Deployment of WordPress and Mysql on AWS using Terraform

Overview:

We must create a web portal for our company with all the security as much as possible. So, we are going to use WordPress software with the dedicated database server (MySQL).

Database should not be accessible from the outside world for security purposes. We only need to public WordPress to clients.

Let us start….

Configure the AWS provider

Before using any provider, it is necessary to configure the provider with the proper credentials. Here I am using my already created profile for configuring AWS provider

```
provider "aws" {

  region = "ap-south-1"

  profile = "anuddeeph"

}
```

Step-1:

I’m creating a Infrastructure as code using terraform, which automatically create a Amazon VPC (Amazon Virtual Private Cloud). For simplicity consider it as a building of our company.

```
resource "aws_vpc" "wpvpc" {

  cidr_block = "10.7.0.0/16"

  enable_dns_hostnames = true
  tags = {
    Name = "main"
  }
}
```

Step-2:

Inside VPC, I am creating two 2 subnets. Consider subnet is like a lab in your company building.

a. Public Subnet (Accessible for public world).

b. Private Subnet (Restricted for public world).

```
resource "aws_subnet" "alpha-1a" {
  vpc_id          = "${aws_vpc.wpvpc.id}"
  availability_zone = "ap-south-1a"
  cidr_block      = "10.7.1.0/24"
  map_public_ip_on_launch = true
  tags = {
    Name = "main-1a"
  }
}
```

```
resource "aws_subnet" "alpha-1b" {

  vpc_id            = "${aws_vpc.wpvpc.id}"

  availability_zone = "ap-south-1b"

  cidr_block        = "10.7.2.0/24"

  tags = {

    Name = "main-1b"

  }

}
```

Step-3:

Now, I am creating a public facing internet gateway to connect our VPC / Network to the internet world and attaching this gateway to our VPC.

```
resource "aws_internet_gateway" "gw" {

  vpc_id = "${aws_vpc.wpvpc.id}"

  tags = {

    Name = "main-1a"

  }

}
```

Step-4:

Creating a routing table for Internet gateway, so that instance can connect to outside world. And associating this routing table with public subnet.

```
resource "aws_route_table" "rt" {

  vpc_id = "${aws_vpc.wpvpc.id}"

  route {

    cidr_block = "0.0.0.0/0"
    gateway_id = "${aws_internet_gateway.gw.id}"
  }
  tags = {
    Name = "main-1a"
  }
}
```

Creating Routing Table

```
resource "aws_route_table_association" "rta" {
  subnet_id      = aws_subnet.alpha-1a.id
  route_table_id = aws_route_table.rt.id
}
```

Step-5:

Creating security group for WordPress which will work as a firewall. This security group allows port 80 (HTTP) for clients, and port 22 (SSH) for admin team.

```
resource "aws_security_group" "allow_http_wordpress" {
  name        = "allow_http_wordpress"
  description = "Allow HTTP inbound traffic"
  vpc_id      = "${aws_vpc.wpvpc.id}"

  ingress {
    description = "Http from VPC"
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = [ "0.0.0.0/0" ]
  }
  ingress {
    description = "SSH from VPC"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [ "0.0.0.0/0" ]
  }
  ingress {
    description = "HTTPS"
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = [ "0.0.0.0/0" ]
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "wpsgroup"
  }
}
```

Step-6:

Creating security group for MySQL instance, this security group allows only port 3306 so that our WordPress VM can connect with the same.

```
resource "aws_security_group" "mysql-sg" {
 name        = "for-mysql"
 description = "MYSQL-setup"
 vpc_id      = "${aws_vpc.wpvpc.id}"


 ingress {
  description = "MYSQL from VPC"
  from_port   = 3306
  to_port     = 3306
  protocol    = "tcp"
  cidr_blocks = [ "0.0.0.0/0" ]
 }
 ingress {
  description = "SSH from VPC"
  from_port   = 22
  to_port     = 22
  protocol    = "tcp"
  cidr_blocks = [ "0.0.0.0/0" ]
 }
 egress {
  from_port   = 0
  to_port     = 0
```

```
   protocol    = "-1"
   cidr_blocks = ["0.0.0.0/0"]
 }

 depends_on = [
 aws_security_group.allow_http_wordpress,
 ]

 tags = {
   Name = "mysqlsgroup"
 }
}
```

Step-7:

Launching an EC2 instance which has MySQL setup already with the security group which I have created for MySQL. And attaching the key to the instance for further login into it.

This MySQL instance is part private subnet so that outside world cannot connect to it.

```
variable "enter_ur_key_name" {
                          type = string
             default = "taskkey"
}
resource "aws_instance" "mysql" {
 ami          = "ami-76166b19"
```

```
  instance_type = "t2.micro"
  key_name      = var.enter_ur_key_name
  availability_zone = "ap-south-1b"
  subnet_id     = "${aws_subnet.alpha-1b.id}"
  security_groups = [ "${aws_security_group.mysql-sg.id}" ]
  tags = {
    Name = "MYSQL"
  }
}
```

Step-8:

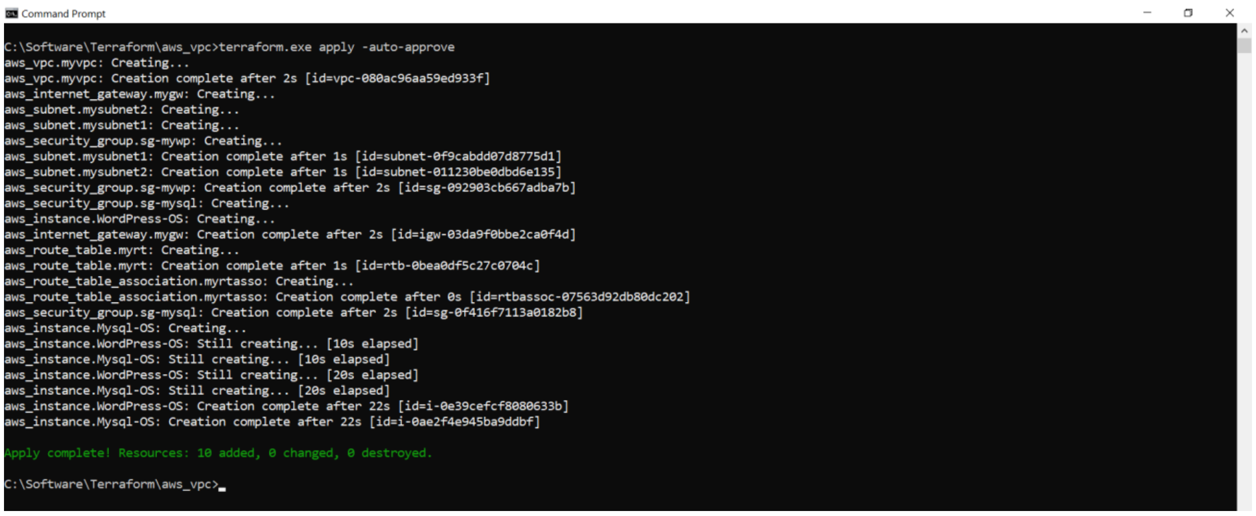
Launching an EC2 instance which has WordPress setup already having the security group which I have created for WordPress. And attaching the key to the instance for further login into it. This instance is a part of public subnet so that our clients can connect our site.

```
variable "enter_ur_key_name" {
                    type = string
          default = "taskkey"
}
resource "aws_instance" "wordpress" {
  ami          = "ami-0979674e4a8c6ea0c"
  instance_type = "t2.micro"
  key_name      = var.enter_ur_key_name
  availability_zone = "ap-south-1a"
  subnet_id     = "${aws_subnet.alpha-1a.id}"
  security_groups = [ "${aws_security_group.allow_http_wordpress.id}" ]
  tags = {
    Name = "Wordpress"
  }
}
```

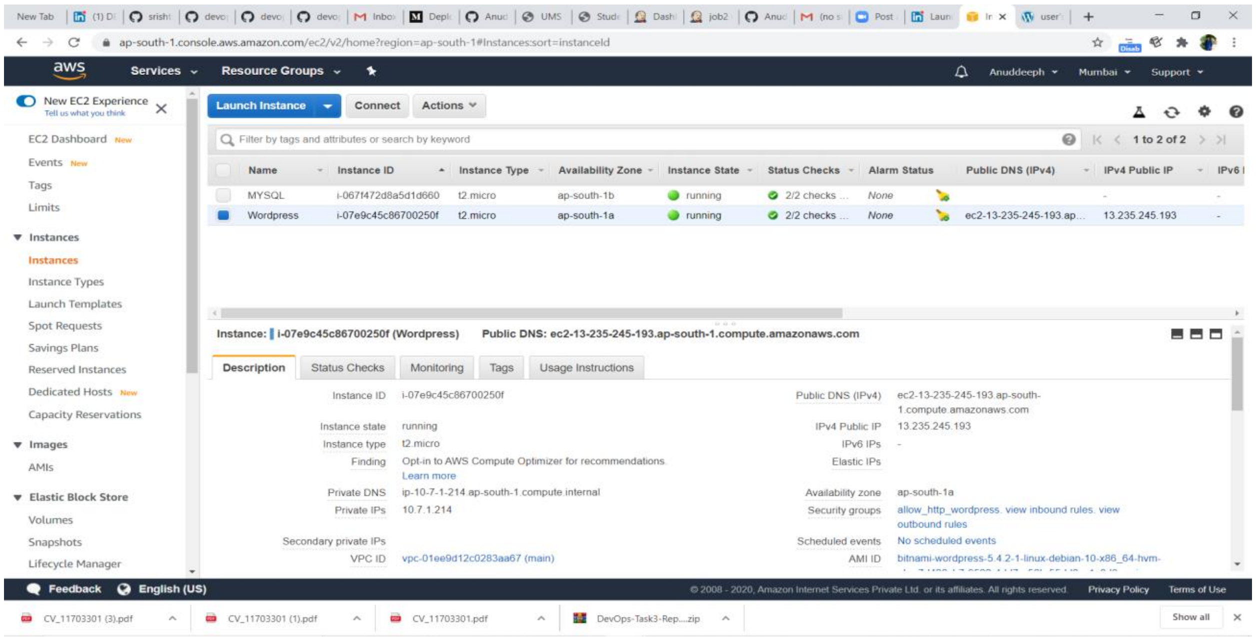
Finally, we are done with the coding part. Now it's time to execute this file.

Execution of Terraform file:

terraform apply –auto-approve



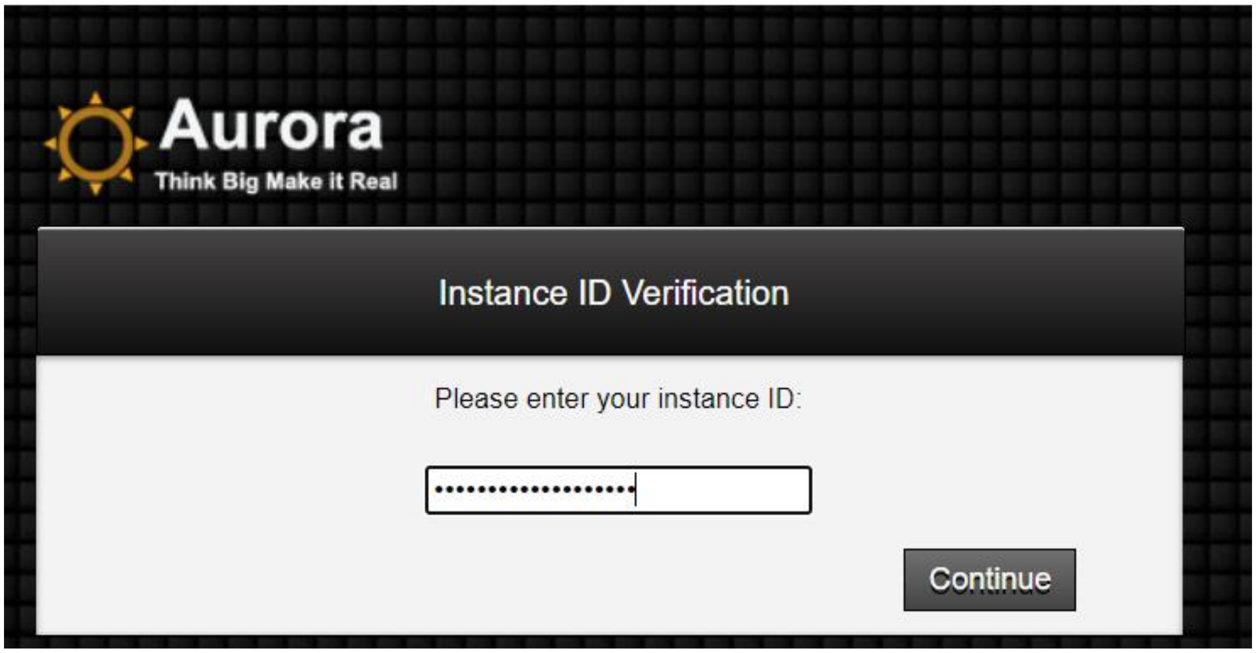
```
C:\Software\Terraform\aws_vpc>terraform.exe apply -auto-approve
aws_vpc.myvpc: Creating...
aws_vpc.myvpc: Creation complete after 2s [id=vpc-080ac96aa59ed933f]
aws_internet_gateway.mygw: Creating...
aws_subnet.mysubnet2: Creating...
aws_subnet.mysubnet1: Creating...
aws_security_group.sg-mywp: Creating...
aws_subnet.mysubnet1: Creation complete after 1s [id=subnet-0f9cabdd07d8775d1]
aws_subnet.mysubnet2: Creation complete after 1s [id=subnet-011230be0dbd6e135]
aws_security_group.sg-mywp: Creation complete after 2s [id=sg-092903cb667adba7b]
aws_security_group.sg-mysql: Creating...
aws_instance.WordPress-OS: Creating...
aws_internet_gateway.mygw: Creation complete after 2s [id=igw-03da9f0bbe2ca0f4d]
aws_route_table.myrt: Creating...
aws_route_table.myrt: Creation complete after 1s [id=rtb-0bea0df5c27c0704c]
aws_route_table_association.myrtasso: Creating...
aws_route_table_association.myrtasso: Creation complete after 0s [id=rtbassoc-07563d92db80dc202]
aws_security_group.sg-mysql: Creation complete after 2s [id=sg-0f416f7113a0182b8]
aws_instance.Mysql-OS: Creating...
aws_instance.WordPress-OS: Still creating... [10s elapsed]
aws_instance.Mysql-OS: Still creating... [10s elapsed]
aws_instance.WordPress-OS: Still creating... [20s elapsed]
aws_instance.Mysql-OS: Still creating... [20s elapsed]
aws_instance.WordPress-OS: Creation complete after 22s [id=i-0e39cefcf8080633b]
aws_instance.Mysql-OS: Creation complete after 22s [id=i-0ae2f4e945ba9ddbf]

Apply complete! Resources: 10 added, 0 changed, 0 destroyed.

C:\Software\Terraform\aws_vpc>
```


ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#Instances:sort=instanceId

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 I
MYSQL	i-067f472d8a5d1d660	t2.micro	ap-south-1b	running	2/2 checks ...	None		-	-
Wordpress	i-07e9c45c86700250f	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-13-235-245-193.ap...	13.235.245.193	-

Instance: i-07e9c45c86700250f (Wordpress) Public DNS: ec2-13-235-245-193.ap-south-1.compute.amazonaws.com

Instance ID	i-07e9c45c86700250f	Public DNS (IPv4)	ec2-13-235-245-193.ap-south-1.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.235.245.193
Instance type	t2.micro	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more	Elastic IPs	
Private DNS	ip-10-7-1-214.ap-south-1.compute.internal	Availability zone	ap-south-1a
Private IPs	10.7.1.214	Security groups	allow_http_wordpress. view inbound rules. view outbound rules
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-01ee9d12c0283aa67 (main)	AMI ID	bitnami-wordpress-5.4.2-1-linux-debian-10-x86_64-hvm-



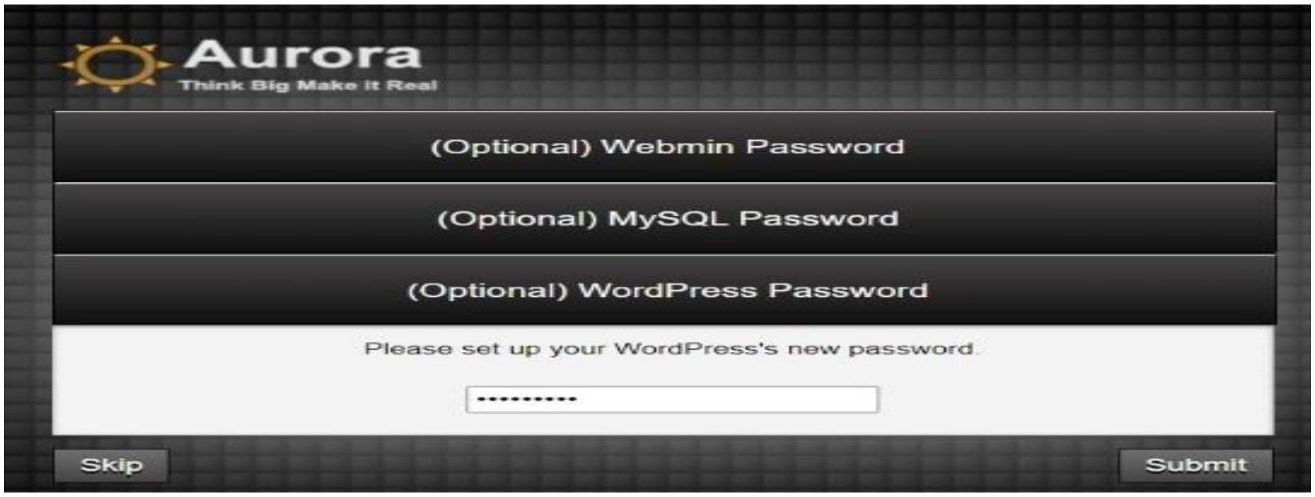
Instance ID Verification

Please enter your instance ID:

•••••••••••••••••••

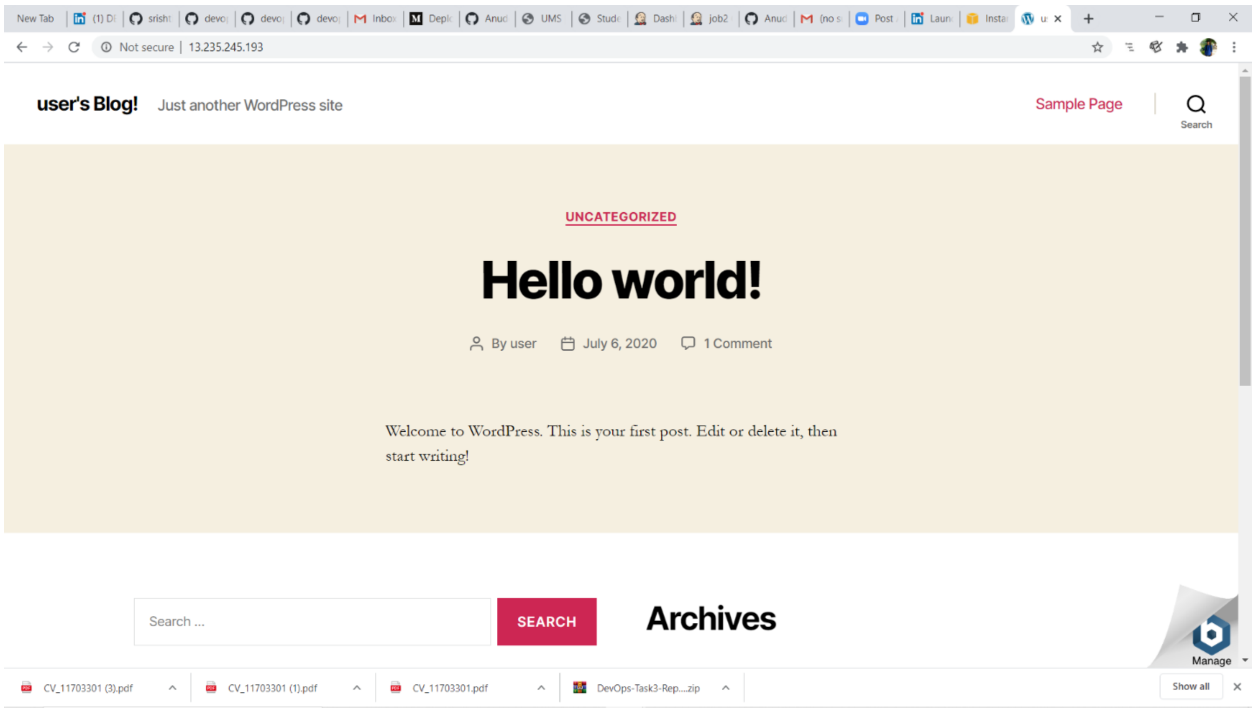
Continue

Using Public IP of WordPress instance site is accessible


Aurora
Think Big Make It Real
(Optional) Webmin Password
(Optional) MySQL Password
(Optional) WordPress Password
Skip
Submit

Please set up your WordPress's new password.

Set Passwords



user's Blog! Just another WordPress site

UNCATEGORIZED

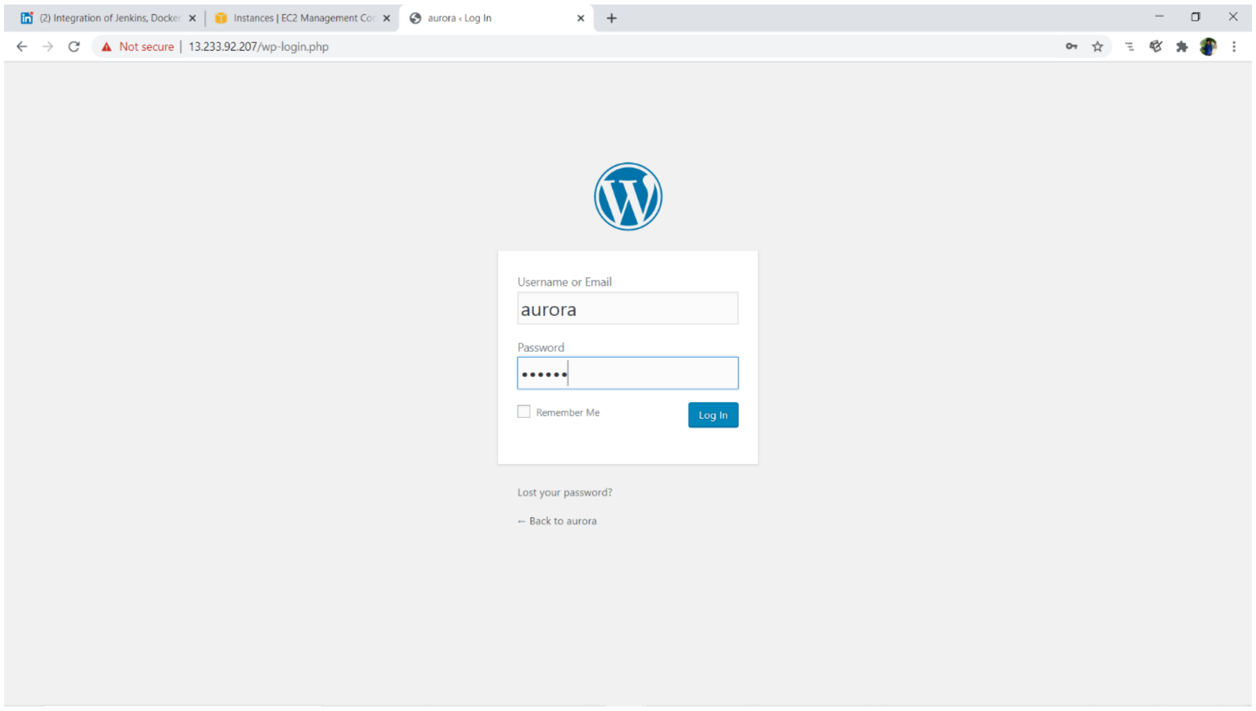
Hello world!

By user July 6, 2020 1 Comment

Welcome to WordPress. This is your first post. Edit or delete it, then start writing!

Archives

WordPress is working properly


(2) Integration of Jenkins, Docker
Instances | EC2 Management Con
aurora ‹ Log In
Not secure | 13.233.92.207/wp-login.php

Username or Email

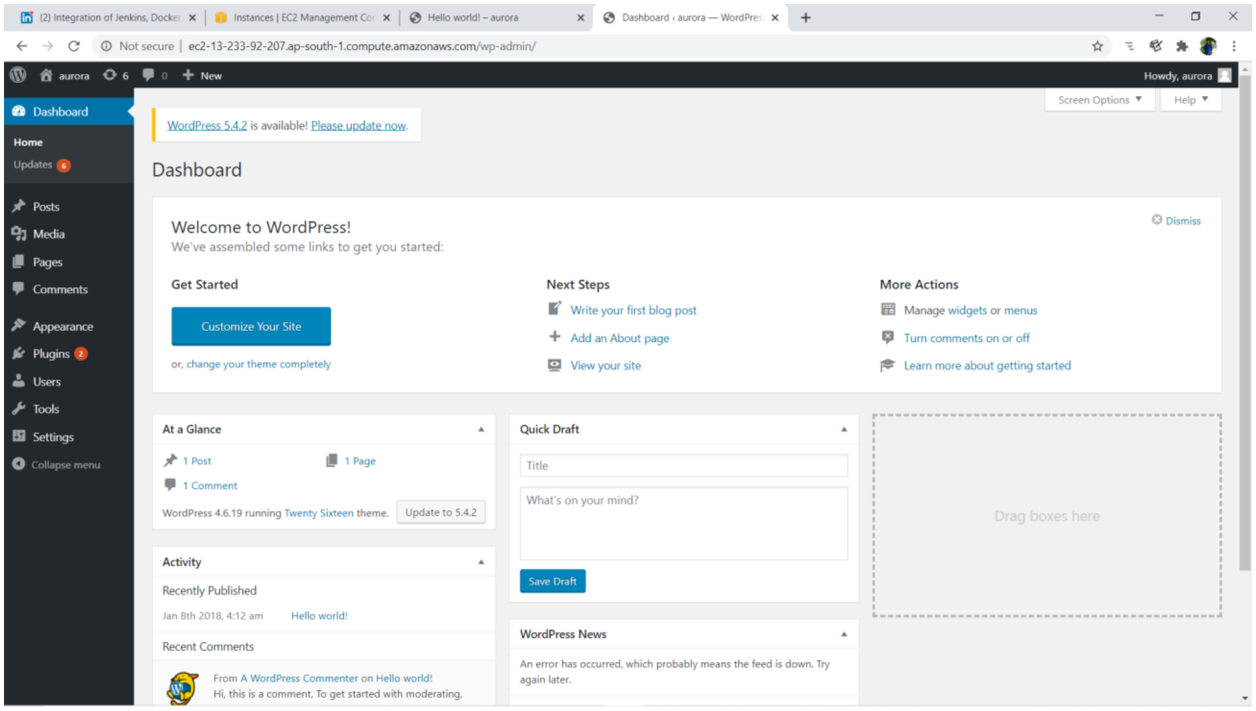
aurora

Password

••••••

Remember Me
Log In
Lost your password?
← Back to aurora

By default initial username is “aurora”


(2) Integration of Jenkins, Docker
Instances | EC2 Management Con
Hello world! – aurora
Dashboard ‹ aurora — WordPress
Not secure | ec2-13-233-92-207.ap-south-1.compute.amazonaws.com/wp-admin/
aurora
6
0
New
Howdy, aurora
Screen Options
Help
Dashboard
Home
Updates 6
Posts
Media
Pages
Comments
Appearance
Plugins 2
Users
Tools
Settings
Collapse menu

WordPress 5.4.2 is available! Please update now.

Dashboard

Welcome to WordPress!
We've assembled some links to get you started:

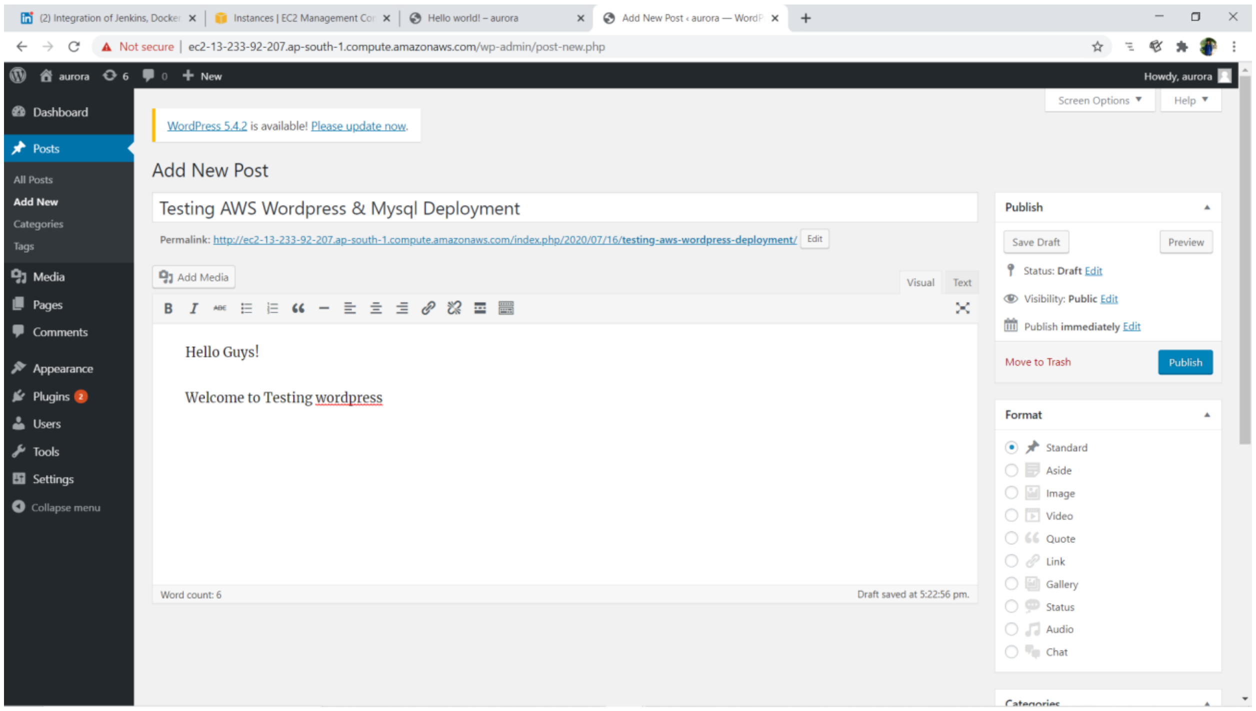
WordPress 4.6.19 running Twenty Sixteen theme.

Jan 8th 2018, 4:12 am Hello world!

From A WordPress Commenter on Hello world!
Hi, this is a comment. To get started with moderating,

An error has occurred, which probably means the feed is down. Try again later.

Dashboard of an account



(2) Integration of Jenkins, Docke
Instances | EC2 Management Co
Hello world! – aurora
Add New Post ‹ aurora — WordP
Not secure | ec2-13-233-92-207.ap-south-1.compute.amazonaws.com/wp-admin/post-new.php
aurora
6
0
New
Howdy, aurora
Dashboard
Posts
All Posts
Add New
Categories
Tags
Media
Pages
Comments
Appearance
Plugins 2
Users
Tools
Settings
Collapse menu
Screen Options
Help

WordPress 5.4.2 is available! Please update now.

Add New Post

Testing AWS Wordpress & Mysql Deployment

Permalink: http://ec2-13-233-92-207.ap-south-1.compute.amazonaws.com/index.php/2020/07/16/testing-aws-wordpress-deployment/ Edit

Hello Guys!

Welcome to Testing wordpress

Word count: 6

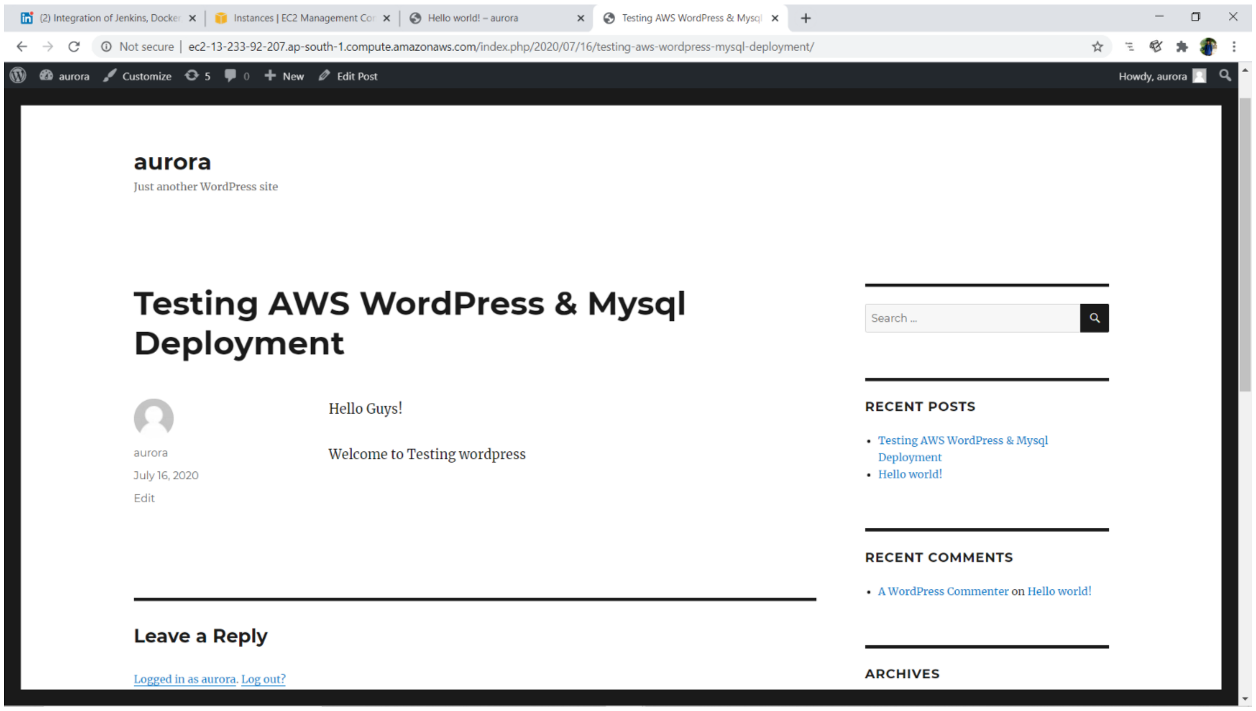
Draft saved at 5:22:56 pm.

Status: Draft Edit

Visibility: Public Edit

Publish immediately Edit

Move to Trash
Publish
Format
Standard
Aside
Image
Video
Quote
Link
Gallery
Status
Audio
Chat
Categories



(2) Integration of Jenkins, Docke
Instances | EC2 Management Co
Hello world! – aurora
Testing AWS WordPress & Mysql
Not secure | ec2-13-233-92-207.ap-south-1.compute.amazonaws.com/index.php/2020/07/16/testing-aws-wordpress-mysql-deployment/
aurora
Customize
5
0
New
Edit Post
Howdy, aurora

aurora

Just another WordPress site

Testing AWS WordPress & Mysql Deployment

aurora
July 16, 2020
Edit

Hello Guys!

Welcome to Testing wordpress

Leave a Reply

Logged in as aurora. Log out?

RECENT POSTS

- Testing AWS WordPress & Mysql Deployment
- Hello world!

RECENT COMMENTS

- A WordPress Commenter on Hello world!

ARCHIVES

Successfully deployed WordPress and Mysql on AWS using Terraform…!!!

Thanks for reading till the end…!!!