**MAJOR PROJECT-1**

**DEPLOYING WORDPRESS WEB APPLICATION USING DOCKER IN**

**AMAZON WEB SERVICES**

# INTRODUCTION

We'll deploy **WordPress** via **docker-composer** onto the AWS EC2 instance (**t2.micro**) and access it with a domain name defined in **Route53**.

WordPress is the easiest way to manage and create content. Its flexibility is loved by authors: with a couple of plugins, you can do everything from hosting a cute kitten’s photo gallery to hosting an e-commerce site.

Let’s face it: seen from the IT guy point of view, WordPress is a technical nightmare. When someone has to deal with it the horror begins: scalability is challenging, installation isn’t scripted and a LAMP stack is not always easy to maintain.

In this article, we’ll give you some technical hints and examples to ease your relationship with WordPress in a cloud environment based on AWS.

We'll try to use as many AWS-managed services as we can to be able to offload boring and dangerous tasks.

# PRE-REQUISITES:

Amazon Web Services Account GitBash Tool



GitHub Account



# Steps to creating the infrastructure in this pipeline/module

Creating and launching an EC2 Instance with AMI – Amazon Linux 2 Installing GIT, Docker, and related repos



Creating Docker images with help of YAML scripting Creating EIP for launching the output of the project statically.

# What is AWS?

Amazon Web Services (AWS) is the world’s most comprehensive and broadly adopted cloud platform, offering over 200 fully featured services from data centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.

# What is WordPress?

WordPress is a free and open-source content management system written in PHP and paired with a MySQL or MariaDB database with supported HTTPS. Features include a plugin architecture and a template system, referred to within WordPress as Themes. WordPress is a content management system (CMS) that allows you to host and build websites. WordPress contains plugin architecture and a template system, so you can customize any website to fit your business, blog, portfolio.

# What is GitBash?

Git Bash is an application for Microsoft Windows environments which provides an emulation layer for a Git command line experience. Bash is an acronym for Bourne Again Shell. A shell is a terminal application used to interface with an operating system through written commands.

# What is GitHub?

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. This tutorial teaches you GitHub essentials like repositories, branches, commits, and pull requests.

**Module – 1: Now Creating and launching an Amazon Linux EC2 instance**

EC2: Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware upfront, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.



# 2. Choosing an AMI – Amazon Linux 2 AMI

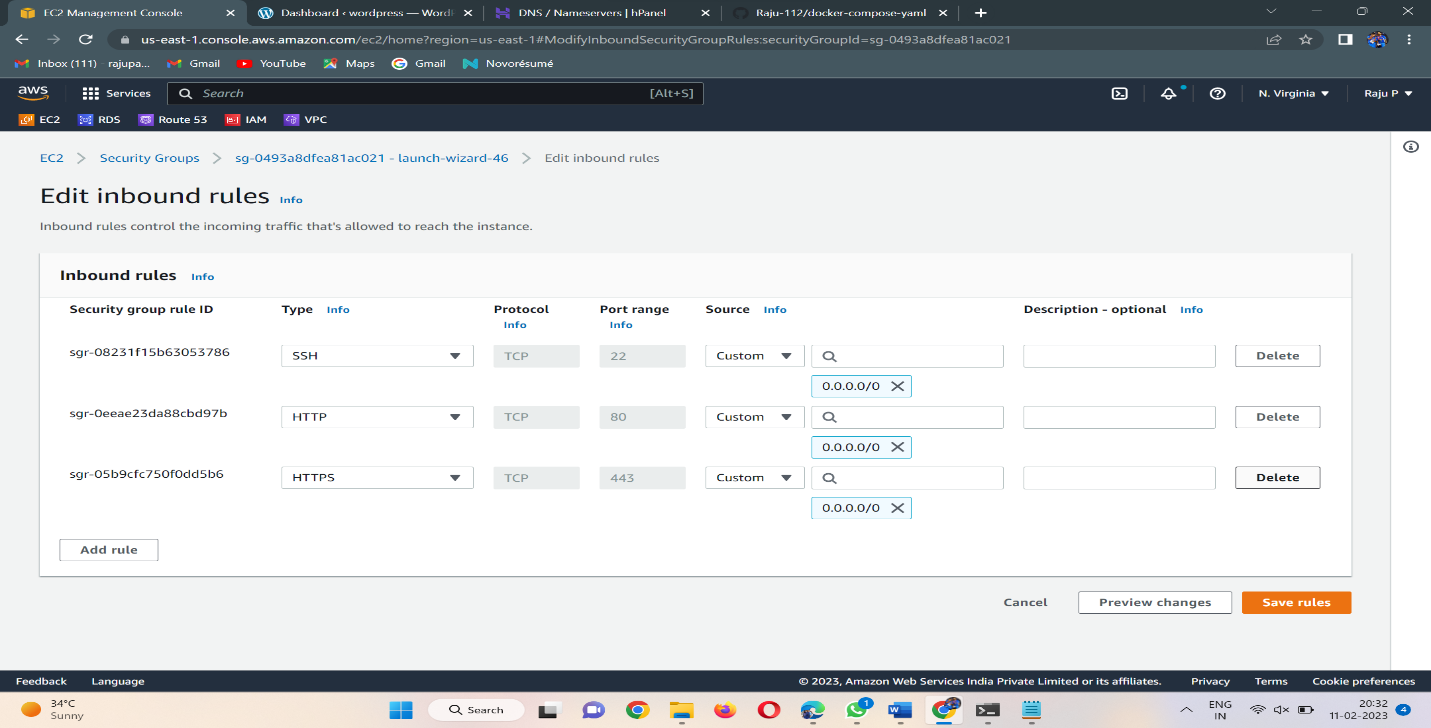
* An AMI is a virtual image used to create a virtual machine within an EC2 instance.
* You can also create multiple instances using a single AMI when you need instances with the same configuration.
* You can also create multiple instances using different AMI when you need instances with a different configuration

# 

# 3. Configure the Security Group

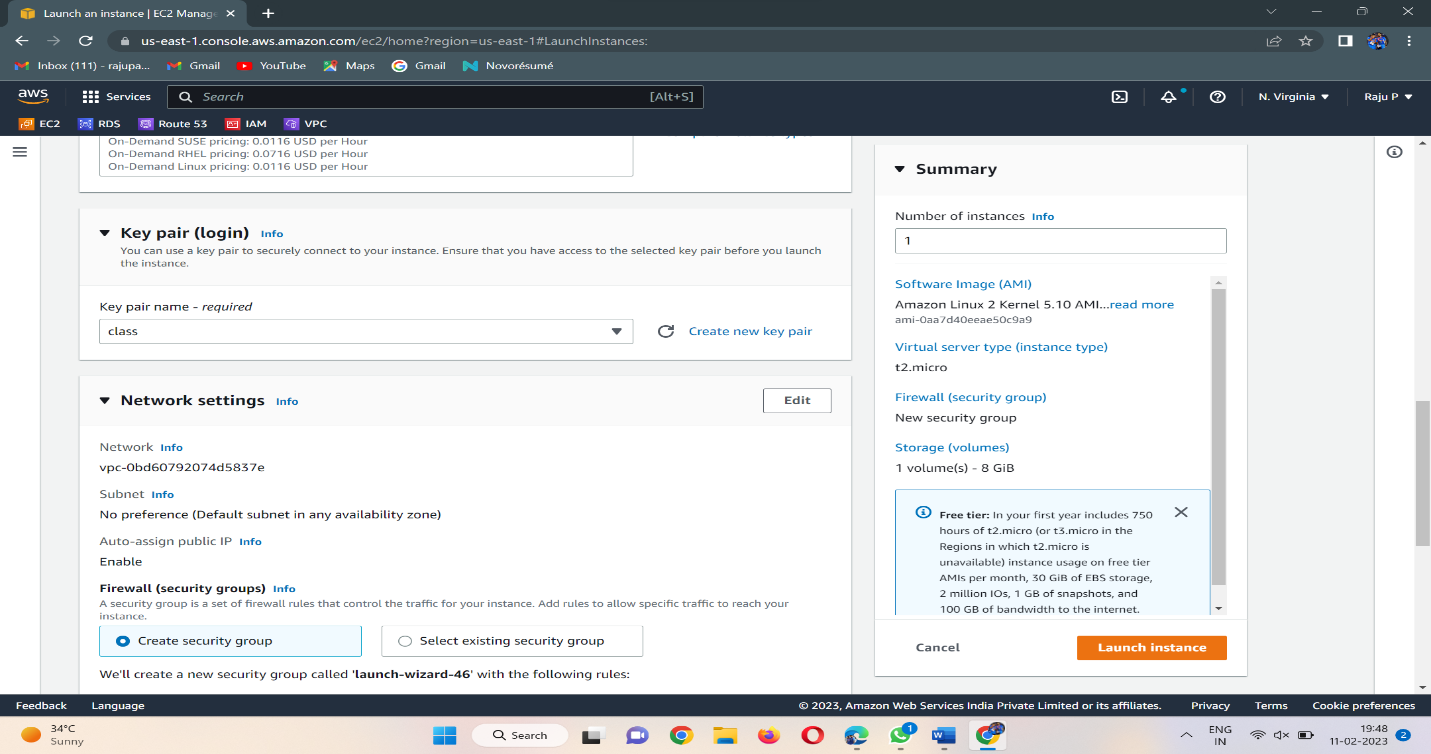
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below

Here, I am assigning the port numbers for the inbound rule SSH-22, HTTP-80, HTTPS-443, and All Traffic for the outbound rule.

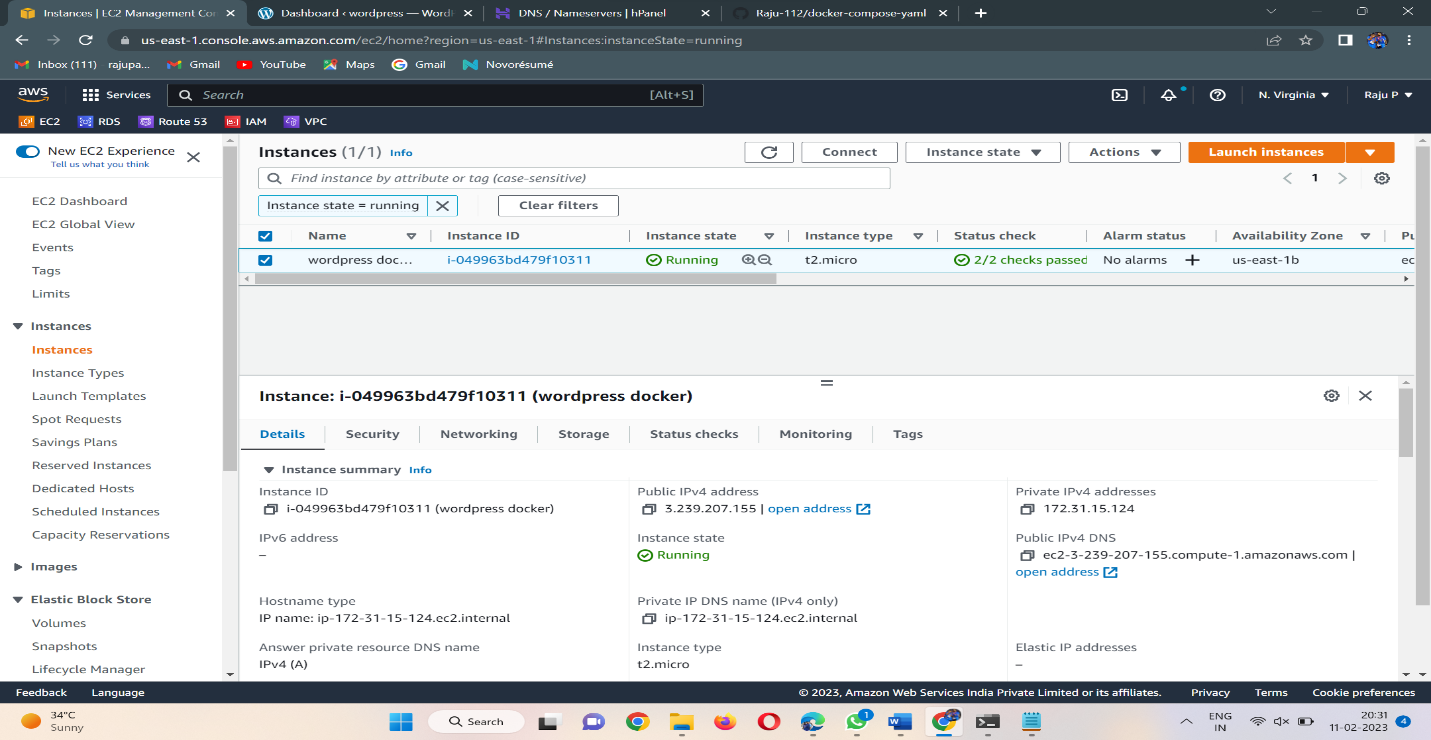


# 4 Select an Existing keypair or create a new key pair.

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types.



* Finally, Instance launched a WordPress-Server Instance. Now to connect an EC2 instance



1. Now to the instance page and click on connect then go to SSH Client/EC2 instance connect.

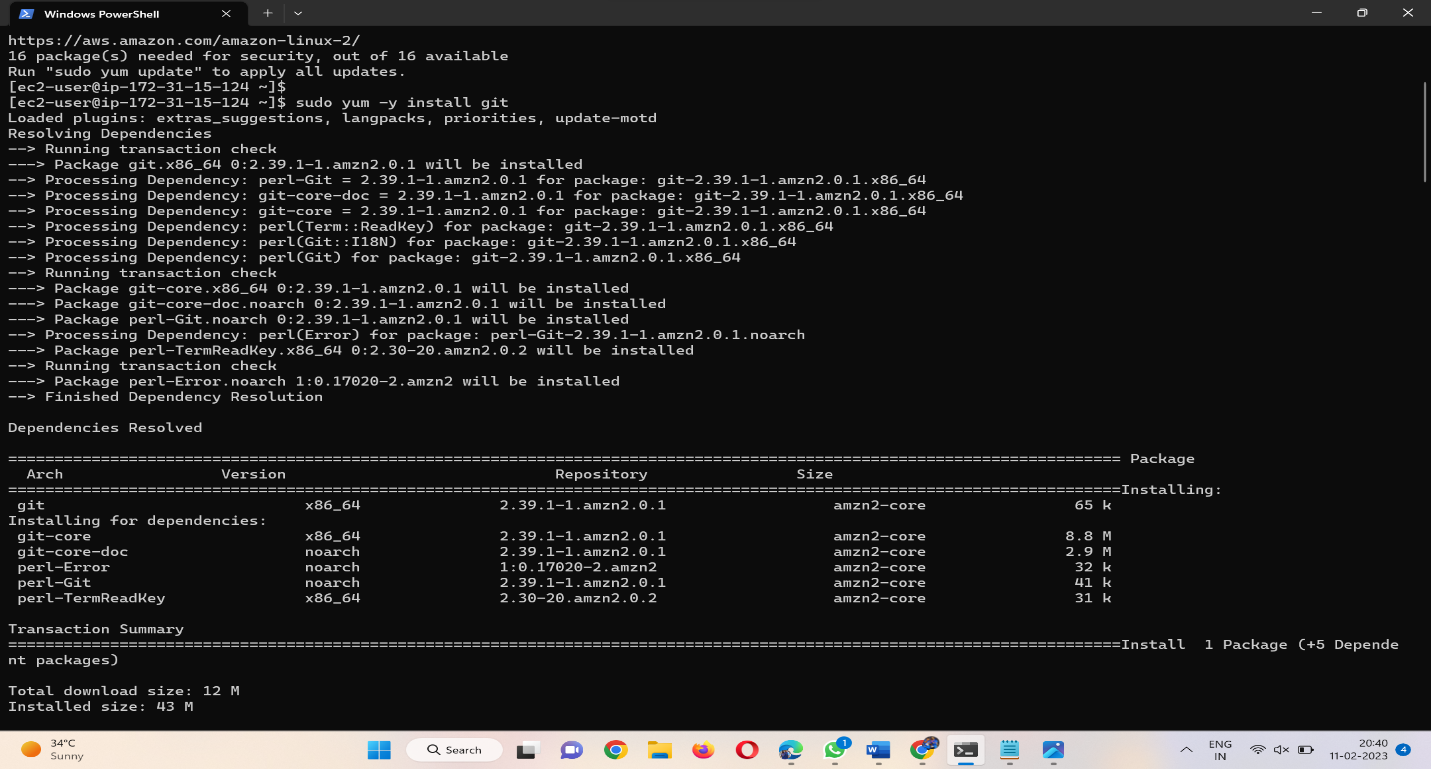
* By using the SSH command we are accessing our instance in Command Line Interface Terminal.



# Module – 2: Installing GIT, Docker, and related repos

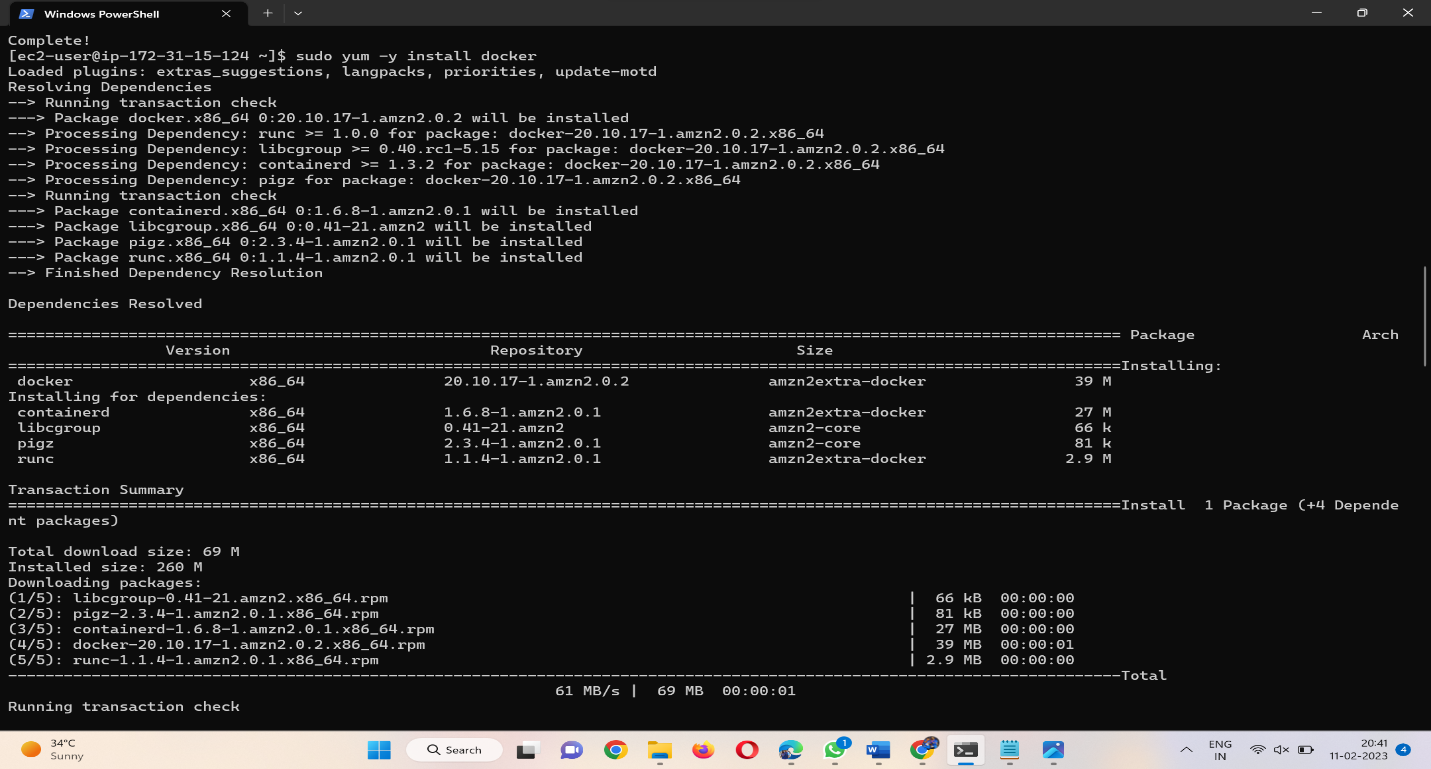
1. Installing GIT in our instance by using below command.

* Sudo yum -y install git



1. Installing Docker in our instance by using the below command.

* Sudo yum -y install docker



# Give the permission to add a limited Linux user account to the "docker" group by using the below start the docker &configure the docker.

* sudo usermod -a -G docker ec2-user
* sudo service docker start
* sudo chkconfig docker on

# 

# Install Docker Compose

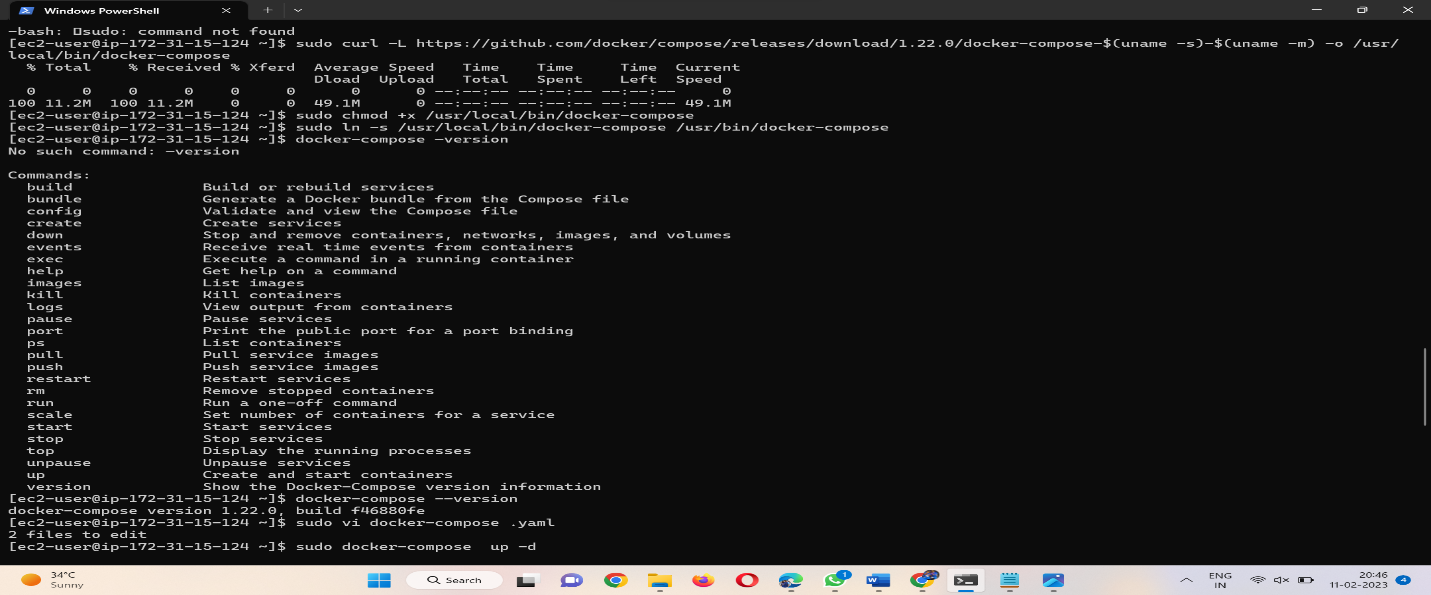
1. Download the latest version of Docker Compose (Install Docker Compose). Run this command to download the current stable release of Docker Compose by using the below command

# sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose

# 

# Apply executable permissions to the binary and create a symbolic link, check the version.

* sudo chmod +x /usr/local/bin/docker-compose
* ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
* docker-compose –version



# Module – 3: Creating WordPress setup for Docker images with help of YAML scripting

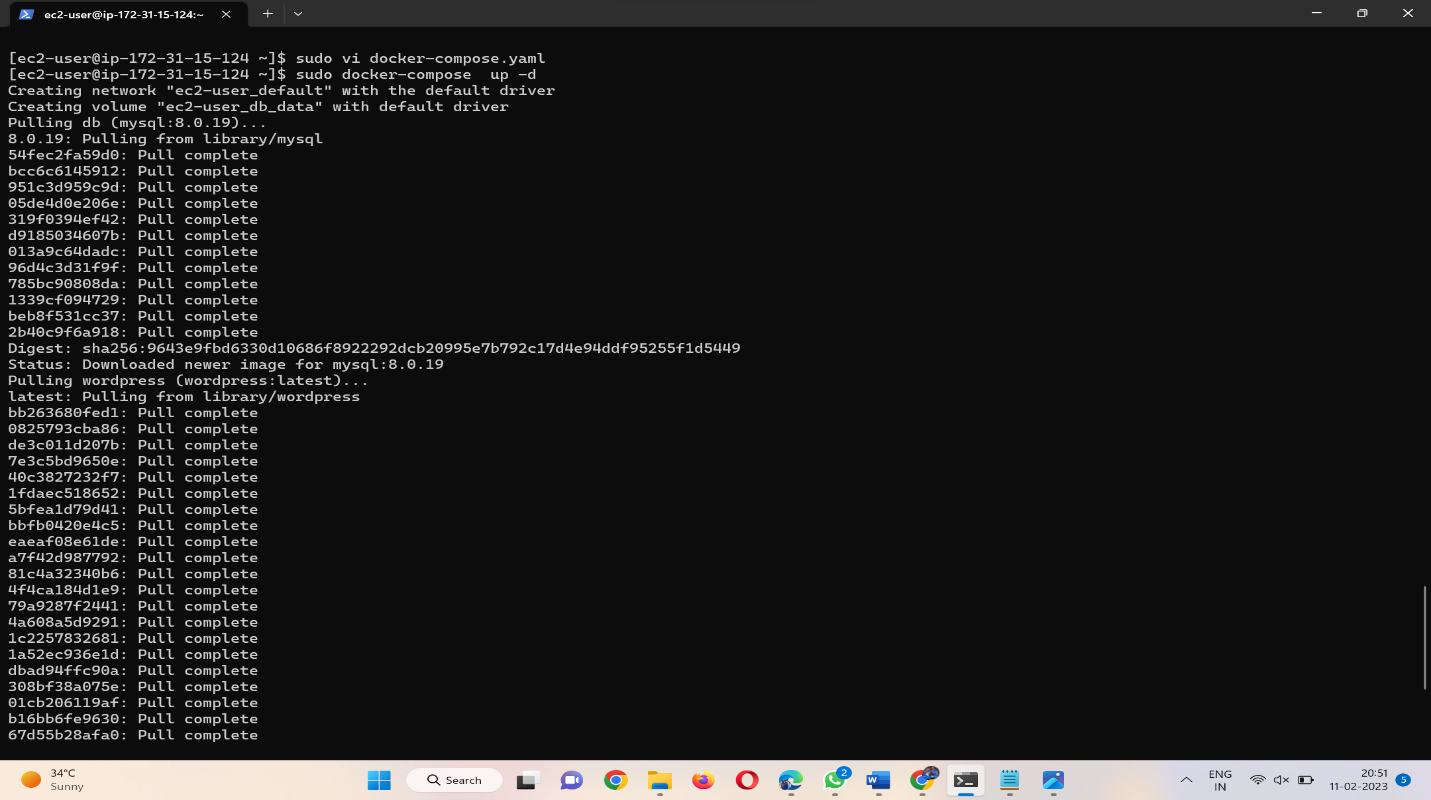
1. Here, we have to create a docker-compose.yaml file to pull images from docker hub:

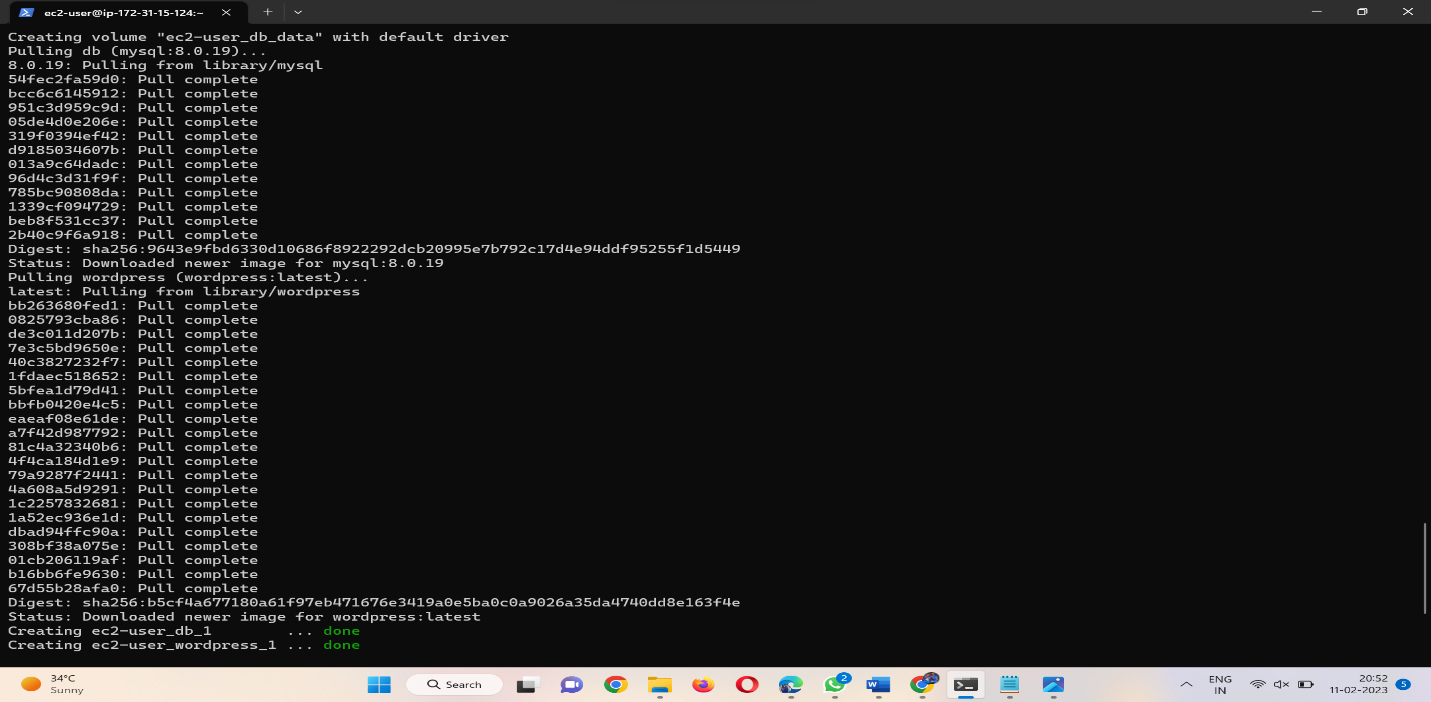
* sudo vi docker-compose.yaml file
* Here is our docker-compose yaml file:



1. Initially I had a go at running this locally on with Docker Compose

* Sudo docker-compose up -d





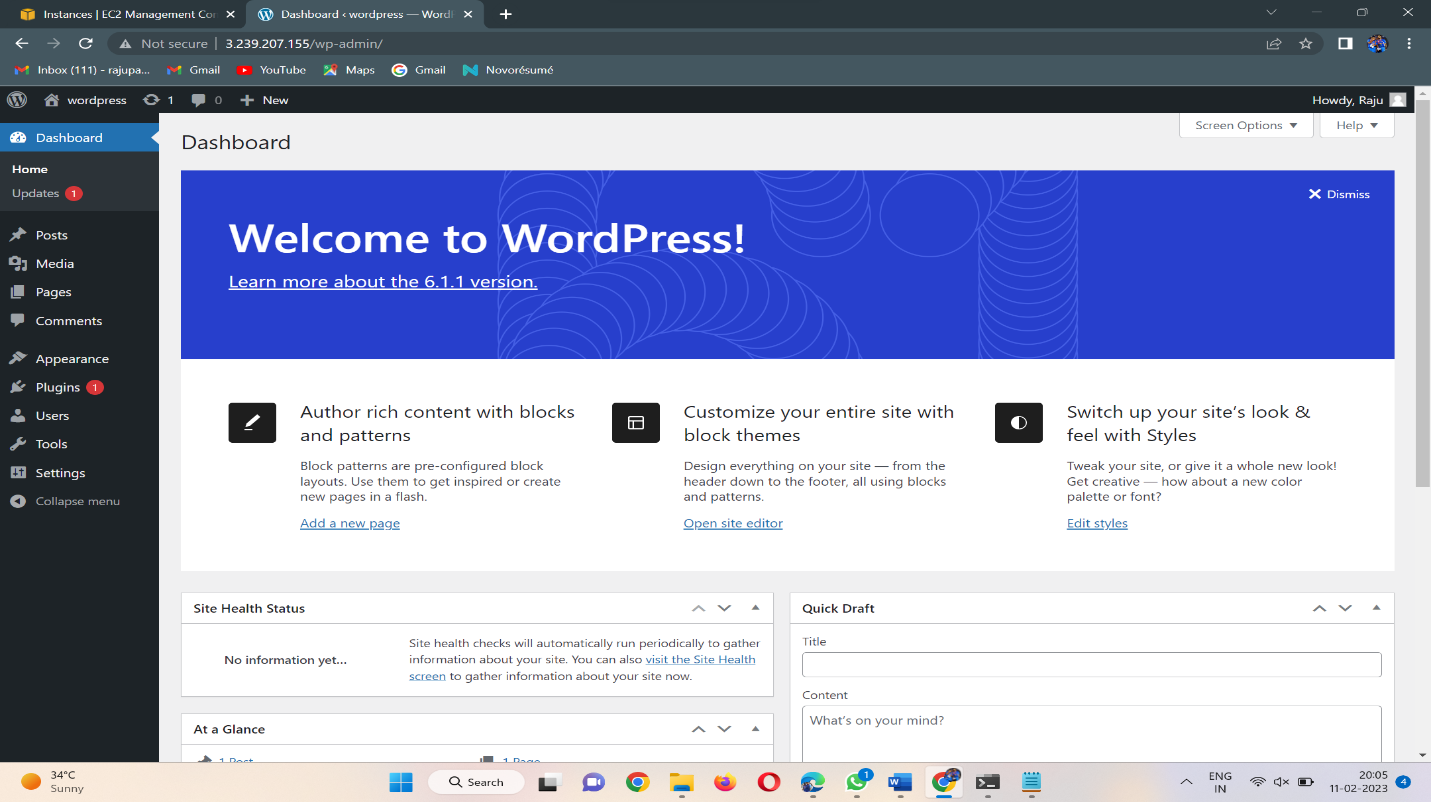
1. By using docker-compose.yaml files it was pulling images are **MySQL** and **WordPress**.

# And then lastly, I had a look to see that this was running correctly

# 

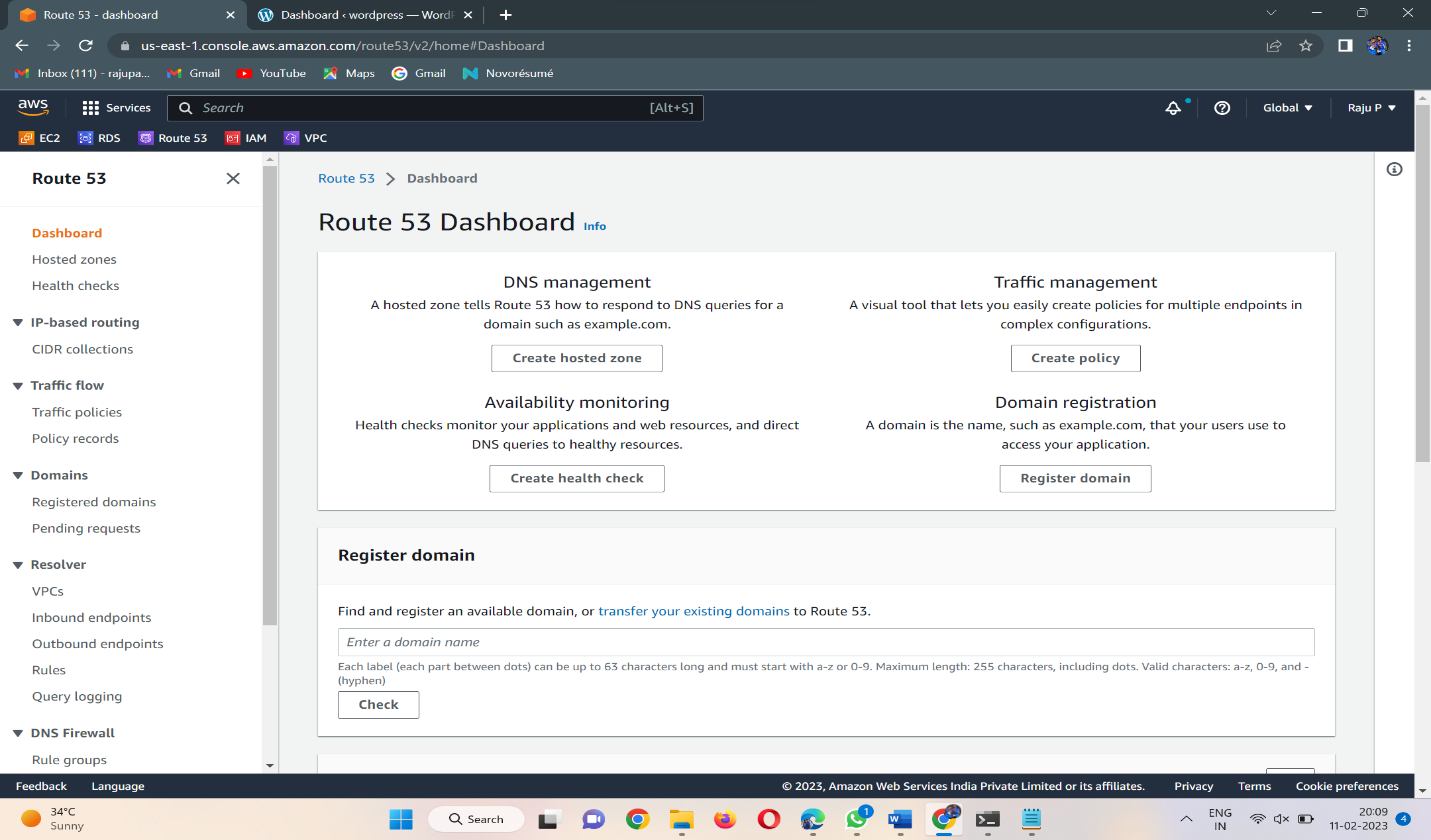
# 

1. This is the WordPress welcome page

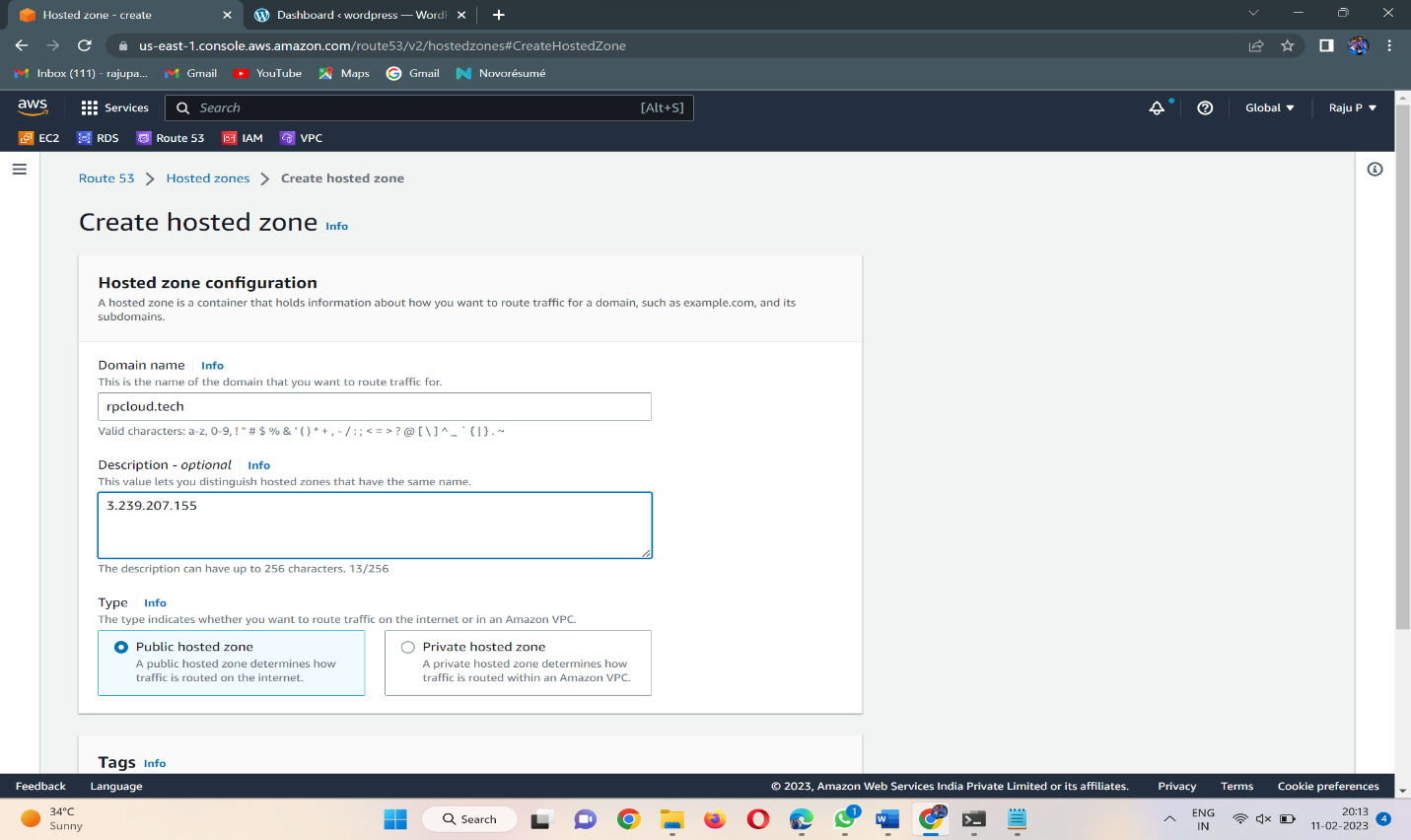


**Route53:**

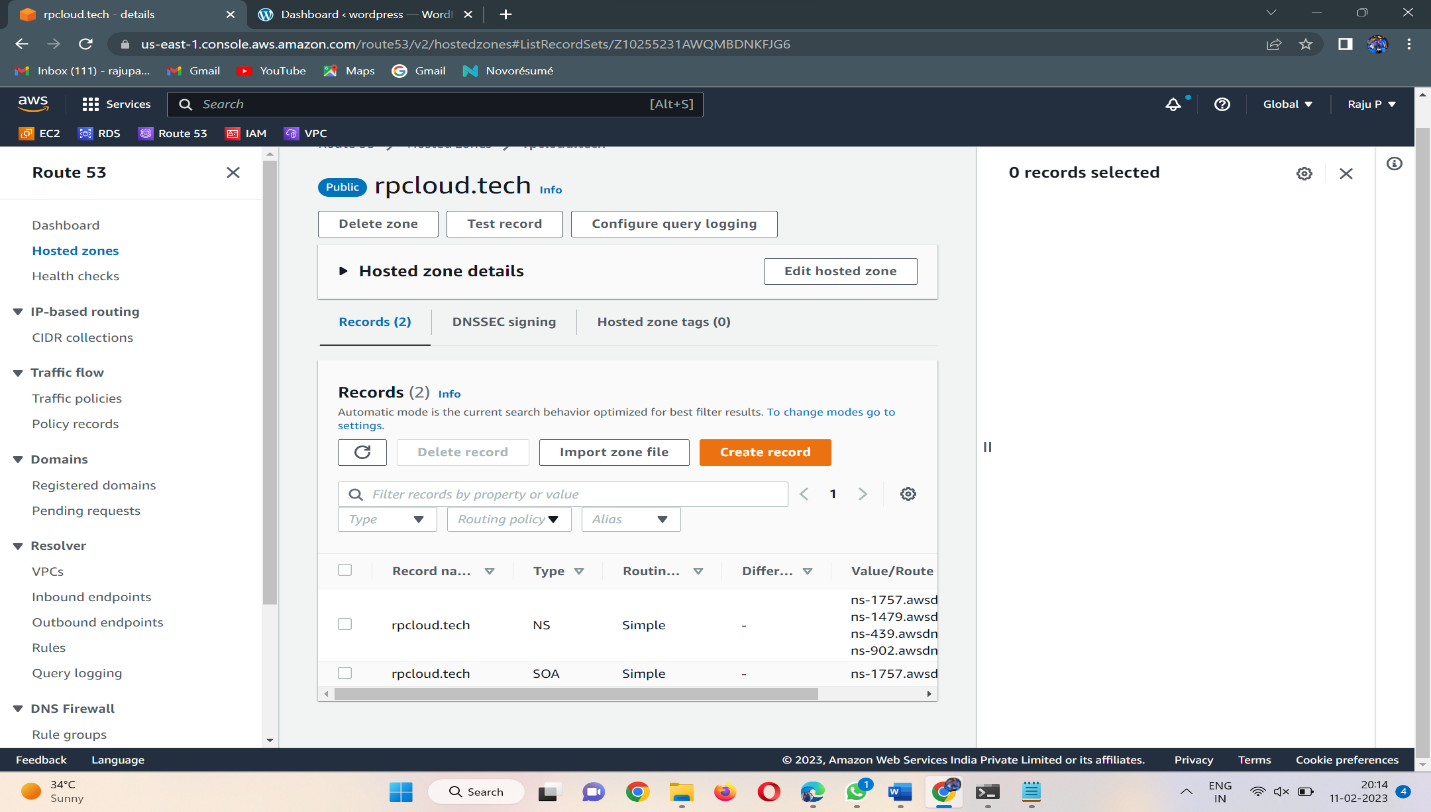
* Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service.
* Route 53 connects user requests to internet applications running on AWS or on-premises.
* Go to Route53 Dashboard, and click on DNS Management system(create hosted zone).
* Now, click on create hosted zone.



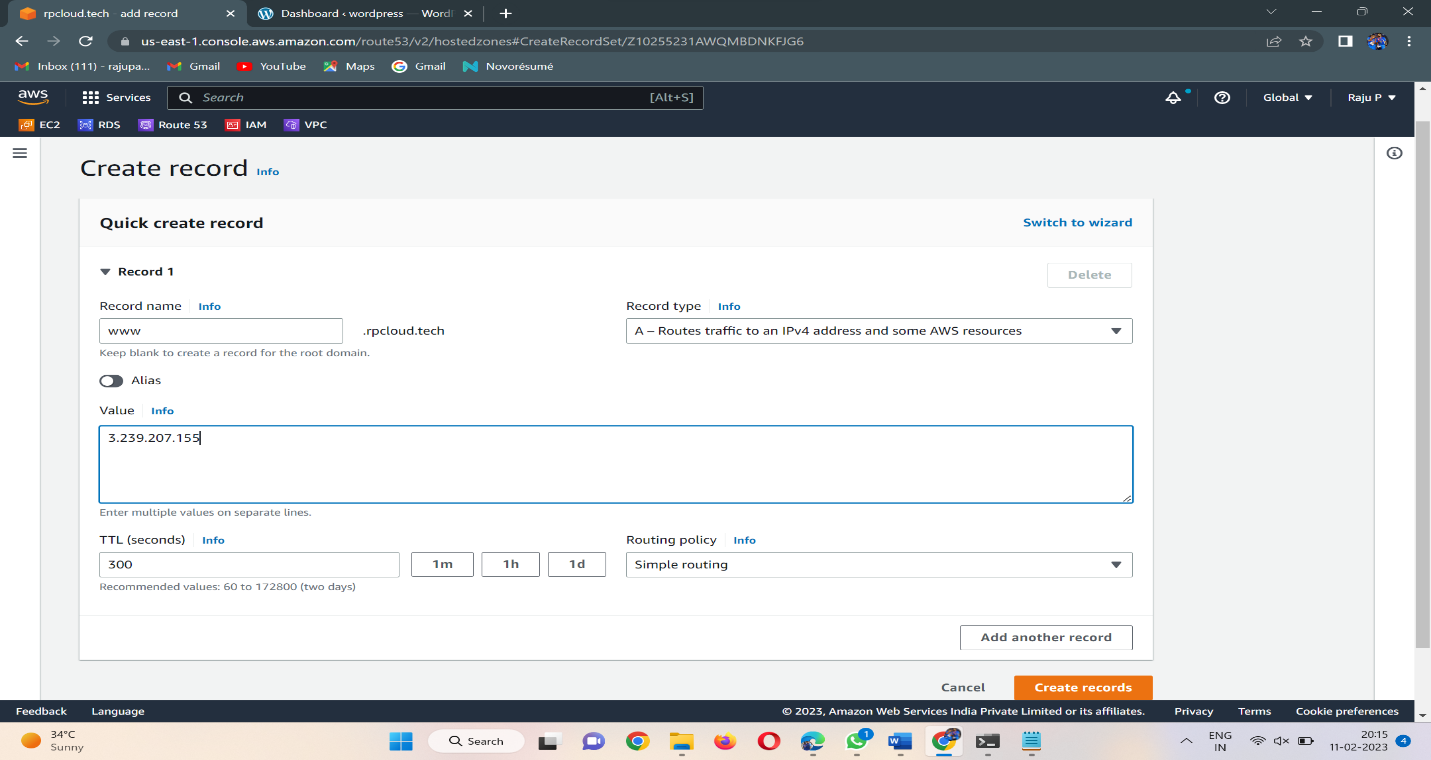
* Now fill up the hosted zone configuration details.
* Provide Domain name, Description and type should be a public hosted zone.
* Click on create Hosted Zone.
* The Hostedzone will be created.



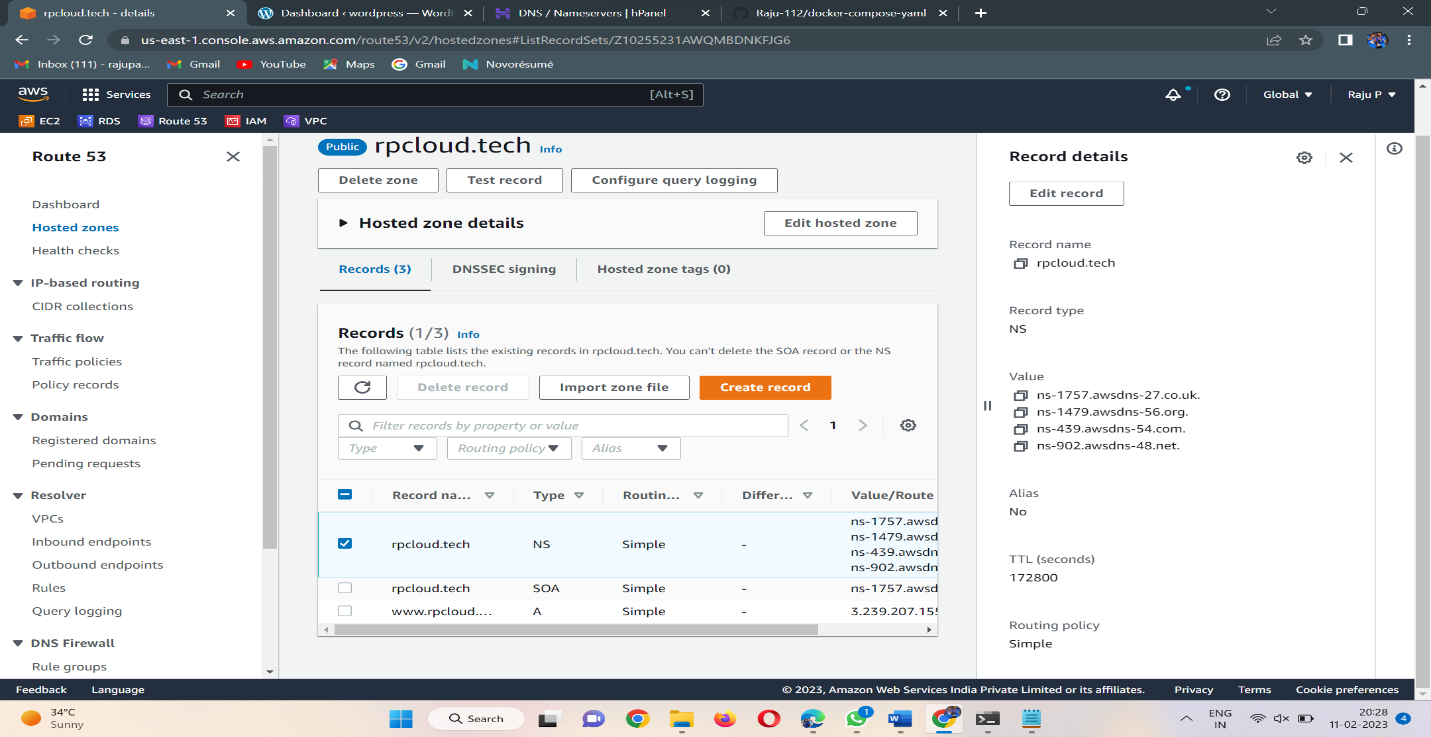
* Now create records as per requirement.
* Click on create records, In record enter the record name and record type



* In value, please enter the ip adress of the instance.



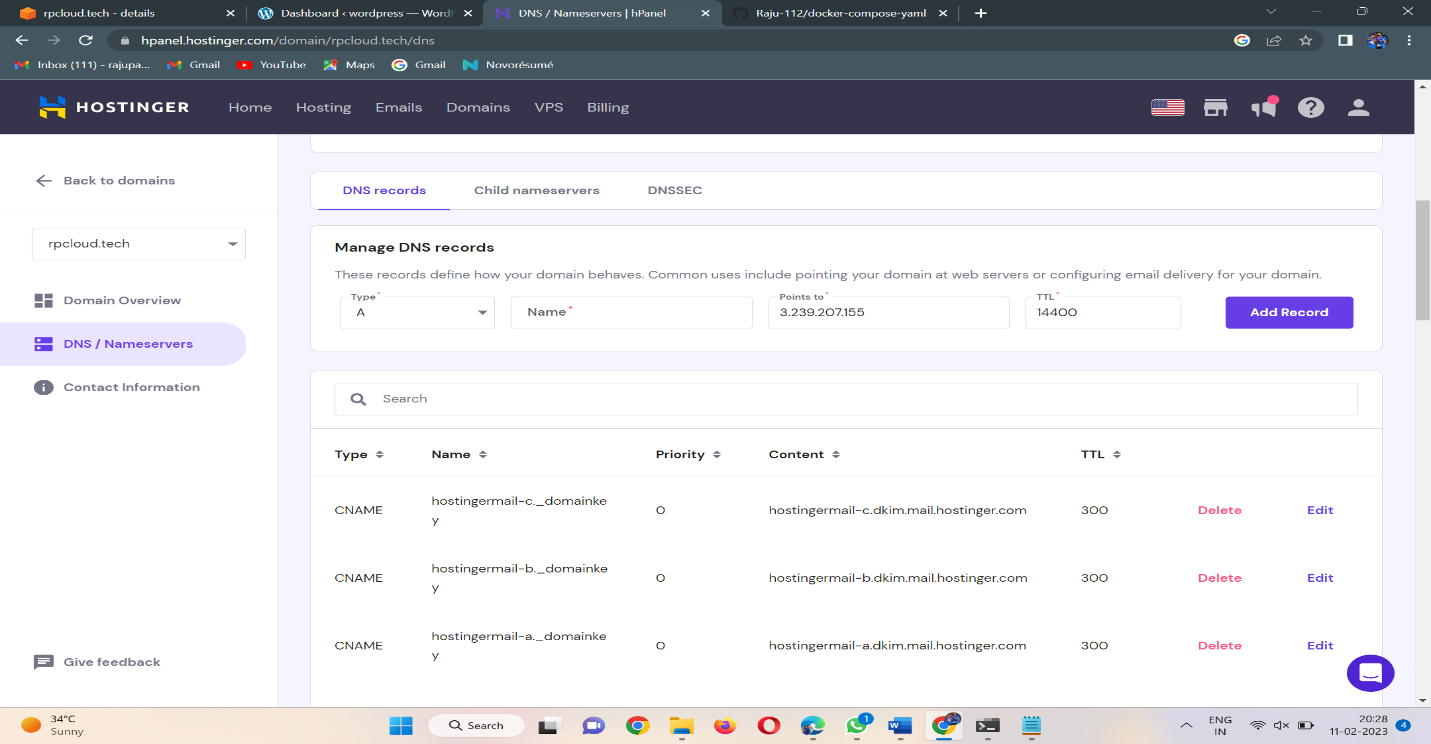
* In records, click on the Naming servers section
* Copy all the four Naming servers .



* After copying the name servers, go to Hostinger which is provided your domain name.
* Click on your domain and click on Manage servers
* Now paste the Name servers Here.



* Now click on the add record and Paste the IP address of the instance in the place of points to box.
* Now Save the details ad search it with your domain name in the browser.



* It will take some time to get ready .

