 (ubuntu)
- Tab 7: 7. 107.22.20.188 (ubuntu)

The content of Tab 7 shows the output of a Docker installation script:

```
Docs: https://docs.docker.com
Main PID: 3331 (dockerd)
Tasks: 8
Memory: 98.7M (peak: 98.8M)
CPU: 259ms
CGroup: /system.slice/docker.service
└─3331 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.053359103Z" level=info msg="Starting up"
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.058751316Z" level=info msg="detected 127.0.0.53 nameserver, assuming system-wide"
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.274014532Z" level=info msg="Loading containers: start."
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.758378587Z" level=info msg="Loading containers: done."
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.789100868Z" level=warning msg="WARNING: bridge-nf-call-iptables is disabled"
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.789255663Z" level=warning msg="WARNING: bridge-nf-call-ip6tables is disabled"
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.789493417Z" level=info msg="Docker daemon" commit=41ca978 containerd-snapshots=0
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.789625095Z" level=info msg="Daemon has completed initialization"
Oct 09 14:12:01 ip-172-31-40-204 dockerd[3331]: time="2024-10-09T14:12:01.854158954Z" level=info msg="API listen on /run/docker.sock"
Oct 09 14:12:01 ip-172-31-40-204 systemd[1]: Started docker.service - Docker Application Container Engine.

% Total    % Received % Xferd  Average Speed   Time     Time  Current
          Dload  Upload   Total Spent    Left  Speed
0       0      0      0      0      0      0      0      0      0      0
100 11.6M 100 11.6M 0      0  42.5M 0      0      0      0      42.5M
docker-compose version 1.28.5, build c4eb3a1f
ubuntu@ip-172-31-40-204:~$ groups
ubuntu adm cdrom sudo dip lxd
ubuntu@ip-172-31-40-204:~$ vim installation.sh
ubuntu@ip-172-31-40-204:~$ history
 1 vim installing docker and docker-compose on ubuntu.sh
 2 ./installing docker and docker-compose on ubuntu.sh
 3 sudo ./installing docker and docker-compose on ubuntu.sh
 4 vim installing docker and docker-compose on ubuntu.sh
 5 vim installation.sh
 6 chmod +x installation.sh
 7 ./installation.sh
 8 groups
 9 vim installation.sh
10 history
ubuntu@ip-172-31-40-204:~$
```

At the bottom of the terminal window, there is a message: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

23- Writing bash script to transfer application code to target server

The screenshot shows a MobaXterm window titled "54.81.94.1 (ubuntu)". The window contains a terminal session with the following bash script:

```
#!/bin/bash

# Variables
EC2_PUBLIC_IP="107.22.20.188" # Destination server's public IP
EC2_USER= "ubuntu" # user id
PEM_FILE="/home/ubuntu/.ssh/id_rsa" # Path to your AWS .pem key
LOCAL_PROJECT_DIR="/home/ubuntu/Node-API-Mongo-App" # Path to local project directory
REMOTE_PROJECT_DIR="/home/ubuntu/Node-API-Mongo-Copy" # Directory on the destination server where the project will be transferred

# Transfer code to Destination server
echo "Transferring code to EC2 instance..."
scp -i $PEM_FILE -r $LOCAL_PROJECT_DIR $EC2_USER@$EC2_PUBLIC_IP:$REMOTE_PROJECT_DIR
```

The terminal window has two tabs: "5. 54.81.94.1 (ubuntu)" and "8. 107.22.20.188 (ubuntu)". The status bar at the bottom right shows "23.0-1 All". The bottom taskbar includes icons for weather (24°C), search, file explorer, and various system applications.

24- Writing bash script to transfer application code to target server

The screenshot shows a MobaXterm window with two tabs: "5. 54.81.94.1 (ubuntu)" and "8. 107.22.20.188 (ubuntu)". The left tab displays a command-line session where a user is navigating through a directory structure and executing a script named "Copy_Automate_Deployment.sh". The right tab shows a file transfer progress bar for multiple files, indicating speeds up to 953.5KB/s.

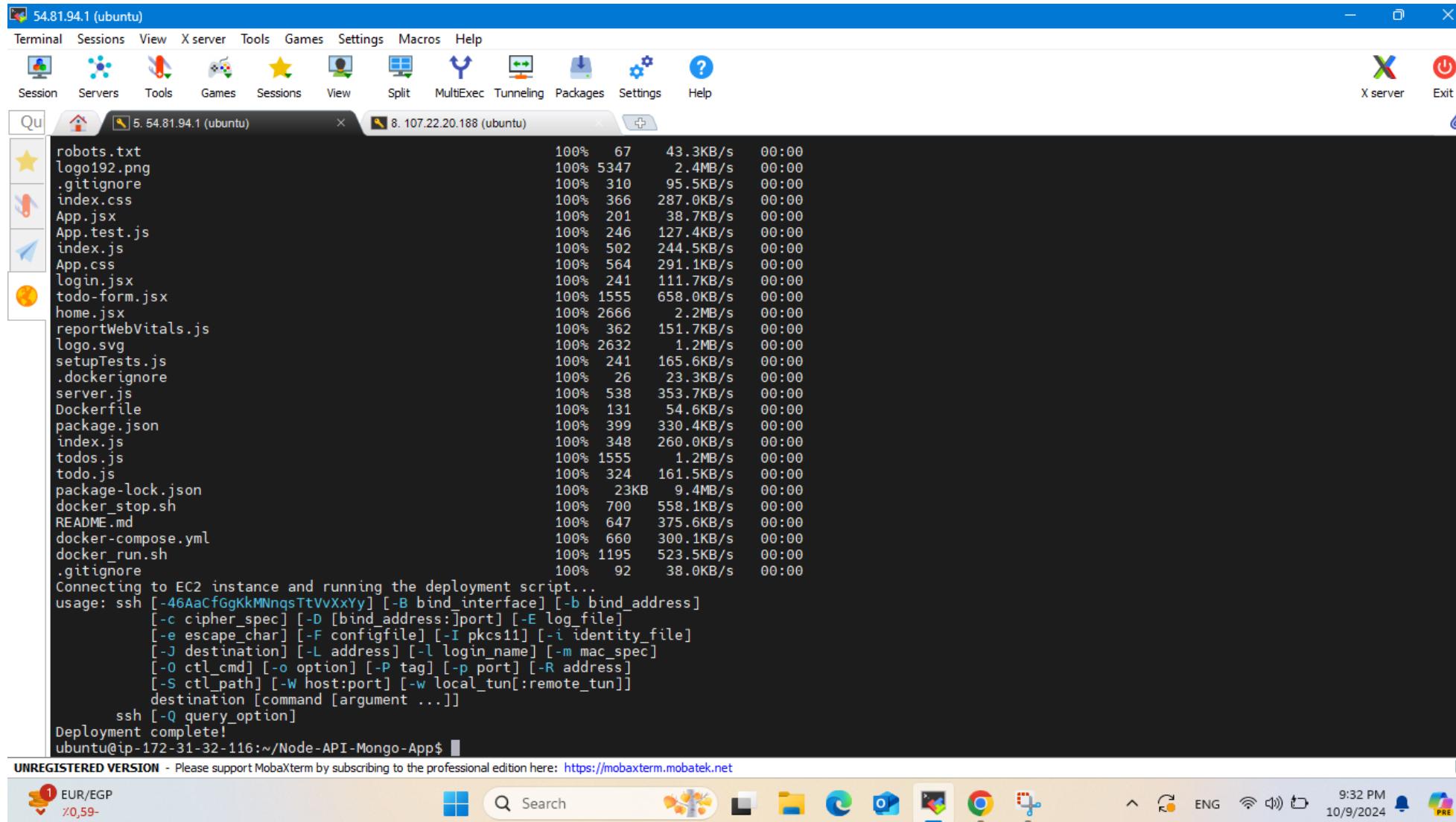
```
ubuntu@ip-172-31-32-116:~$ chmod 600 ~/.ssh/id_rsa
ubuntu@ip-172-31-32-116:~$ cd Node-API-Mongo-App
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim Copy_Automate_Deploy
ment.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ./Copy_Automate_Deployment.sh
./Copy_Automate_Deployment.sh: line 5: ubuntu: command not found
Transferring code to EC2 instance...
Copy_Automate_Deployment.sh
docker_stop.log
docker_run.log
main
HEAD
main
d2e77b695daa401054016b68ed07406b5f989a
62cb1ce8b8a95306280799350aa722c016f9ed
ee2f65be83ba3c38ed81a180ce18d380b11a11
9e6b074e029aba03512e16d1b56fc8dd47c086
pack-3895397ddf14d6f779f6a034b372dd0feb4eed.rev
pack-3895397ddf14d6f779f6a034b372dd0feb4eed.pack
pack-3895397ddf14d6f779f6a034b372dd0feb4eed.idx
3538ec4961e76a52955fa7abb5b09ec74aa00e
fc4b33996ceed41ddf00e727161a56d0012e13
index
config
exclude
main
HEAD
main
HEAD
packed-refs
description
HEAD
prepare-commit-msg.sample
pre-applypatch.sample
post-update.sample
push-to-checkout.sample
pre-merge-commit.sample
fsmonitor-watchman.sample
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

1 EUR/EGP
X0,59-

Search 9:30 PM 10/9/2024 ENG PRE

25- Application code is successfully copied to target server

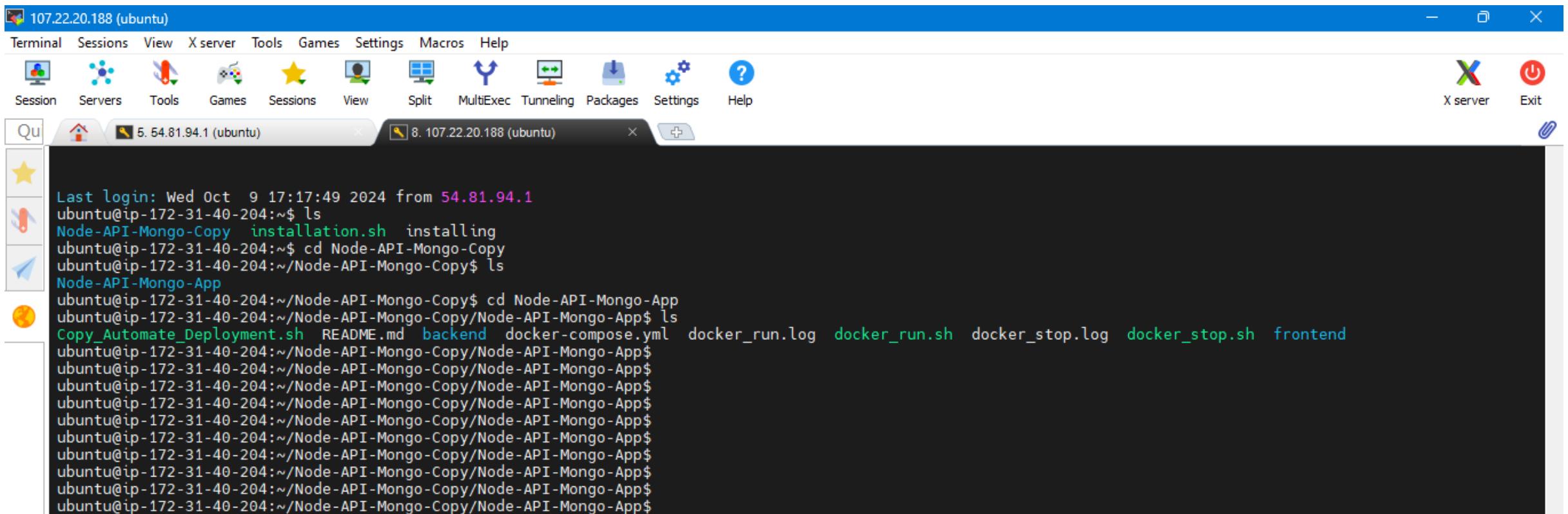


The screenshot shows a MobaXterm window titled "54.81.94.1 (ubuntu)" with two tabs: "5. 54.81.94.1 (ubuntu)" and "8. 107.22.20.188 (ubuntu)". The left pane displays a file tree with files like robots.txt, logo192.png, .gitignore, index.css, App.jsx, App.test.js, index.js, App.css, login.jsx, todo-form.jsx, home.jsx, reportWebVitals.js, logo.svg, setupTests.js, .dockerignore, server.js, Dockerfile, package.json, index.js, todos.js, todo.js, package-lock.json, docker_stop.sh, README.md, docker-compose.yml, docker_run.sh, and .gitignore. The right pane shows a terminal session on the second tab with the following output:

```
robots.txt          100%   67    43.3KB/s  00:00
logo192.png        100% 5347    2.4MB/s  00:00
.gitignore          100%   310   95.5KB/s  00:00
index.css           100%   366   287.0KB/s  00:00
App.jsx             100%   201   38.7KB/s  00:00
App.test.js         100%   246   127.4KB/s  00:00
index.js            100%   502   244.5KB/s  00:00
App.css             100%   564   291.1KB/s  00:00
login.jsx           100%   241   111.7KB/s  00:00
todo-form.jsx       100% 1555   658.0KB/s  00:00
home.jsx            100% 2666   2.2MB/s  00:00
reportWebVitals.js 100% 362    151.7KB/s  00:00
logo.svg            100% 2632   1.2MB/s  00:00
setupTests.js       100%   241   165.6KB/s  00:00
.dockerignore        100%   26    23.3KB/s  00:00
server.js           100%   538   353.7KB/s  00:00
Dockerfile          100%   131   54.6KB/s  00:00
package.json         100%   399   330.4KB/s  00:00
index.js             100%   348   260.0KB/s  00:00
todos.js             100% 1555   1.2MB/s  00:00
todo.js              100%   324   161.5KB/s  00:00
package-lock.json   100%   23KB   9.4MB/s  00:00
docker_stop.sh       100%   700   558.1KB/s  00:00
README.md            100%   647   375.6KB/s  00:00
docker-compose.yml   100%   660   300.1KB/s  00:00
docker_run.sh         100% 1195   523.5KB/s  00:00
.gitignore            100%   92    38.0KB/s  00:00
Connecting to EC2 instance and running the deployment script...
usage: ssh [-46AaCfGgKkMNqsTtVvXxYy] [-B bind_interface] [-b bind_address]
           [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
           [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
           [-J destination] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-P tag] [-p port] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           destination [command [argument ...]]
ssh [-Q query_option]
Deployment complete!
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

At the bottom, a status bar indicates "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>". The system tray shows icons for battery (EUR/EGP X0,59-), search, file, clipboard, and network.

26- Application files in destination server



The screenshot shows a Linux desktop environment with a blue header bar. The title bar reads "107.22.20.188 (ubuntu)". The menu bar includes "Terminal", "Sessions", "View", "X server", "Tools", "Games", "Settings", "Macros", and "Help". Below the menu is a toolbar with icons for Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help. On the right side of the header are "X server" and "Exit" buttons. The main window contains a terminal session titled "Qu". The terminal output shows the following command history:

```
Last login: Wed Oct  9 17:17:49 2024 from 54.81.94.1
ubuntu@ip-172-31-40-204:~$ ls
Node-API-Mongo-Copy  installation.sh  installing
ubuntu@ip-172-31-40-204:~$ cd Node-API-Mongo-Copy
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy$ ls
Node-API-Mongo-App
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy$ cd Node-API-Mongo-App
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$ ls
Copy_Automate_Deployment.sh  README.md  backend  docker-compose.yml  docker_run.log  docker_run.sh  docker_stop.log  docker_stop.sh  frontend
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$
```

27- Writing bash script to ssh from main server to destination server and run the application using(docker-compose up – build –d)

The screenshot shows a terminal window titled "54.81.94.1 (ubuntu)" in MobaXterm. The window contains a bash script for deploying an application. The script uses SSH to connect to a destination server (IP 8.107.22.20.188) and runs a deployment script. It then checks if the application is running successfully. If it is, it prints a success message; if not, it prints a failure message and exits with code 1.

```
#!/bin/bash

# Step 1: SSH into the destination server and build containers to run the application
echo "Connecting to EC2 instance and running the deployment script..."
ssh -i /home/ubuntu/.ssh/id_rsa ubuntu@107.22.20.188 << EOF
    cd Node-API-Mongo-Copy/Node-API-Mongo-App
    docker-compose up -d --build
EOF

# Step 2: Checking if the application is successfully running
if [ $? -eq 0 ]; then
    echo "Application is running and Deployed Successfully ."
else
    echo "Failed to the application ." >&2
    exit 1
fi
```

At the bottom of the terminal window, there is a footer message: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>". The status bar at the bottom right shows the date and time: "29, 0-1 All 1:48 AM 10/10/2024".

28- Deploying application in destination server by executing the bash script in main server

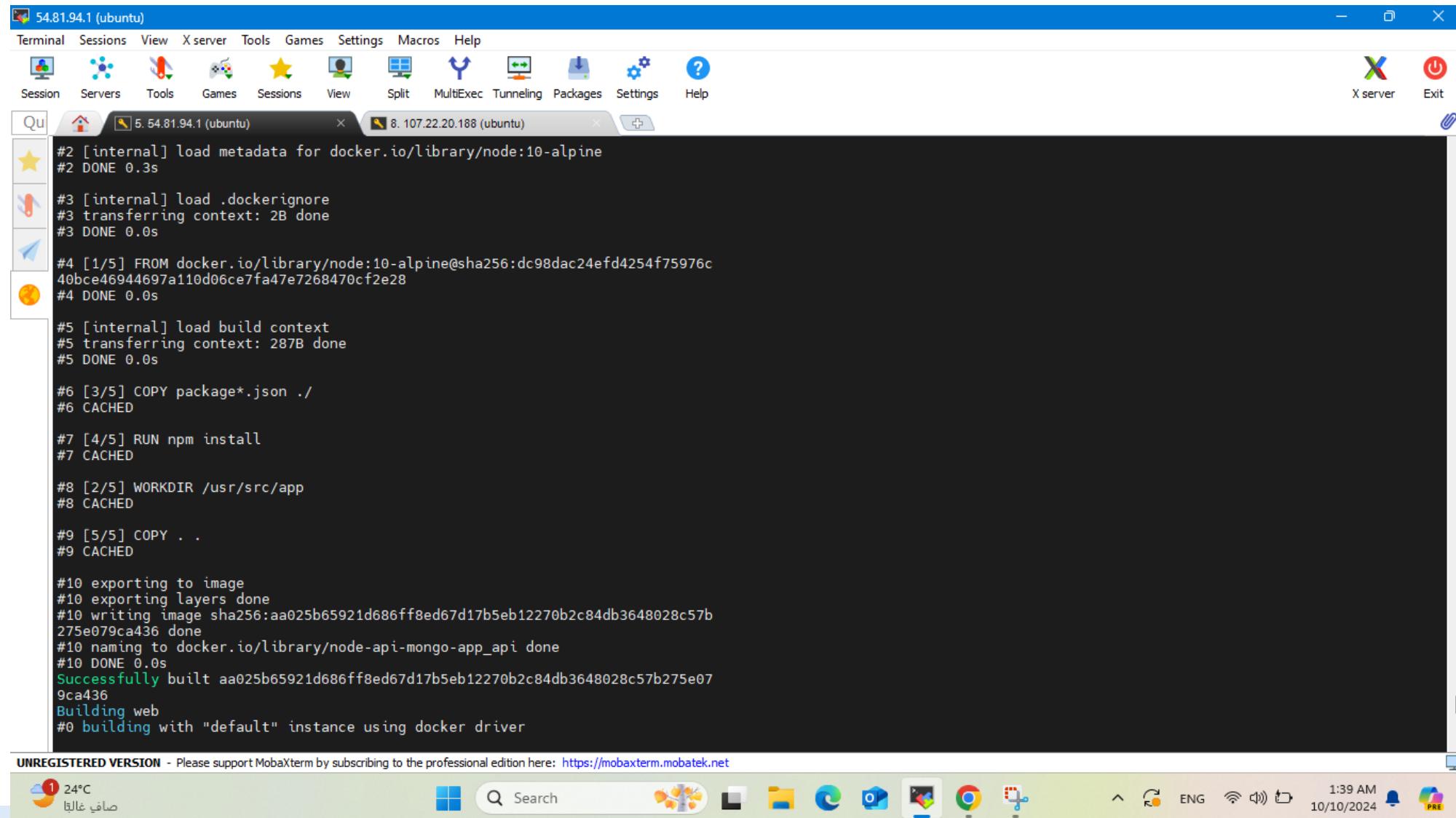
The screenshot shows a terminal window in MobaXterm with two tabs open:

- 54.81.94.1 (ubuntu)**: This tab displays the deployment command being run on the destination server. The output shows the user is connecting to an EC2 instance and running a deployment script. It also provides system information for Ubuntu 24.04.1 LTS.
- 8.107.22.20.188 (ubuntu)**: This tab is currently active and shows the deployment process. The user runs `./SSH_Build_Deploy. ssh`, which connects to the EC2 instance and runs the deployment script. The terminal then displays system information for the destination server, including load average, memory usage, and network details. It also checks for security updates and enables ESM Apps. Finally, it creates a Docker network and builds the application using Dockerfile.

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

1 24°C صافي غالباً 1:38 AM ENG 10/10/2024 PRE

28- Deploying application in destination server by executing the bash script in main server



The screenshot shows a MobaXterm window titled "54.81.94.1 (ubuntu)". The terminal session displays the following Docker build logs:

```
#2 [internal] load metadata for docker.io/library/node:10-alpine
#2 DONE 0.3s

#3 [internal] load .dockerrcignore
#3 transferring context: 2B done
#3 DONE 0.0s

#4 [1/5] FROM docker.io/library/node:10-alpine@sha256:dc98dac24efd4254f75976c
40bce46944697a110d06ce7fa47e7268470cf2e28
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 287B done
#5 DONE 0.0s

#6 [3/5] COPY package*.json .
#6 CACHED

#7 [4/5] RUN npm install
#7 CACHED

#8 [2/5] WORKDIR /usr/src/app
#8 CACHED

#9 [5/5] COPY ..
#9 CACHED

#10 exporting to image
#10 exporting layers done
#10 writing image sha256:aa025b65921d686ff8ed67d17b5eb12270b2c84db3648028c57b
275e079ca436 done
#10 naming to docker.io/library/node-api-mongo-app_api done
#10 DONE 0.0s
Successfully built aa025b65921d686ff8ed67d17b5eb12270b2c84db3648028c57b275e07
9ca436
Building web
#0 building with "default" instance using docker driver
```

At the bottom of the terminal window, a message reads: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

28- Deploying application in destination server by executing the bash script in main server

The screenshot shows a MobaXterm window titled "5. 54.81.94.1 (ubuntu)". The terminal output displays the following Docker build logs:

```
Successfully built aa025b65921d686ff8ed67d17b5eb12270b2c84db3648028c57b275e07
9ca436
Building web
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 320B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/node:14-alpine
#2 DONE 0.1s

#3 [internal] load .dockerignore
#3 transferring context: 123B done
#3 DONE 0.0s

#4 [1/5] FROM docker.io/library/node:14-alpine@sha256:434215b487a329c9e867202
ff89e704d3a75e554822e07f3e0c0f9e606121b33
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 914B done
#5 DONE 0.0s

#6 [2/5] WORKDIR /app
#6 CACHED

#7 [3/5] COPY package*.json .
#7 CACHED

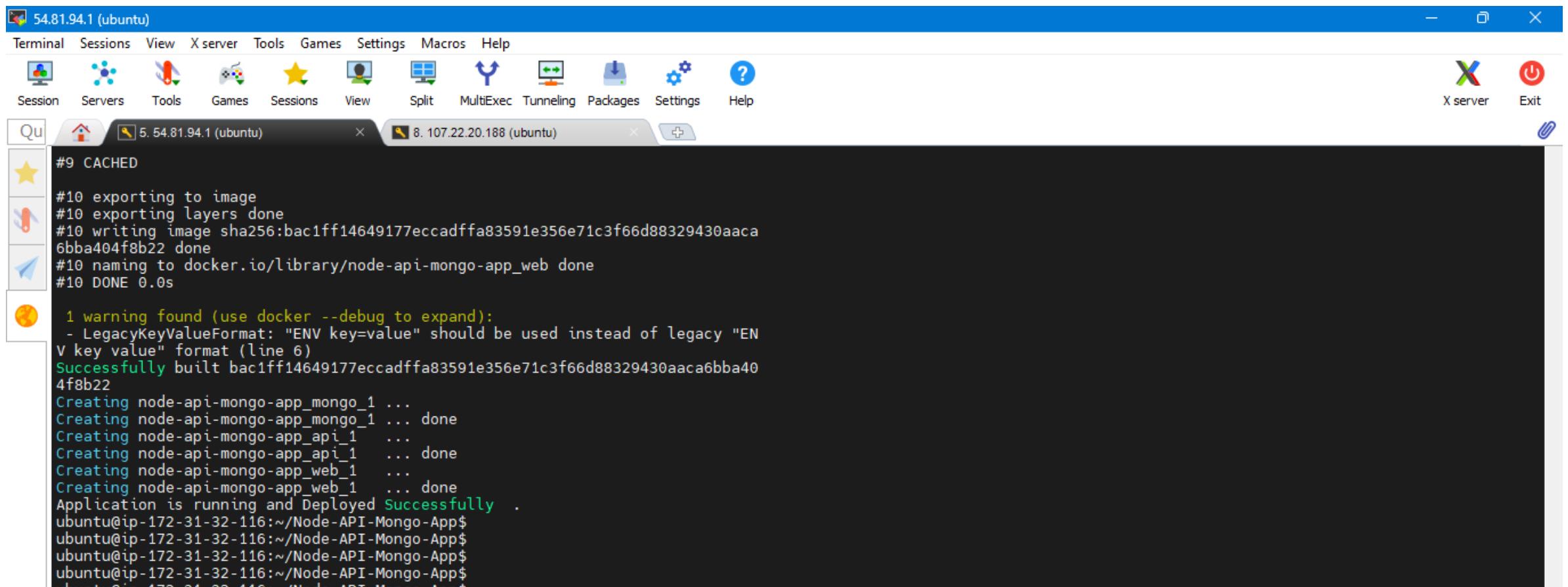
#8 [4/5] RUN npm install
#8 CACHED

#9 [5/5] COPY . .
#9 CACHED

#10 exporting to image
#10 exporting layers done
```

At the bottom of the terminal window, a message reads: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

29- Application successfully deployed in main server and the three containers are running by executing run bash script from main server



The screenshot shows a desktop environment with a terminal window open. The terminal window title is "54.81.94.1 (ubuntu)". The terminal content displays the output of a Docker build command:

```
#9 CACHED
#10 exporting to image
#10 exporting layers done
#10 writing image sha256:bac1ff14649177eccadffa83591e356e71c3f66d88329430aaca
6bba404f8b22 done
#10 naming to docker.io/library/node-api-mongo-app_web done
#10 DONE 0.0s

1 warning found (use docker --debug to expand):
- LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "EN
V key value" format (line 6)
Successfully built bac1ff14649177eccadffa83591e356e71c3f66d88329430aaca6bba40
4f8b22
Creating node-api-mongo-app_mongo_1 ...
Creating node-api-mongo-app_mongo_1 ... done
Creating node-api-mongo-app_api_1 ...
Creating node-api-mongo-app_api_1 ... done
Creating node-api-mongo-app_web_1 ...
Creating node-api-mongo-app_web_1 ... done
Application is running and Deployed Successfully .
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

30- Successfully serving the application from destination server

The screenshot shows a browser window titled "React App" displaying a "Todos" application. The URL in the address bar is "Not secure 107.22.20.188:3000". The page lists two items:

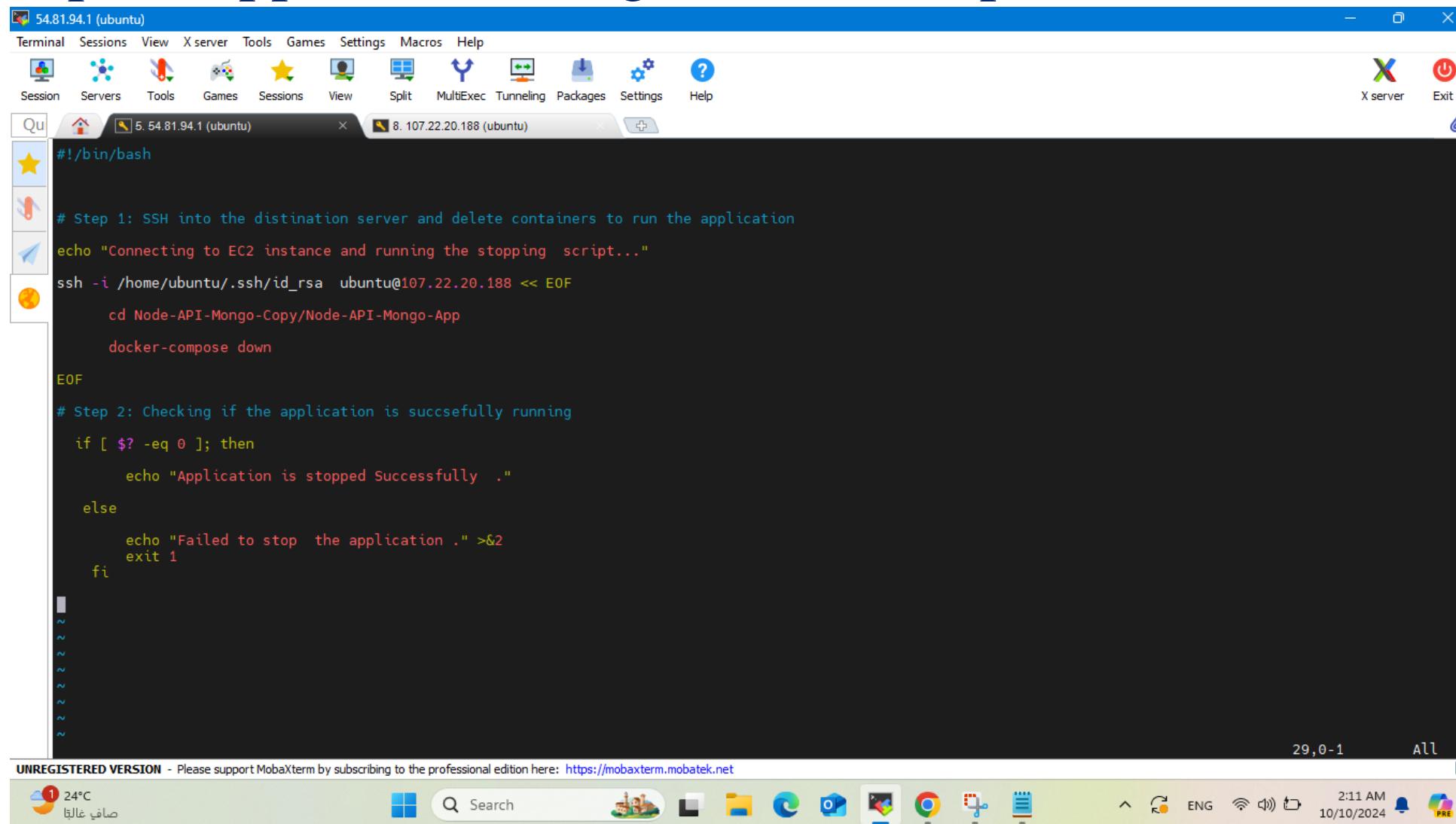
- Target Server on AWS** Due: 2024-10-10T00:00:00.000Z
This is the application served from target server after copying the files from the main server using scp
- Steps for Deployment** Due: 2024-10-09T21:09:24.093Z
1. Run the first script to copy application files to target server. 2. Run the second script to: 1) ssh into destination server 2) run docker-compose up to run application 3. Run the third script to send an HTTP request to the EC2 public IP to ensure the application is running and returning a 200 OK status. 4. Run the fourth script to stop all containers

A blue "Add Todo" button is located at the bottom left of the list.

At the bottom of the screen, a Windows taskbar is visible with various icons and information:

- Weather: 25°C صافي غالباً
- Search bar
- Icons for File Explorer, Edge, OneDrive, Paint, and Google Chrome
- Language: ENG
- Network: Wi-Fi signal
- Date and Time: 12:12 AM 10/10/2024
- Volume and Battery icons
- Pins icon

31- Writing bash script to ssh from main server to destination server and stop the application using(docker-compose down)



The screenshot shows a MobaXterm window titled "54.81.94.1 (ubuntu)". The terminal session contains the following bash script:

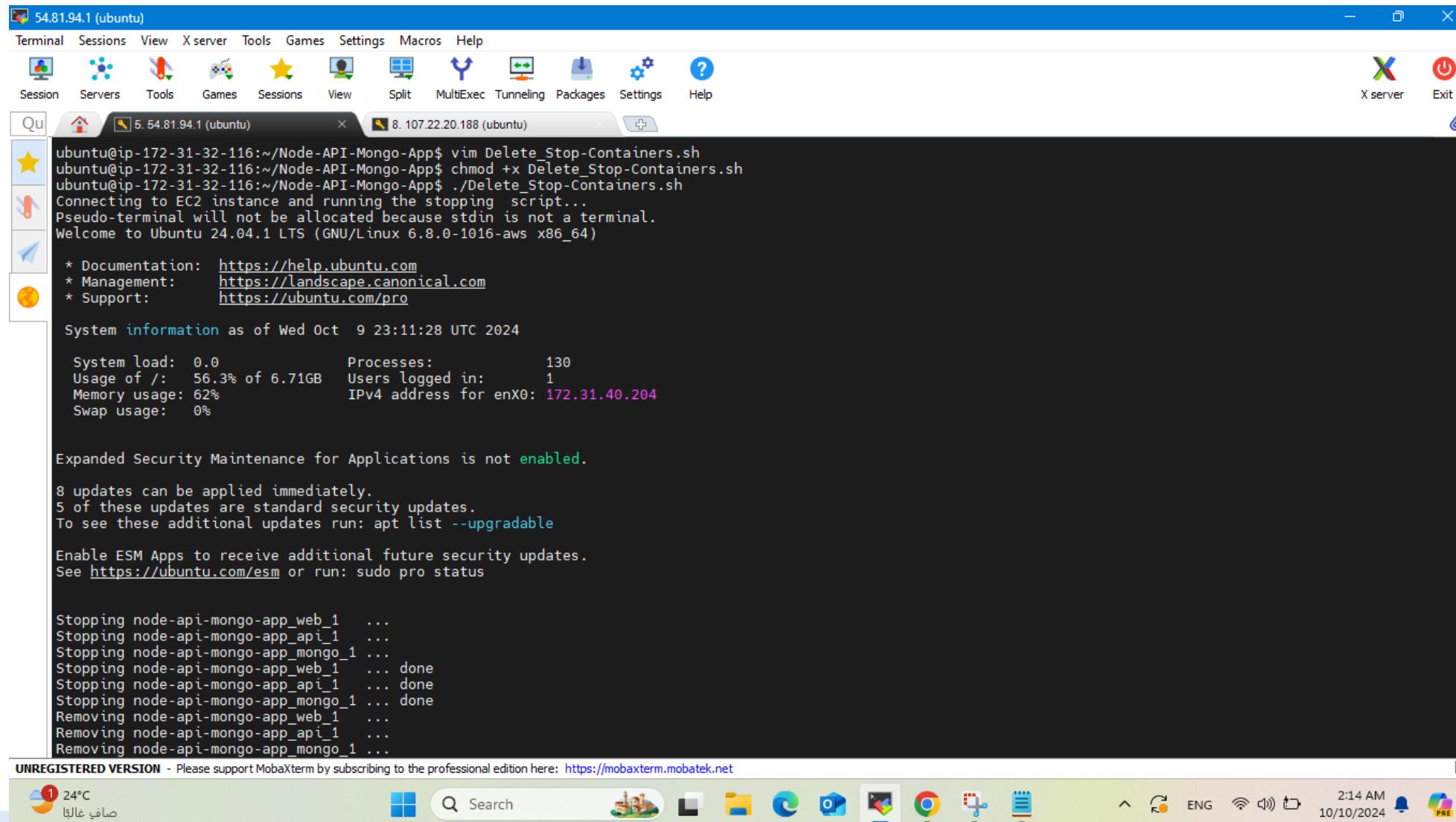
```
#!/bin/bash

# Step 1: SSH into the destination server and delete containers to run the application
echo "Connecting to EC2 instance and running the stopping script..."
ssh -i /home/ubuntu/.ssh/id_rsa ubuntu@107.22.20.188 << EOF
cd Node-API-Mongo-Copy/Node-API-Mongo-App
docker-compose down
EOF

# Step 2: Checking if the application is successfully running
if [ $? -eq 0 ]; then
    echo "Application is stopped Successfully ."
else
    echo "Failed to stop the application ." >&2
    exit 1
fi
```

The terminal window has two tabs: "54.81.94.1 (ubuntu)" and "8.107.22.20.188 (ubuntu)". The status bar at the bottom right shows "29,0-1 All". The taskbar at the bottom includes icons for File Explorer, Edge, and other system tools.

32- Stopping the application in destination server by executing the bash script in main server



The screenshot shows a MobaXterm window titled "54.81.94.1 (ubuntu)". The terminal session is connected to port 54.81.94.1. The user is executing a bash script named "Delete_Stop-Containers.sh". The script connects to an EC2 instance and runs the stopping script. It shows system information and security maintenance status. Finally, it stops multiple containers: node-api-mongo-app_web_1, node-api-mongo-app_api_1, and node-api-mongo-app_mongo_1.

```
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim Delete_Stop-Containers.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ chmod +x Delete_Stop-Containers.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ./Delete_Stop-Containers.sh
Connecting to EC2 instance and running the stopping script...
Pseudo-terminal will not be allocated because stdin is not a terminal.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Wed Oct 9 23:11:28 UTC 2024

System load: 0.0 Processes: 130
Usage of /: 56.3% of 6.71GB Users logged in: 1
Memory usage: 62% IPv4 address for enX0: 172.31.40.204
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

8 updates can be applied immediately.
5 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

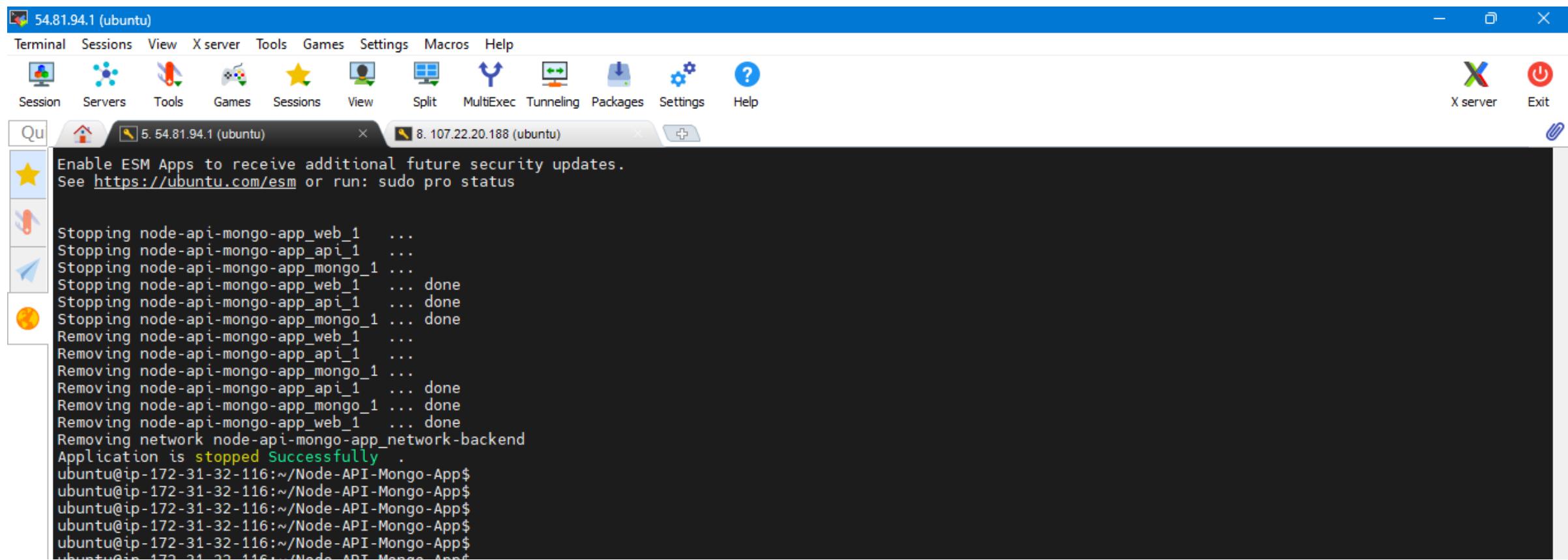
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Stopping node-api-mongo-app_web_1 ...
Stopping node-api-mongo-app_api_1 ...
Stopping node-api-mongo-app_mongo_1 ...
Stopping node-api-mongo-app_web_1 ... done
Stopping node-api-mongo-app_api_1 ... done
Stopping node-api-mongo-app_mongo_1 ... done
Removing node-api-mongo-app_web_1 ...
Removing node-api-mongo-app_api_1 ...
Removing node-api-mongo-app_mongo_1 ...

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

At the bottom, the Windows taskbar shows the date (10/10/2024), time (2:14 AM), battery level (24°C), and system icons.

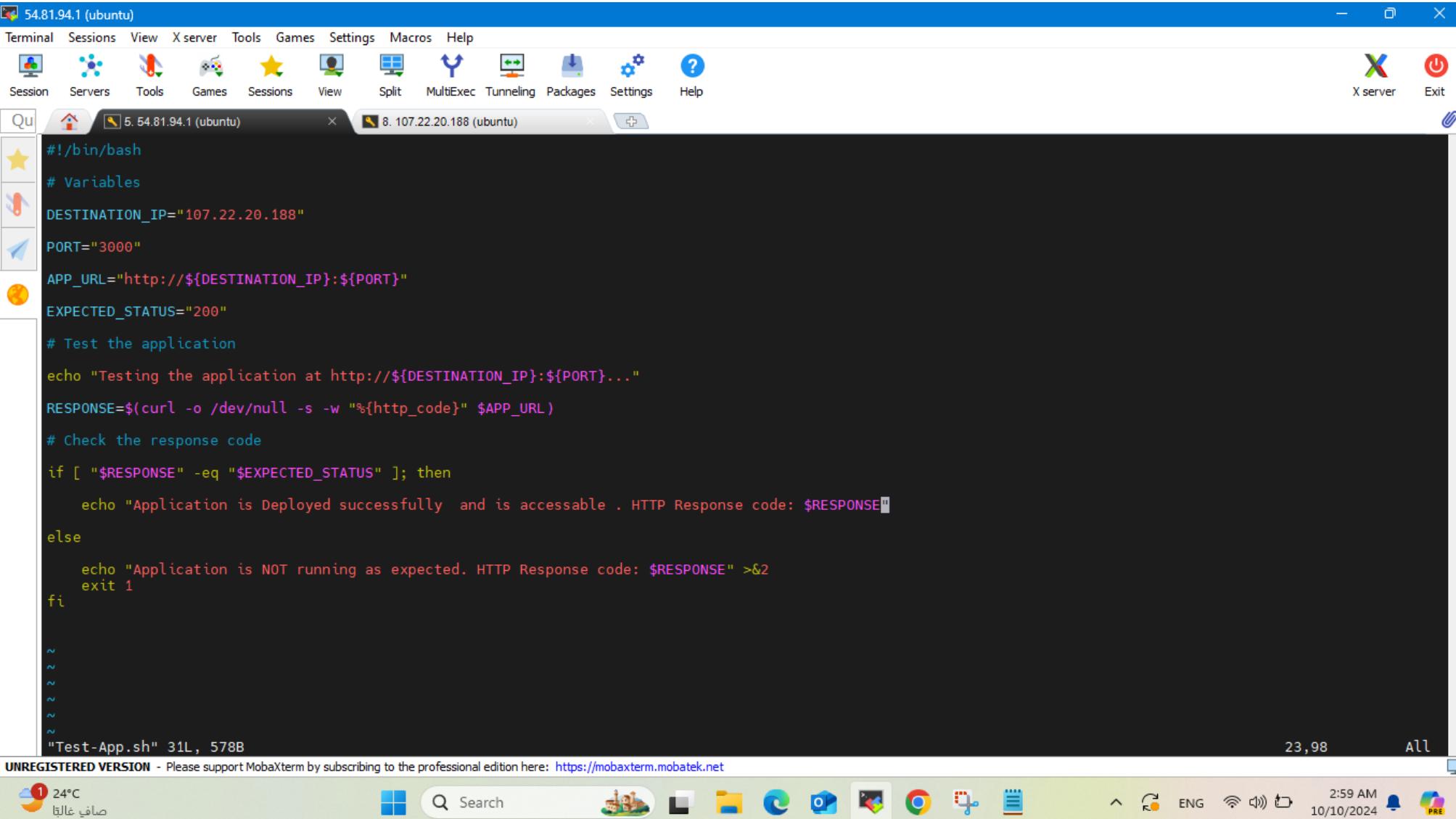
33- Application successfully stopped in destination server and the three containers are removed by executing stop bash script from main server



The screenshot shows a desktop interface with a terminal window open. The terminal window title is "5. 54.81.94.1 (ubuntu)". The terminal content displays the output of a command to stop and remove Docker containers:

```
Stopping node-api-mongo-app_web_1 ...
Stopping node-api-mongo-app_api_1 ...
Stopping node-api-mongo-app_mongo_1 ...
Stopping node-api-mongo-app_web_1 ... done
Stopping node-api-mongo-app_api_1 ... done
Stopping node-api-mongo-app_mongo_1 ... done
Removing node-api-mongo-app_web_1 ...
Removing node-api-mongo-app_api_1 ...
Removing node-api-mongo-app_mongo_1 ...
Removing node-api-mongo-app_web_1 ... done
Removing network node-api-mongo-app_network-backend
Application is stopped successfully .
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

34- Writing bash script to test if the application has successfully deployed using (curl)



The screenshot shows a terminal window titled "54.81.94.1 (ubuntu)" in the MobaXterm interface. The window contains a bash script named "Test-App.sh". The script uses curl to check if an application is running at a specific IP and port, and then prints a message indicating whether it was successful or not.

```
#!/bin/bash
# Variables
DESTINATION_IP="107.22.20.188"
PORT="3000"
APP_URL="http://${DESTINATION_IP}:${PORT}"
EXPECTED_STATUS="200"

# Test the application
echo "Testing the application at http://${DESTINATION_IP}:${PORT}..."
RESPONSE=$(curl -o /dev/null -s -w "%{http_code}" $APP_URL)

# Check the response code
if [ "$RESPONSE" -eq "$EXPECTED_STATUS" ]; then
    echo "Application is Deployed successfully and is accessible . HTTP Response code: $RESPONSE"
else
    echo "Application is NOT running as expected. HTTP Response code: $RESPONSE" >&2
    exit 1
fi

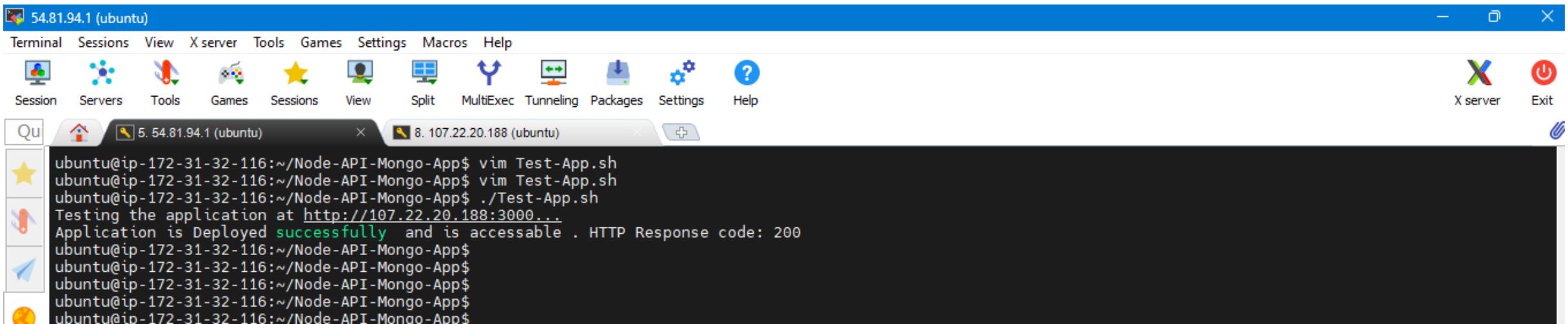
~
```

"Test-App.sh" 31L, 578B

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

24°C 2:59 AM 10/10/2024 ENG PRE

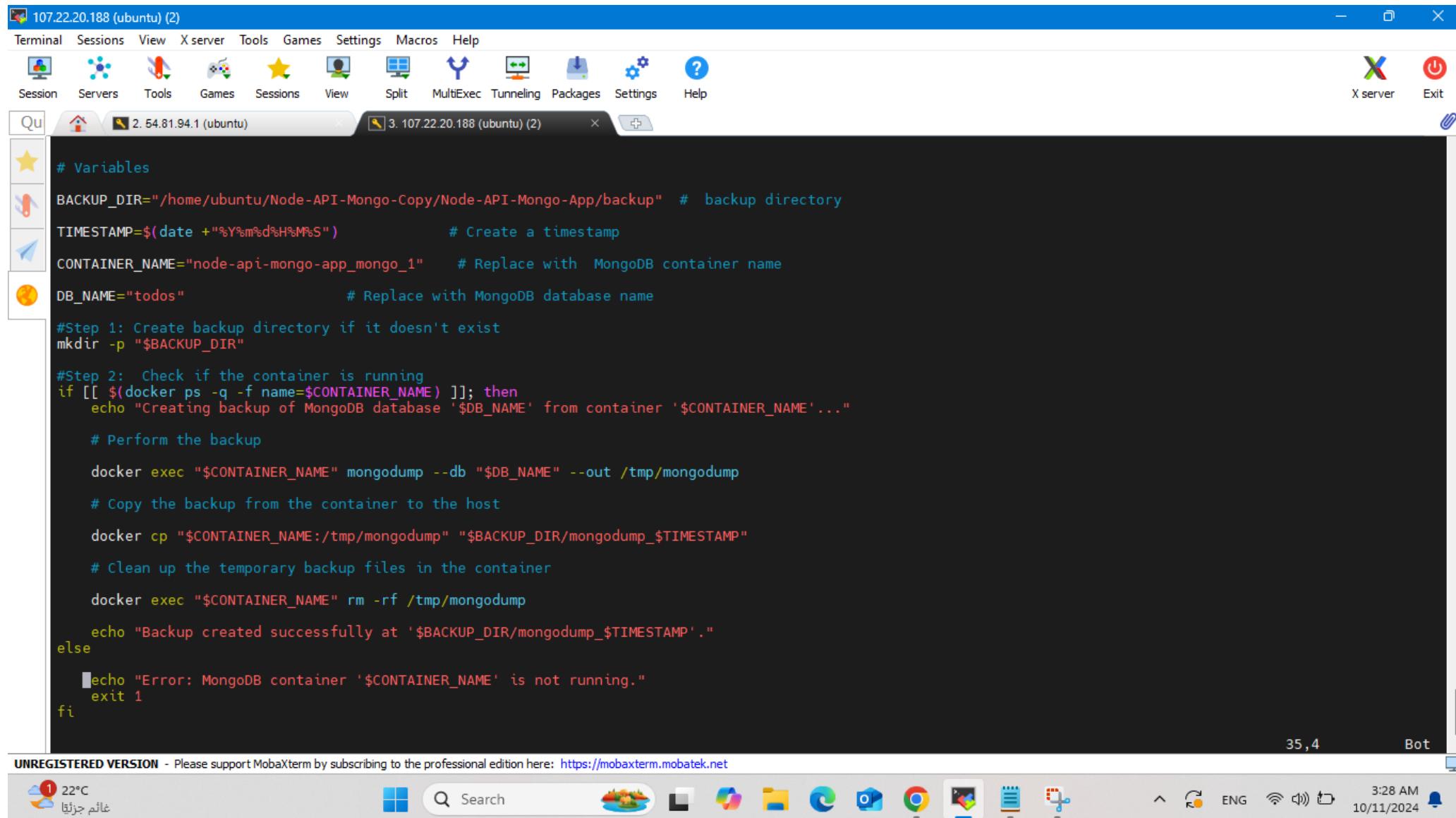
35- Successfully testing the application and get the expected response



The screenshot shows a terminal window titled "54.81.94.1 (ubuntu)" with a blue header bar. The window contains a command-line interface where a user is testing a deployed application. The user runs "vim Test-App.sh" twice, then executes the script with "./Test-App.sh". The output indicates that the application is successfully deployed and accessible at <http://107.22.20.188:3000>. The terminal window has a dark background and light-colored text. The title bar also lists another session: "8. 107.22.20.188 (ubuntu)".

```
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim Test-App.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ vim Test-App.sh
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ ./Test-App.sh
Testing the application at http://107.22.20.188:3000...
Application is Deployed successfully and is accessible . HTTP Response code: 200
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

36- Writing bash script to backup all application data



The screenshot shows a MobaXterm window titled "107.22.20.188 (ubuntu) (2)". The terminal window contains a bash script for backing up a MongoDB database from a Docker container. The script defines variables for the backup directory, timestamp, container name, and database name. It checks if the container is running, performs a mongodump, copies the backup to the host, and then removes temporary files from the container. If the container is not running, it exits with an error code.

```
# Variables
BACKUP_DIR="/home/ubuntu/Node-API-Mongo-Copy/Node-API-Mongo-App/backup" # backup directory
TIMESTAMP=$(date +"%Y%m%d%H%M%S") # Create a timestamp
CONTAINER_NAME="node-api-mongo-app_mongo_1" # Replace with MongoDB container name
DB_NAME="todos" # Replace with MongoDB database name

#Step 1: Create backup directory if it doesn't exist
mkdir -p "$BACKUP_DIR"

#Step 2: Check if the container is running
if [[ $(docker ps -q -f name=$CONTAINER_NAME) ]]; then
    echo "Creating backup of MongoDB database '$DB_NAME' from container '$CONTAINER_NAME'..."
    # Perform the backup
    docker exec "$CONTAINER_NAME" mongodump --db "$DB_NAME" --out /tmp/mongodump
    # Copy the backup from the container to the host
    docker cp "$CONTAINER_NAME:/tmp/mongodump" "$BACKUP_DIR/mongodump_$TIMESTAMP"
    # Clean up the temporary backup files in the container
    docker exec "$CONTAINER_NAME" rm -rf /tmp/mongodump
    echo "Backup created successfully at '$BACKUP_DIR/mongodump_$TIMESTAMP'."
else
    echo "Error: MongoDB container '$CONTAINER_NAME' is not running."
    exit 1
fi
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

22°C غامن جزئیا 3:28 AM 10/11/2024

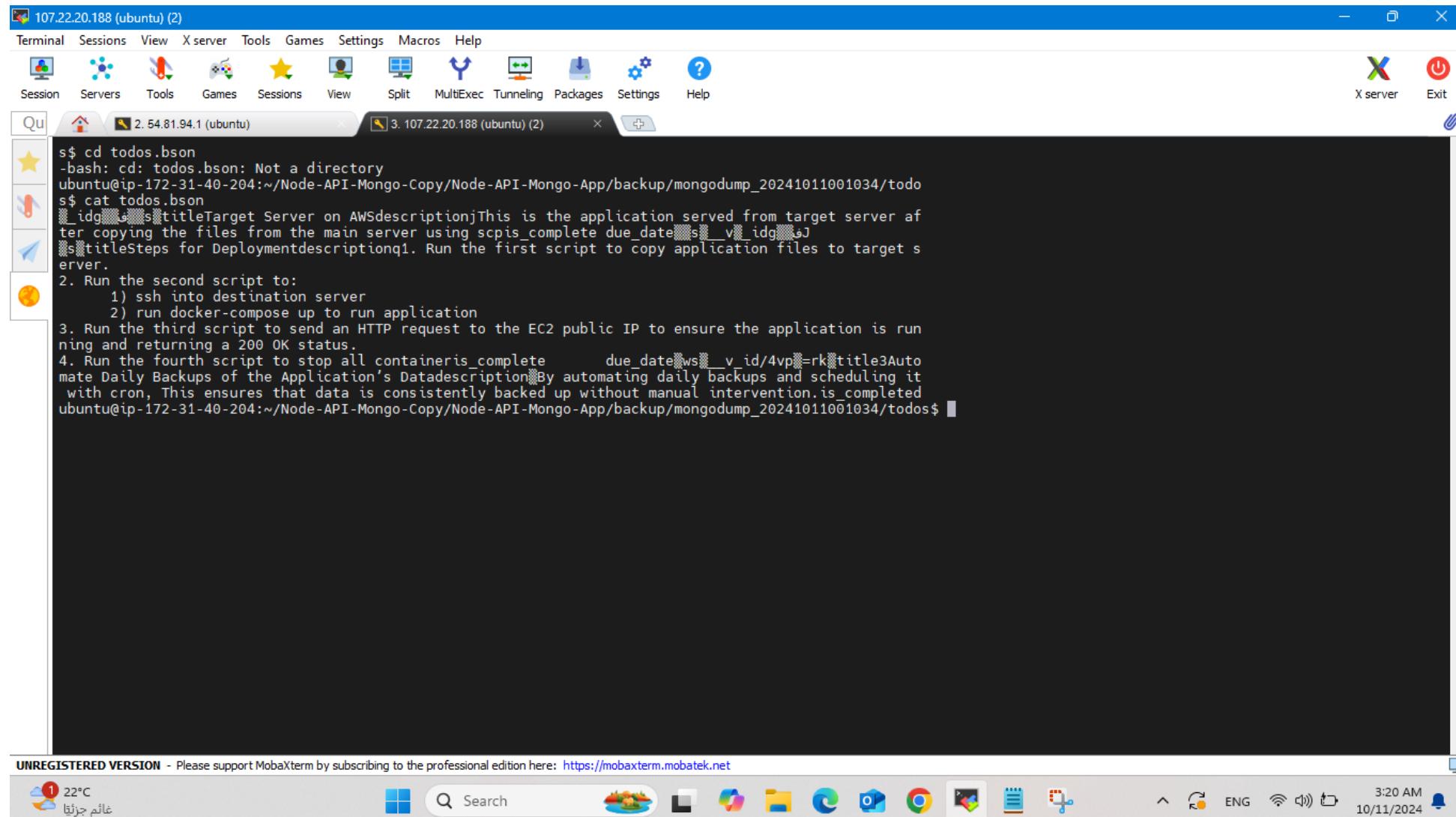
37- Testing and executing the backup script

The screenshot shows a MobaXterm window titled "107.22.20.188 (ubuntu) (2)". The interface includes a top menu bar with "Terminal", "Sessions", "View", "X server", "Tools", "Games", "Settings", "Macros", and "Help". Below the menu is a toolbar with icons for Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help. On the right side, there are "X server" and "Exit" buttons. The main area contains a terminal window with the following text:

```
api_1
e7e951db2708    mongo          "docker-entrypoint.s..."  4 hours ago
Up 4 hours  0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  node-api-mongo-app
_mongo_1
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$ vim mongodb-b
ackup.vim
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$ chmod +x mong
odb-backup.vim
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$ ./mongodb-bac
kup.vim
Creating backup of MongoDB database 'todos' from container 'node-api-mongo-app_
mongo_1'...
2024-10-11T00:10:35.262+0000      writing todos.todos to /tmp/mongodump/todos/tod
os.bson
2024-10-11T00:10:35.266+0000      done dumping todos.todos (3 documents)
Successfully copied 4.61kB to /home/ubuntu/Node-API-Mongo-Copy/Node-API-Mongo-A
pp/backup/mongodump_20241011001034
Backup created successfully at '/home/ubuntu/Node-API-Mongo-Copy/Node-API-Mongo
-App/backup/mongodump_20241011001034'.
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App$ cd backup
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup$ cd mon
godump_20241011001034'.
>
>
> ^C
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup$ ls
mongodump_20241011001034
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup$ cd mongodump_20241011001034
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034$ ls
todos
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034$ cd
todos
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034/todo
s$ ls
todos.bson  todos.metadata.json
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034/todo
s$ cd todos.bson
-bash: cd: todos.bson: Not a directory
```

The bottom status bar shows system information like battery level (22%), language (English), signal strength, and date/time (10/11/2024, 3:19 AM).

38- Content of backup file indicating successfully execution of backup script

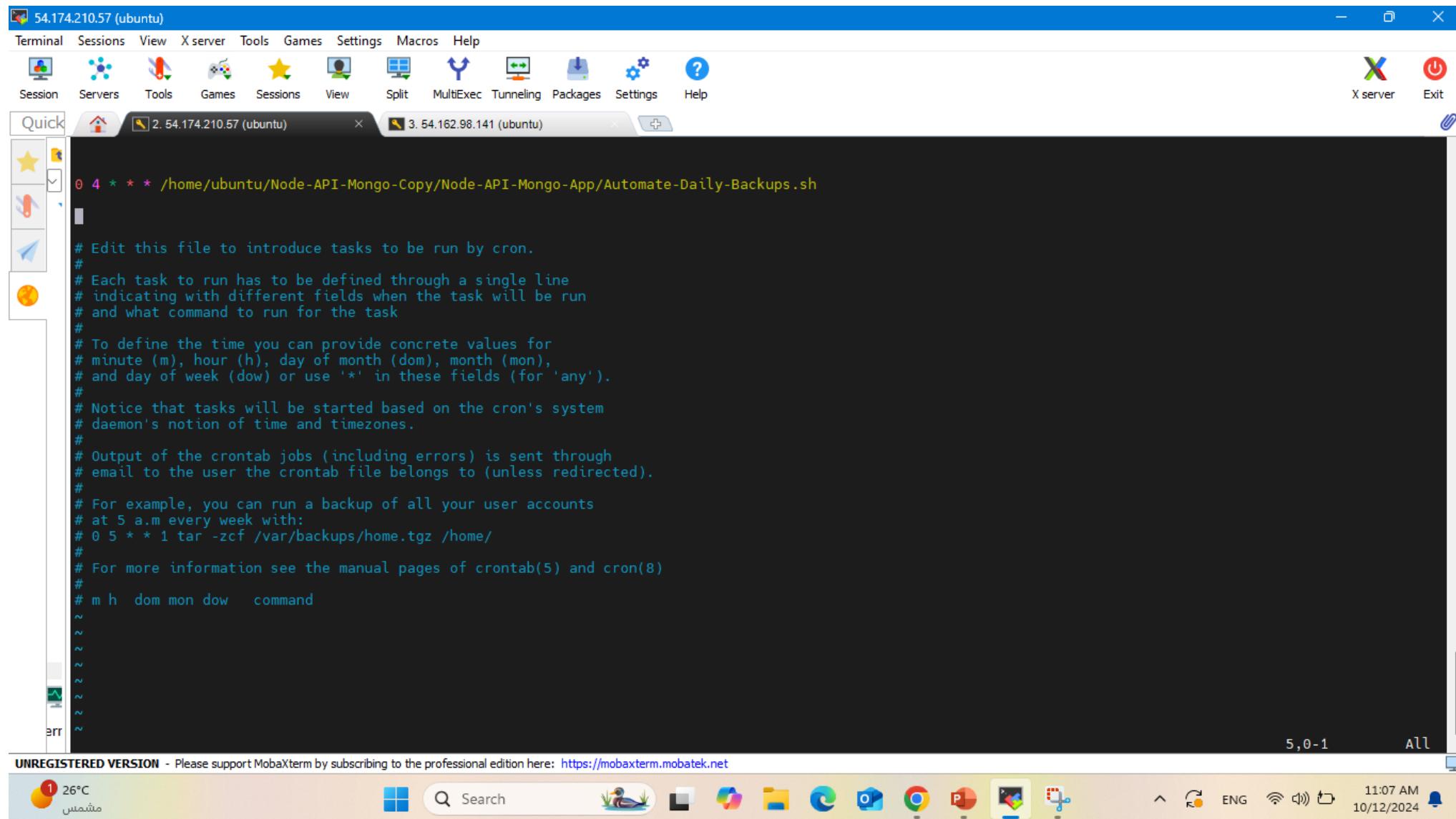


The screenshot shows a MobaXterm window titled "107.22.20.188 (ubuntu) (2)". The terminal session is running on port 107.22.20.188. The user has run the command "cat todos.bson" and the output is displayed:

```
s$ cd todos.bson
-bash: cd: todos.bson: Not a directory
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034/todo
s$ cat todos.bson
idg[titleTarget Server on AWSdescription]This is the application served from target server af
ter copying the files from the main server using scpis_complete due_date[ws] v_idg[titleSteps for Deploymentdescriptionq1. Run the first script to copy application files to target s
erver.
2. Run the second script to:
   1) ssh into destination server
   2) run docker-compose up to run application
3. Run the third script to send an HTTP request to the EC2 public IP to ensure the application is run
ning and returning a 200 OK status.
4. Run the fourth script to stop all containeris_complete      due_date[ws] v_id/4vp[title3Auto
mate Daily Backups of the Application's Data]description]By automating daily backups and scheduling it
with cron, This ensures that data is consistently backed up without manual intervention.is_completed
ubuntu@ip-172-31-40-204:~/Node-API-Mongo-Copy/Node-API-Mongo-App/backup/mongodump_20241011001034/todos$
```

At the bottom of the terminal window, there is a watermark: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

39- Create a cron job to automate daily backup at 4 am



The screenshot shows a MobaXterm window titled "54.174.210.57 (ubuntu)". The terminal tab displays a cron configuration script:

```
0 4 * * * /home/ubuntu/Node-API-Mongo-Copy/Node-API-Mongo-App/Automate-Daily-Backups.sh

# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
~ ~ ~ ~ ~
```

The status bar at the bottom indicates an unregistered version and shows system information like temperature (26°C), battery level (1), search bar, and system icons.

40- Problems I faced and solved

- 1- Authentication with git I made ssh
- 2- API URL was local host and has to be changed to Public IP of aws ec2
- 3- Run ssh in interactive shell
- 4- Connection to mongo dB to make backup

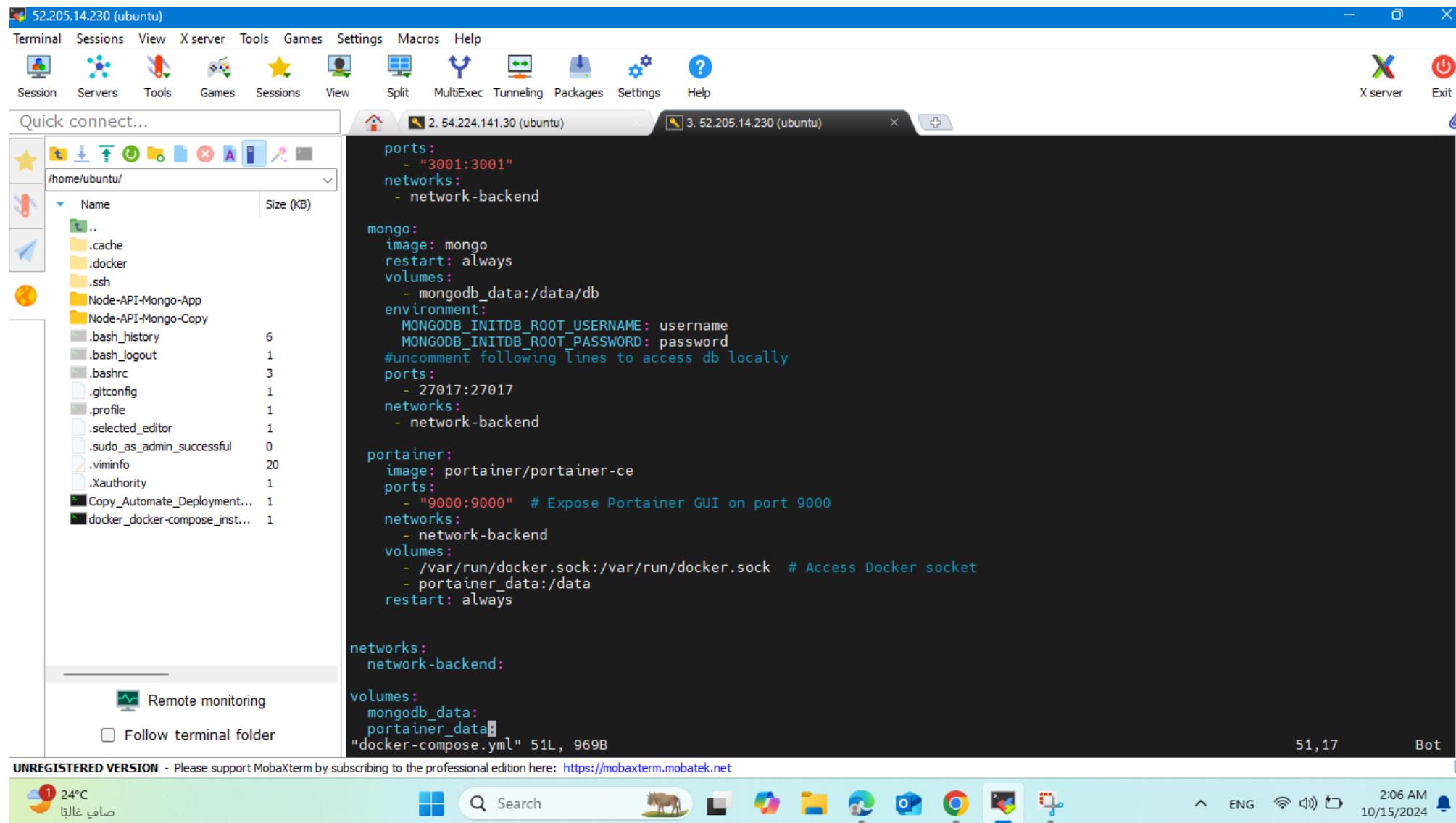
41- Github Repository

<https://github.com/salmasalam024/Node-API-Mongo-App>

Part 02

Monitoring

42- Add service in the docker-compose.yml file to run portainer



The screenshot shows a MobaXterm window titled "52.205.14.230 (ubuntu)". The terminal pane displays a `docker-compose.yml` configuration file. The file defines services for a MongoDB database and a Portainer UI, both connected via a shared network named "network-backend". The MongoDB service runs on port 3001. The Portainer service runs on port 9000 and exposes the Docker socket at /var/run/docker.sock.

```
ports:
  - "3001:3001"
networks:
  - network-backend

mongo:
  image: mongo
  restart: always
  volumes:
    - mongodb_data:/data/db
  environment:
    MONGODB_INITDB_ROOT_USERNAME: username
    MONGODB_INITDB_ROOT_PASSWORD: password
  #uncomment following lines to access db locally
  ports:
    - 27017:27017
  networks:
    - network-backend

portainer:
  image: portainer/portainer-ce
  ports:
    - "9000:9000" # Expose Portainer GUI on port 9000
  networks:
    - network-backend
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock # Access Docker socket
    - portainer_data:/data
  restart: always

networks:
  network-backend:

volumes:
  mongodb_data:
  portainer_data:
"docker-compose.yml" 51L, 969B
```

The left sidebar of the MobaXterm interface shows a file browser with the current directory set to `/home/ubuntu/`. The file list includes standard Linux configuration files like .cache, .docker, .ssh, and various shell history and configuration files. There are also some temporary files related to the deployment process.

43- Portainer Successfully running on port 9000

New Portainer installation

Please create the initial administrator user.

Username: admin

Password: (redacted)

Confirm password: (redacted) ✓

The password must be at least 12 characters long. ✓

Create user

Allow collection of anonymous statistics. You can find more information about this in our [privacy policy](#).

Restore Portainer from backup

24°C صافي غالباً

Search

2:05 AM 10/15/2024 ENG

43- Portainer Successfully running on port 9000

Portainer

Not secure 52.205.14.230:9000/#!/wizard

Gmail Console Home | Co... Home - Microsoft A... (3479) DevOpsified |... DevOps Beginners t... (3004) All About RH... DevOps Track - Goo... Your Repositories All Bookmarks

Upgrade to Business Edition

portainer.io COMMUNITY EDITION

Home

Environment: None selected

Administration

User-related

Environment-related

Registries

Logs

Notifications

Settings

portainer.io Community Edition 2.21.3

Environment Wizard

Quick Setup

admin

Welcome to Portainer

We have connected your local environment of Docker to Portainer.

Get started below with your local portainer or connect more container environments.

Get Started

Proceed using the local environment which Portainer is running in

Add Environments

Connect to other environments

1 24°C صافي غالباً

Search

2:07 AM 10/15/2024

44- Docker in portainer

The screenshot shows the Portainer web interface running in a browser. The title bar indicates the URL is 52.205.14.230:9000#!/endpoints. The main content area is titled "Environments" and displays a table of environment configurations. The table has columns for Name, Type, URL, Group Name, and Actions. One entry is visible: "local" (Type: Docker, URL: unix:///var/run/docker.sock, Group Name: Unassigned). There is a search bar and a "Remove" button for each row. The sidebar on the left is titled "Administration" and includes sections for User-related, Environment-related (selected), Groups, Tags, Registries, Logs, and Notifications. The bottom status bar shows system information like temperature, battery level, and network connectivity.

Name ↓↑	Type ↓↑	URL ↓↑	Group Name ↓↑	Actions
local	Docker	unix:///var/run/docker.sock	Unassigned	Manage access

Environment management

Environments

Search... Remove Add environment

Name ↓↑ Type ↓↑ URL ↓↑ Group Name ↓↑ Actions

local Docker unix:///var/run/docker.sock Unassigned [Manage access](#)

Items per page 10

None selected

Home

Administration

User-related

Environment-related

Groups

Tags

Registries

Logs

Notifications

portainer.io Community Edition 2.21.3

24°C صافي غالباً

Search

ENG

2:08 AM

10/15/2024

44- Docker in portainer

The screenshot shows the Portainer.io Community Edition dashboard for a local environment. The environment summary indicates a local setup with 1 CPU, 1 GB memory, and Standalone version 27.3.1. The URL is set to /var/run/docker.sock. GPU usage is listed as none. There are no tags present.

Environment info:

Environment	local	1 CPU	1 GB - Standalone 27.3.1
URL	/var/run/docker.sock		
GPU	none		
Tags	-		

Resource Statistics:

1 Stack	4 Containers
8 Images	15 Volumes
2.7 GB	

Status Summary:

4 running	0 stopped
0 healthy	0 unhealthy

Left Sidebar:

- Home
- local (selected)
- Dashboard
- Templates
- Stacks
- Containers
- Images
- Networks
- Volumes
- Events
- Host

Bottom Bar:

- Shari' al-Nasr / E... طریق مغلق
- Search
- File Explorer
- Terminal
- File Manager
- Logs
- Docker Engine
- Compose
- Portainer

System tray icons include: battery (2:10 AM), ENG, WiFi, signal strength, and date (10/15/2024).

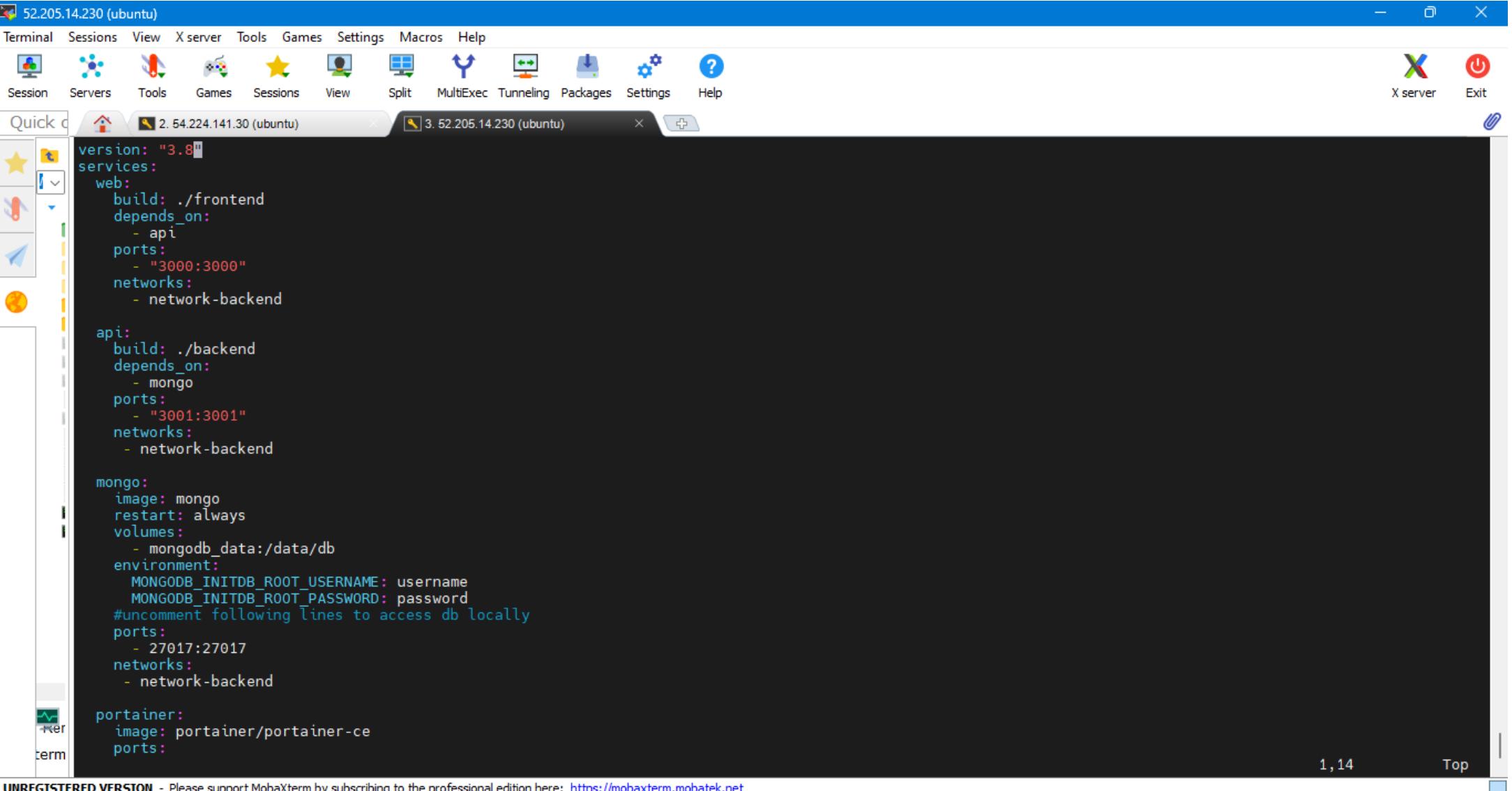
44- Docker containers in portainer

The screenshot shows the Portainer.io interface running in a browser window. The title bar indicates the URL is 52.205.14.230:9000/#!/2/docker/containers. The sidebar on the left is titled 'local' and includes links for Home, Dashboard, Templates, Stacks, Containers (which is selected), Images, Networks, Volumes, Events, and Host. The main content area is titled 'Container list' and displays a table of running containers. The table columns are: Name, State, Quick Actions, Stack, Image, Created, IP Address, and Published Port. There are five entries:

Name	State	Quick Actions	Stack	Image	Created	IP Address	Published Port
node-api-mongo-app_api_1	running	Restart, Stop, Kill, Restart, Pause, Resume, Remove	node-api-mongo-app	node-api-mongo-app_api	2024-10-15 02:01:09	172.18.0.4	3001:3001
node-api-mongo-app_mongo_1	running	Restart, Stop, Kill, Restart, Pause, Resume, Remove	node-api-mongo-app	mongo	2024-10-15 02:01:08	172.18.0.3	27017:27017
node-api-mongo-app_portainer_1	running	Restart, Stop, Kill, Restart, Pause, Resume, Remove	node-api-mongo-app	portainer/portainer-ce	2024-10-15 02:01:08	172.18.0.2	9000:9000
node-api-mongo-app_web_1	running	Restart, Stop, Kill, Restart, Pause, Resume, Remove	node-api-mongo-app	node-api-mongo-app_web	2024-10-15 02:01:10	172.18.0.5	3000:3000

At the bottom of the interface, there is a toolbar with icons for search, file operations, and system status, along with a footer showing the date and time (2:11 AM, 10/15/2024).

45- Adding service in the docker-compose.yml file to run portainer



The screenshot shows a MobaXterm window titled "52.205.14.230 (ubuntu)". The terminal pane displays a `docker-compose.yml` configuration file. The file defines services for a frontend web application, an API, a MongoDB database, and a Portainer instance. The Portainer service is explicitly added at the bottom of the file.

```
version: "3.8"
services:
  web:
    build: ./frontend
    depends_on:
      - api
    ports:
      - "3000:3000"
    networks:
      - network-backend

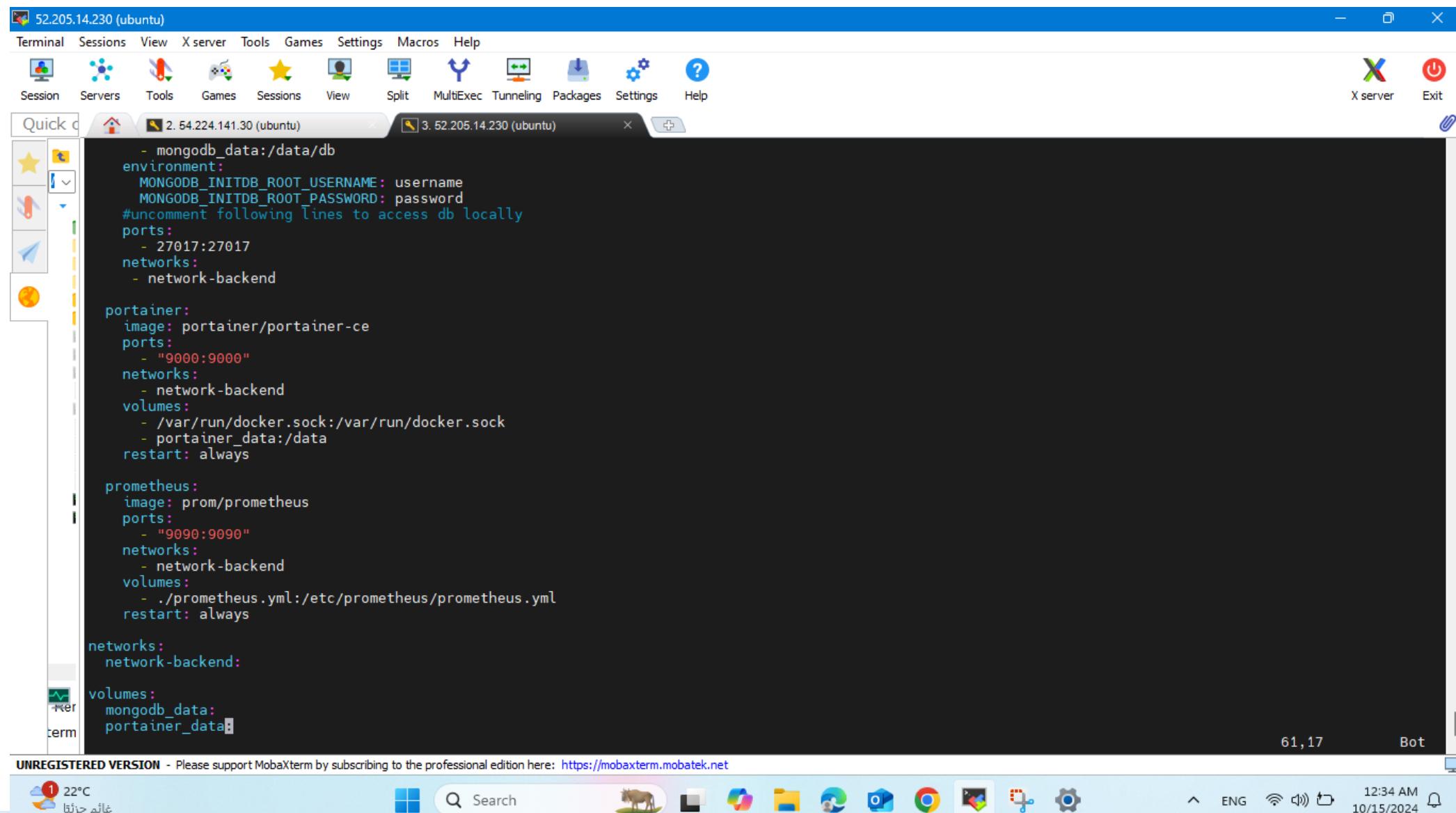
  api:
    build: ./backend
    depends_on:
      - mongo
    ports:
      - "3001:3001"
    networks:
      - network-backend

  mongo:
    image: mongo
    restart: always
    volumes:
      - mongodb_data:/data/db
    environment:
      MONGODB_INITDB_ROOT_USERNAME: username
      MONGODB_INITDB_ROOT_PASSWORD: password
    #uncomment following lines to access db locally
    ports:
      - 27017:27017
    networks:
      - network-backend

  portainer:
    image: portainer/portainer-ce
    ports:
```

At the bottom of the terminal, there is a message: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

45- Adding service in the docker-compose.yml file to run portainer



The screenshot shows a terminal window in MobaXterm with the title "52.205.14.230 (ubuntu)". The window displays a portion of a Docker Compose configuration file (`docker-compose.yml`). The file defines services for MongoDB, Portainer, Prometheus, and a network named "network-backend". The MongoDB service uses environment variables for root user and password, and maps port 27017 to 27017. The Portainer service uses the image `portainer/portainer-ce`, maps port 9000 to 9000, and volumes `/var/run/docker.sock` and `portainer_data:/data`. The Prometheus service uses the image `prom/prometheus`, maps port 9090 to 9090, and volumes `./prometheus.yml:/etc/prometheus/prometheus.yml`. The network section defines a network named "network-backend". The volumes section defines two volumes: `mongodb_data:` and `portainer_data:`.

```
mongodbs_data:/data/db
environment:
  MONGODB_INITDB_ROOT_USERNAME: username
  MONGODB_INITDB_ROOT_PASSWORD: password
#uncomment following lines to access db locally
ports:
  - 27017:27017
networks:
  - network-backend

portainer:
  image: portainer/portainer-ce
  ports:
    - "9000:9000"
  networks:
    - network-backend
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock
    - portainer_data:/data
  restart: always

prometheus:
  image: prom/prometheus
  ports:
    - "9090:9090"
  networks:
    - network-backend
  volumes:
    - ./prometheus.yml:/etc/prometheus/prometheus.yml
  restart: always

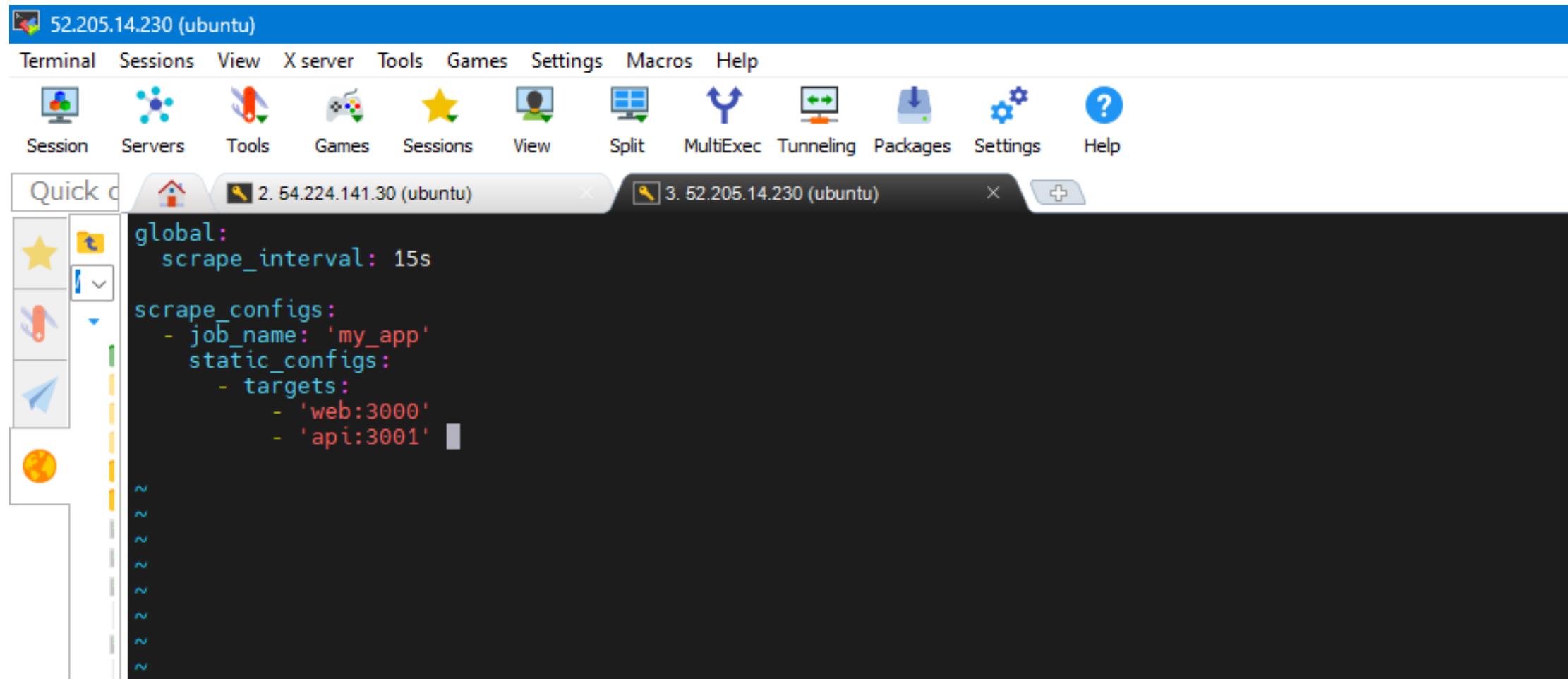
networks:
  network-backend:

volumes:
  mongodb_data:
  portainer_data:
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

22°C غائم جزئی 12:34 AM 10/15/2024

46- Prometheus.yml file with targets declaration



The screenshot shows a desktop interface with a blue header bar. The title bar of the active window displays "52.205.14.230 (ubuntu)". The menu bar includes "Terminal", "Sessions", "View", "X server", "Tools", "Games", "Settings", "Macros", and "Help". Below the menu is a toolbar with icons for "Session", "Servers", "Tools", "Games", "Sessions", "View", "Split", "MultiExec", "Tunneling", "Packages", "Settings", and a question mark icon. The main window contains a terminal session with the following content:

```
global:
  scrape_interval: 15s

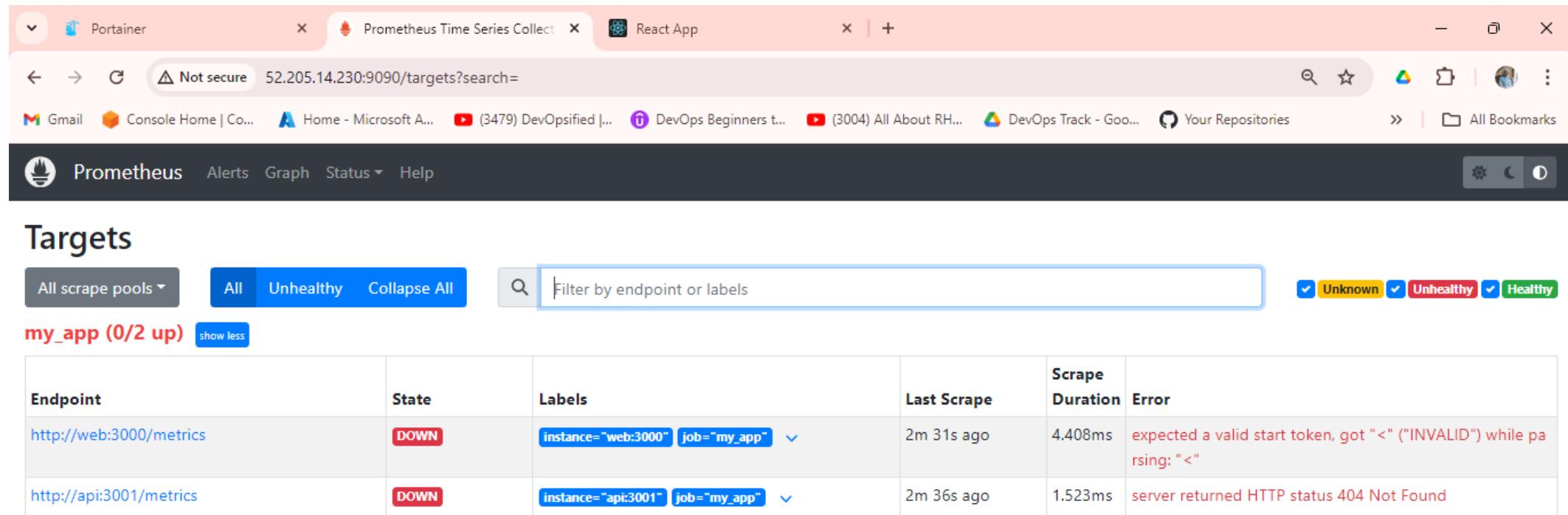
scrape_configs:
  - job_name: 'my_app'
    static_configs:
      - targets:
          - 'web:3000'
          - 'api:3001'
```

47- An error in time synchronization between the server and browser

The screenshot shows a web browser window with the following details:

- Browser Tabs:** Portainer, Prometheus Time Series Collector (active tab), React App.
- Address Bar:** Not secure 52.205.14.230:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_input=1h
- Toolbar:** Includes icons for Gmail, Console Home, Microsoft Edge, Home - Microsoft A..., (3479) DevOpsified, DevOps Beginners, (3004) All About RH..., DevOps Track, Your Repositories, and All Bookmarks.
- Prometheus UI Header:** Prometheus, Alerts, Graph, Status, Help.
- Configuration:** Checkboxes for "Use local time", "Enable query history", "Enable autocomplete" (checked), "Enable highlighting" (checked), and "Enable linter" (checked).
- Warning Message:** Warning: Error fetching server time: Detected 148.08500003814697 seconds time difference between your browser and the server. Prometheus relies on accurate time and time drift might cause unexpected query results.
- Search Bar:** Expression (press Shift+Enter for newlines) with Execute button.
- Panel Options:** Table (selected), Graph, Evaluation time (with back and forward arrows), and Remove Panel.
- Bottom Buttons:** Add Panel.

47- An error in time synchronization between the server and browser

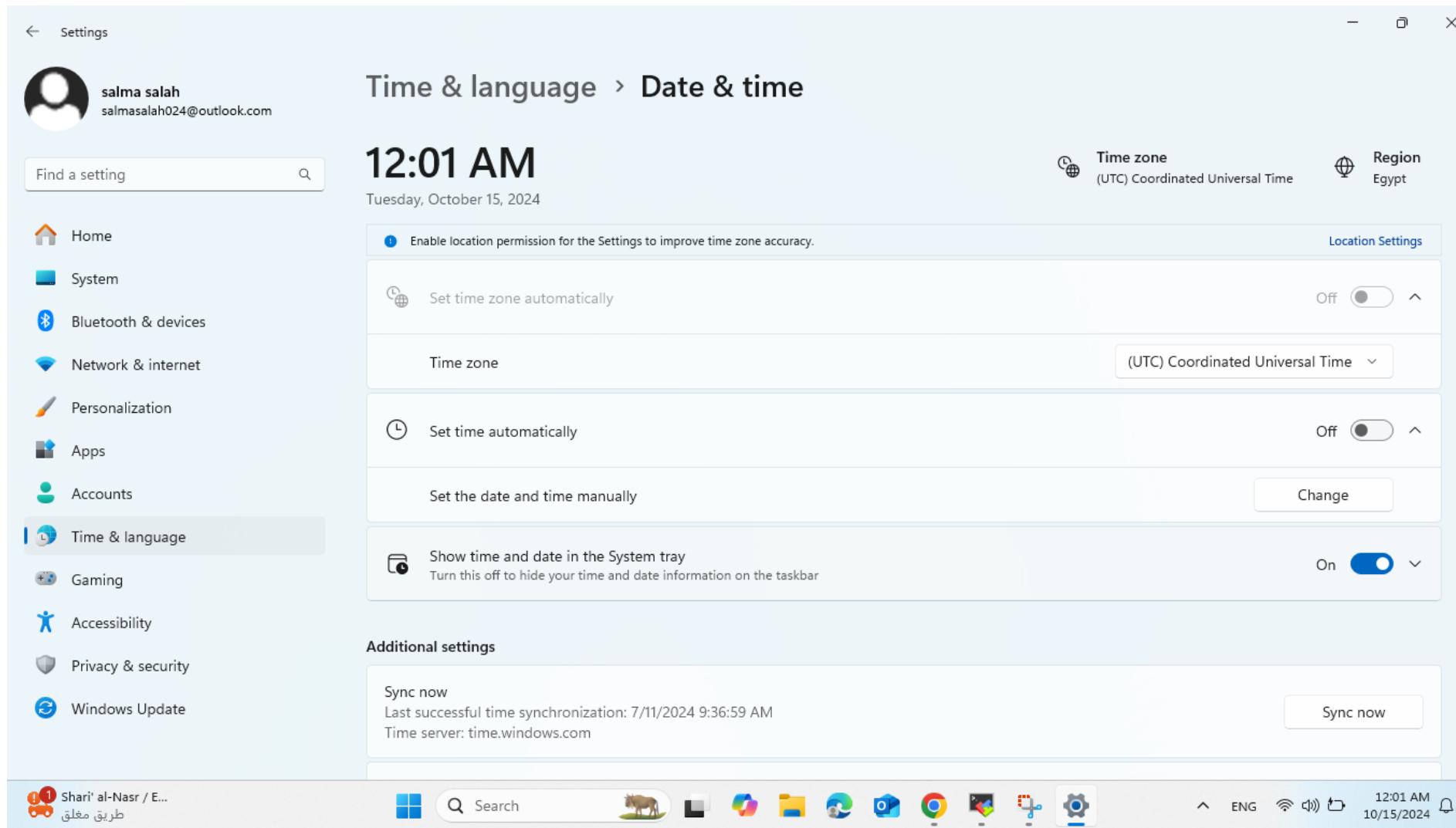


The screenshot shows a web browser window with three tabs open: Portainer, Prometheus Time Series Collector, and React App. The Prometheus tab is active, displaying the 'Targets' page. The URL in the address bar is 52.205.14.230:9090/targets?search=. The page title is Prometheus.

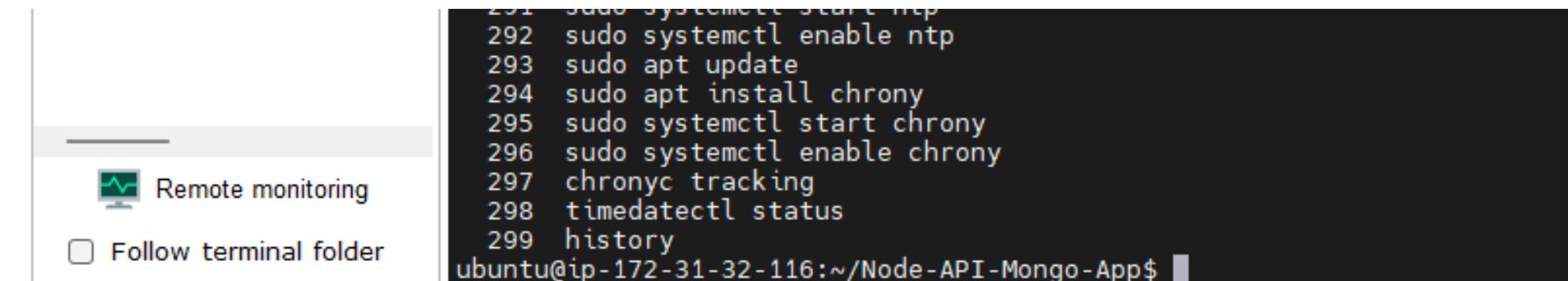
The 'Targets' section shows two entries for 'my_app':

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://web:3000/metrics	DOWN	instance="web:3000" job="my_app"	2m 31s ago	4.408ms	expected a valid start token, got < ("INVALID") while parsing: <
http://api:3001/metrics	DOWN	instance="api:3001" job="my_app"	2m 36s ago	1.523ms	server returned HTTP status 404 Not Found

48- Changing system's time to be UTC as the server



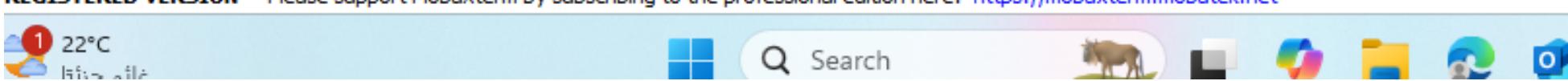
49- Installing chrony



The screenshot shows a terminal window in MobaXterm. On the left, there's a sidebar with a 'Remote monitoring' icon and a checkbox for 'Follow terminal folder'. The main terminal area displays the following command sequence:

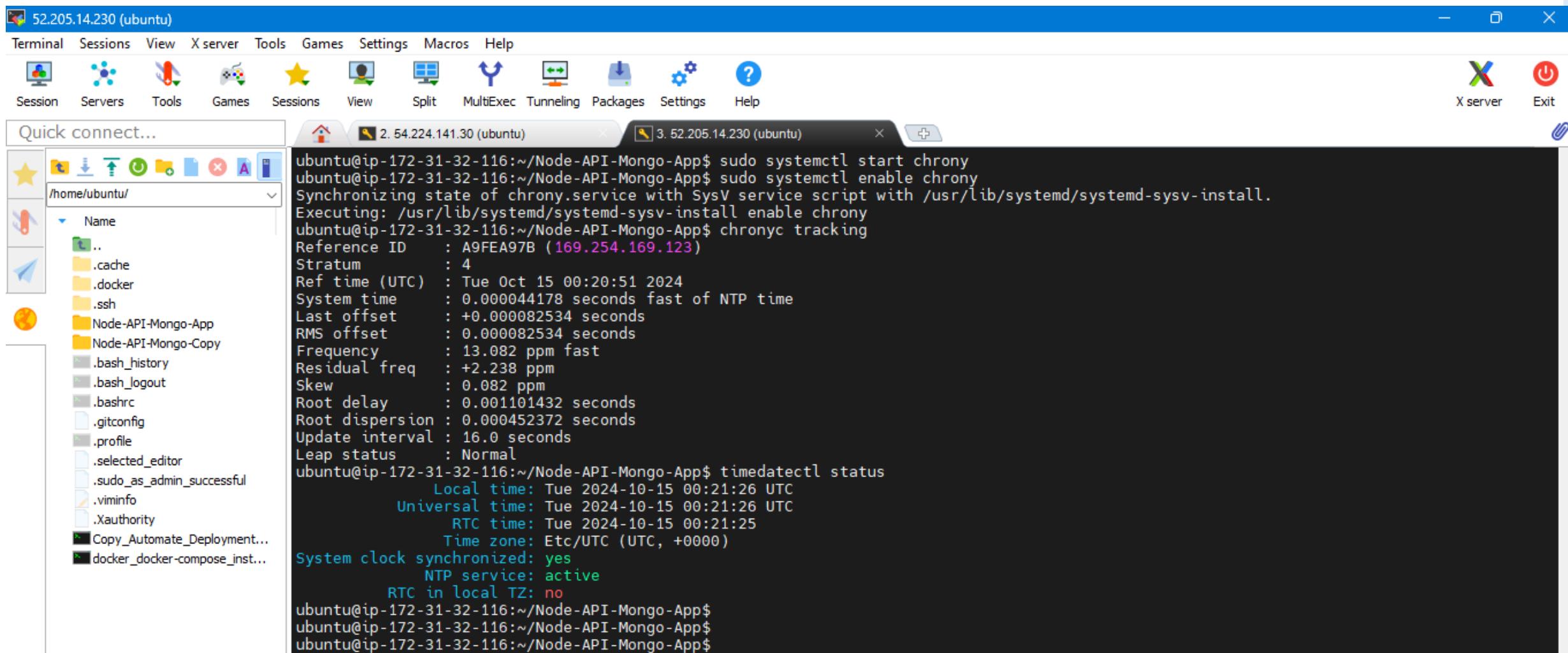
```
291 sudo systemctl start ntp  
292 sudo systemctl enable ntp  
293 sudo apt update  
294 sudo apt install chrony  
295 sudo systemctl start chrony  
296 sudo systemctl enable chrony  
297 chronyc tracking  
298 timedatectl status  
299 history  
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

At the bottom of the terminal window, there's a watermark: 'REGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.



The taskbar at the bottom of the screen shows several pinned icons: a weather widget (22°C), a Microsoft Edge browser icon, a search bar, a file explorer icon, a task manager icon, a user profile icon, and a OneDrive icon.

50- Checking synchronization



The screenshot shows a desktop environment with a terminal window open. The terminal window title is "3. 52.205.14.230 (ubuntu)". The terminal content displays the output of several commands related to system synchronization:

```
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ sudo systemctl start chrony
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ sudo systemctl enable chrony
Synchronizing state of chrony.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable chrony
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ chronyc tracking
Reference ID      : A9FEA97B (169.254.169.123)
Stratum          : 4
Ref time (UTC)   : Tue Oct 15 00:20:51 2024
System time      : 0.000044178 seconds fast of NTP time
Last offset      : +0.000082534 seconds
RMS offset       : 0.000082534 seconds
Frequency        : 13.082 ppm fast
Residual freq    : +2.238 ppm
Skew              : 0.082 ppm
Root delay       : 0.001101432 seconds
Root dispersion  : 0.000452372 seconds
Update interval  : 16.0 seconds
Leap status       : Normal
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$ timedatectl status
          Local time: Tue 2024-10-15 00:21:26 UTC
          Universal time: Tue 2024-10-15 00:21:26 UTC
                    RTC time: Tue 2024-10-15 00:21:25
                   Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: yes
                         NTP service: active
                RTC in local TZ: no
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
ubuntu@ip-172-31-32-116:~/Node-API-Mongo-App$
```

50- Checking synchronization

The screenshot shows a web browser window with multiple tabs open. The active tab is titled "What is my Timezone - WebBro" and displays the URL "webbrowsertools.com/timezone/". The page content is a table titled "Timezone info" with the following data:

Time on Server	Tue, 15 Oct 2024 01:21:20 GMT
Time on Local Machine	Tue Oct 15 2024 01:21:21 GMT+0000 (Coordinated Universal Time)
Time from Intl.DateTimeFormat #1	Tuesday, October 15, 2024 at 1:21:21 AM UTC
Time from Intl.DateTimeFormat #2	Tuesday, October 15, 2024 at 1:21:21 AM Coordinated Universal Time
Internet Beat	around @98
Timezone	UTC
Timezone Offset	0 minutes
Numbering System	latn
Locale	en-US
Calendar	gregory
Day	numeric
Month	numeric
Year	numeric
Milliseconds since January 1, 1970	1728955281153

At the bottom of the page, there is a cookie consent message: "We use cookies to enhance your experience. By continuing to visit this site you agree to our use of cookies. [Learn more](#)". There are "Got it!" and "Decline" buttons next to it.

The browser's taskbar at the bottom shows various pinned icons and the system tray with the date and time (10/15/2024, 1:21 AM).

51- Prometheus Successfully running

The screenshot shows a web browser window with the following details:

- Tab Bar:** Portainer, React App, What is my Timezone - WebBro, Prometheus Time Series Collector.
- Address Bar:** Not secure 52.205.14.230:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_input=1h
- Toolbar:** Back, Forward, Stop, Refresh, Home, DevTools, Bookmarks, Profile, More.
- Prometheus Header:** Prometheus, Alerts, Graph, Status, Help.
- Query Settings:** Use local time (unchecked), Enable query history (unchecked), Enable autocomplete (checked), Enable highlighting (checked), Enable linter (checked).
- Search Bar:** Expression (press Shift+Enter for newlines).
- Graph Panel:** Evaluation time, Graph tab selected, Table tab (disabled). Message: No data queried yet.
- Buttons:** Execute, Remove Panel, Add Panel.

51- Prometheus Successfully running

The screenshot shows the Portainer.io interface running on a local host. The left sidebar navigation bar includes Home, local, Dashboard, Templates, Stacks, Containers (which is currently selected), Images, Networks, Volumes, Events, and Host. The main content area displays a table titled "Container list" with the following data:

Name	State	Image	Created	IP Address	Published Ports
node-api-mongo-app_api_1	running	node-api-mongo-app	2024-10-15 00:24:49	172.18.0.5	3001:3001
node-api-mongo-app_mongo_1	running	node-api-mongo-app	2024-10-15 00:24:48	172.18.0.3	27017:2701
node-api-mongo-app_portainer_1	running	node-api-mongo-app	2024-10-15 00:24:48	172.18.0.2	9000:9000
node-api-mongo-app_prometheus_1	running	prom/prometheus	2024-10-15 00:24:48	172.18.0.4	9090:9090
node-api-mongo-app_web_1	running	node-api-mongo-app	2024-10-15 00:24:50	172.18.0.6	3000:3000

The bottom status bar shows system information including weather (22°C), search bar, taskbar icons (File Explorer, File Manager, Task View, Google Chrome, Microsoft Edge, File History, Task Scheduler, Control Panel, and Settings), language (ENG), battery level, signal strength, and the date/time (10/15/2024, 12:35 AM).

52- Next Steps

- Install (prom-client) to expose metrics in a format that Prometheus can scrape
- Set Up the Metrics Endpoint in the applications file
- Add service for Grafana to Visualize metrics