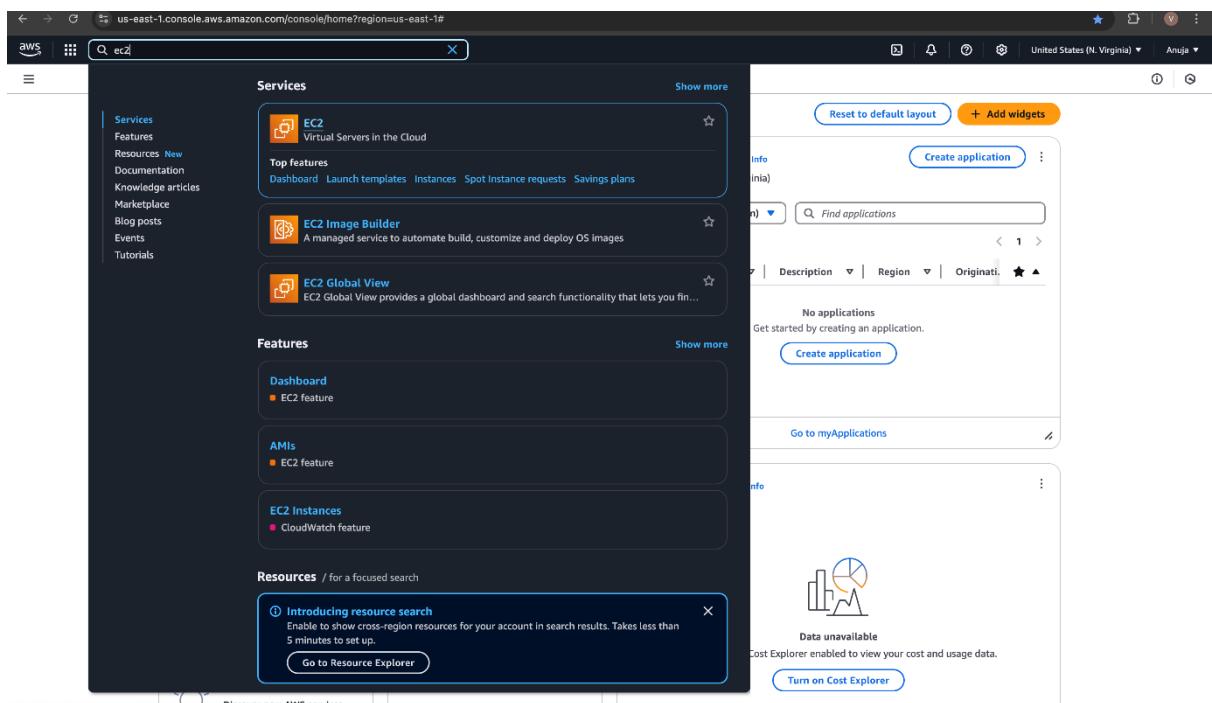


Word Press Website Creation

Using EC2 Creating WordPress:

First we need to create one Ec2 Linux instance using AWS console.

Go to AWS and search for EC2.



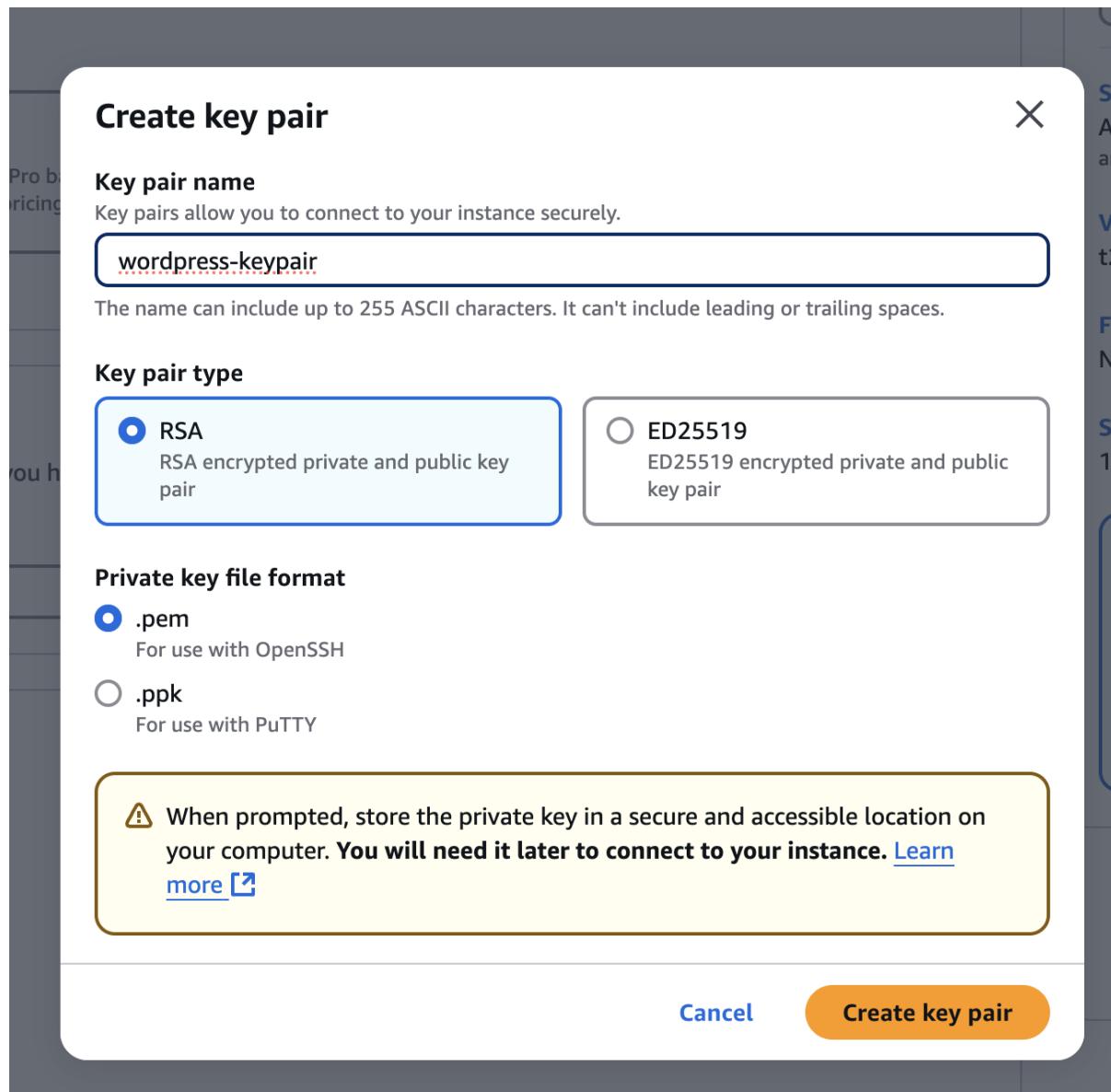
And then press on launch instance.

Give name of your server as **wordpress-server**

And give application and OS images here I am selecting Image as **Amazon Linux 2 AMI 64 bit.**

Select instance type as **T2. micro** and in keypair section press on create new keypair.

Give keypair name as **wordpress-keypair** keypair type as **RSA** and private key file format as **.pem** to work with open ssh. And press create keypair.

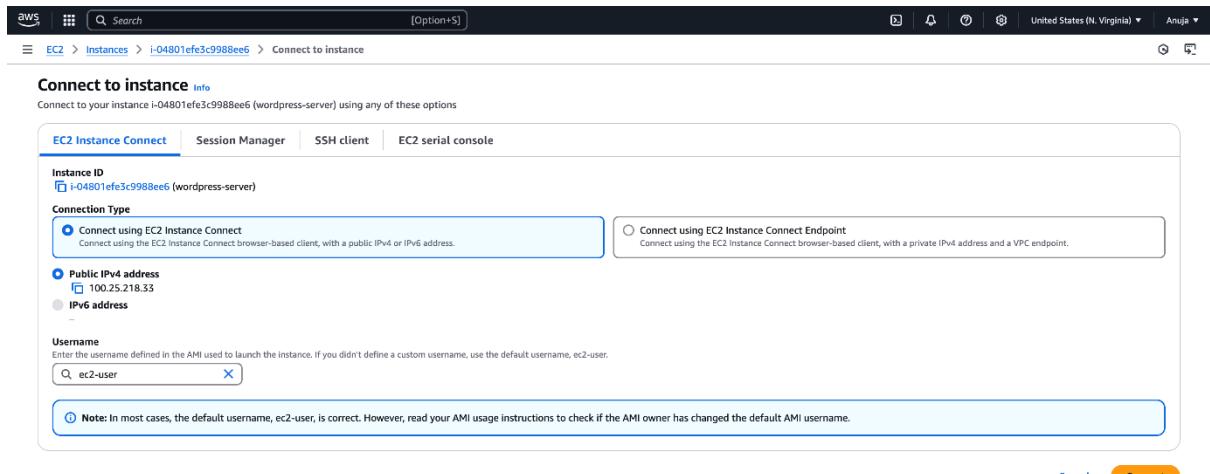


In network setting give security group as create security group and allow **ssh**, **http** and **https** traffic. And press on launch instance.

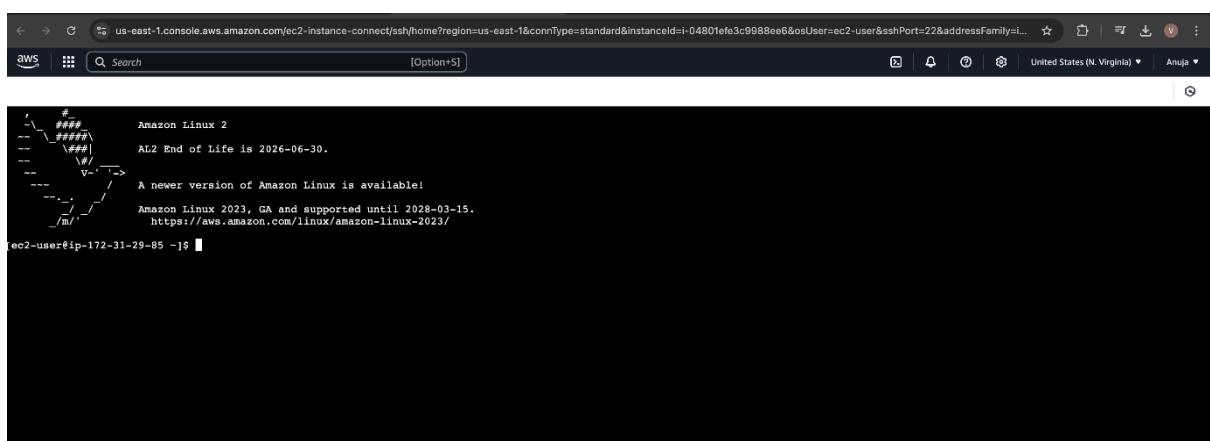
Your instance will launch and it will take some time to get running.

And press the instance and click on connect button.

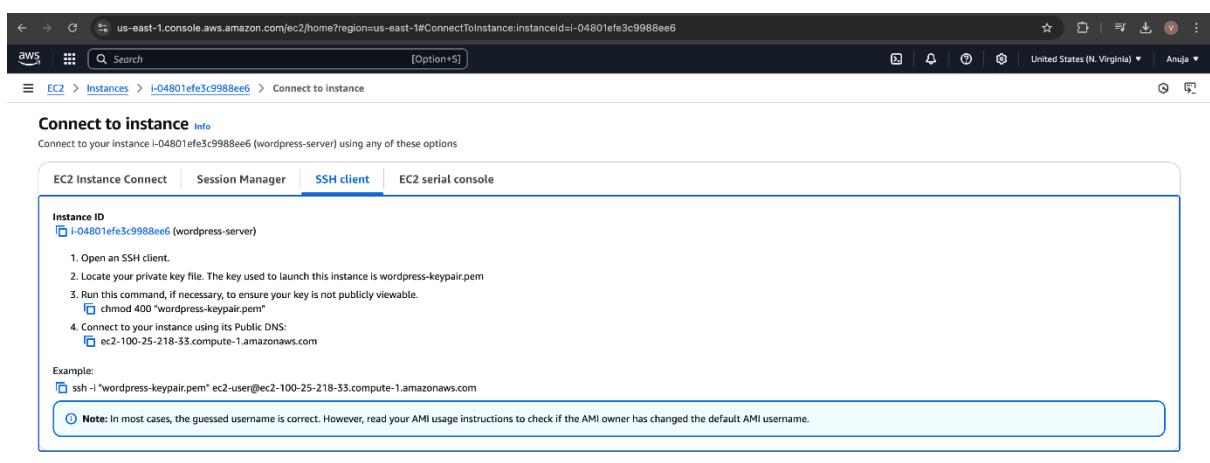
And then connect to Ec2 instance connect it will look like this.



And then press on connect then we will get this prompt.



Another way to create EC2 instance is using SSH client.



Then press on connect then you can have another way to create your instance.

```
Last login: Wed Feb 26 11:44:09 on console
raj@Rajs-Laptop ~ % cd desktop
raj@Rajs-Laptop desktop % chmod 400 "wordpress-keypair.pem"
raj@Rajs-Laptop desktop % █
```

After login in you can change the permission of your keypair using this command.

```
chmod 400 "wordpress-keypair.pem"
```

Then connect your instance using this command.

```
ssh -i "wordpress-keypair.pem" ec2-user@ec2-100-25-218-33.compute-1.amazonaws.com
```

You can get this command on your AWS EC2 instance connect.

```
Last login: Wed Feb 26 11:44:09 on console
raj@Rajs-Laptop ~ % cd desktop
raj@Rajs-Laptop desktop % chmod 400 "wordpress-keypair.pem"
raj@Rajs-Laptop desktop % ssh -i "wordpress-keypair.pem" ec2-user@ec2-100-25-218-33.compute-1.amazonaws.com
The authenticity of host 'ec2-100-25-218-33.compute-1.amazonaws.com (100.25.218.33)' can't be established.
ED25519 key fingerprint is SHA256:Sb0RkbRW2hEk54WQsLyQwriDwJs/4txm+PTKj/+Fuw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-100-25-218-33.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Last login: Wed Feb 26 06:22:56 2025 from ec2-18-206-107-28.compute-1.amazonaws.com
'          #
~\_\_ #####_      Amazon Linux 2
~~ \_\_#####\_
~~      \###|      AL2 End of Life is 2026-06-30.
~~      \#/  ---_
~~      V~' '-'>
~~      /      A newer version of Amazon Linux is available!
~~ .- .- /      Amazon Linux 2023, GA and supported until 2028-03-15.
_/_m/`      https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-29-85 ~]$ █
```

Till now we have EC2 machine ready with us now we have to install Apache webserver in this EC2 Linux machine

First we have to update the yum package using `sudo yum update -y` .

```
[ec2-user@ip-172-31-29-85 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
No packages marked for update
[ec2-user@ip-172-31-29-85 ~]$ | 3.6 kB 00:00:00
```

After that start installing Apache server in your ec2 machine. Using the following command.

```
sudo yum install -y httpd
```

```
[ec2-user@ip-172-31-29-85 ~]$ sudo yum install -y httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
--> Package httpd.x86_64 0:2.4.62-1.amzn2.0.2 will be installed
--> Processing Dependency: httpd-filesystem = 2.4.62-1.amzn2.0.2 for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: httpd-tools = 2.4.62-1.amzn2.0.2 for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: system-logos-httdp for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.62-1.amzn2.0.2.x86_64
--> Running transaction check
--> Package apr.x86_64 0:1.7.2-1.amzn2.0.1 will be installed
--> Package apr-util.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.3-1.amzn2.0.1 for package: apr-util-1.6.3-1.amzn2.0.1.x86_64
--> Package generic-logos-httdp.noarch 0:18.0.0-4.amzn2 will be installed
--> Package httpd-filesystem.noarch 0:2.4.62-1.amzn2.0.2 will be installed
--> Package httpd-tools.x86_64 0:2.4.62-1.amzn2.0.2 will be installed
--> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
--> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.2 will be installed
--> Running transaction check
--> Package apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package           Arch    Version        Repository      Size
=====
Installing:
httpd           x86_64  2.4.62-1.amzn2.0.2      amzn2-core   1.4 M
Installing for dependencies:
apr              x86_64  1.7.2-1.amzn2.0.1      amzn2-core   130 k
apr-util         x86_64  1.6.3-1.amzn2.0.1      amzn2-core   101 k
apr-util-bdb    x86_64  1.6.3-1.amzn2.0.1      amzn2-core   22 k
generic-logos-httdp noarch 18.0.0-4.amzn2      amzn2-core   19 k
httpd-filesystem noarch 2.4.62-1.amzn2.0.2      amzn2-core   25 k
httpd-tools      x86_64  2.4.62-1.amzn2.0.2      amzn2-core   89 k
mailcap          noarch 2.1.41-2.amzn2        amzn2-core   31 k
mod_http2        x86_64  1.15.19-1.amzn2.0.2    amzn2-core   149 k
```

```
Installed:
httpd.x86_64 0:2.4.62-1.amzn2.0.2

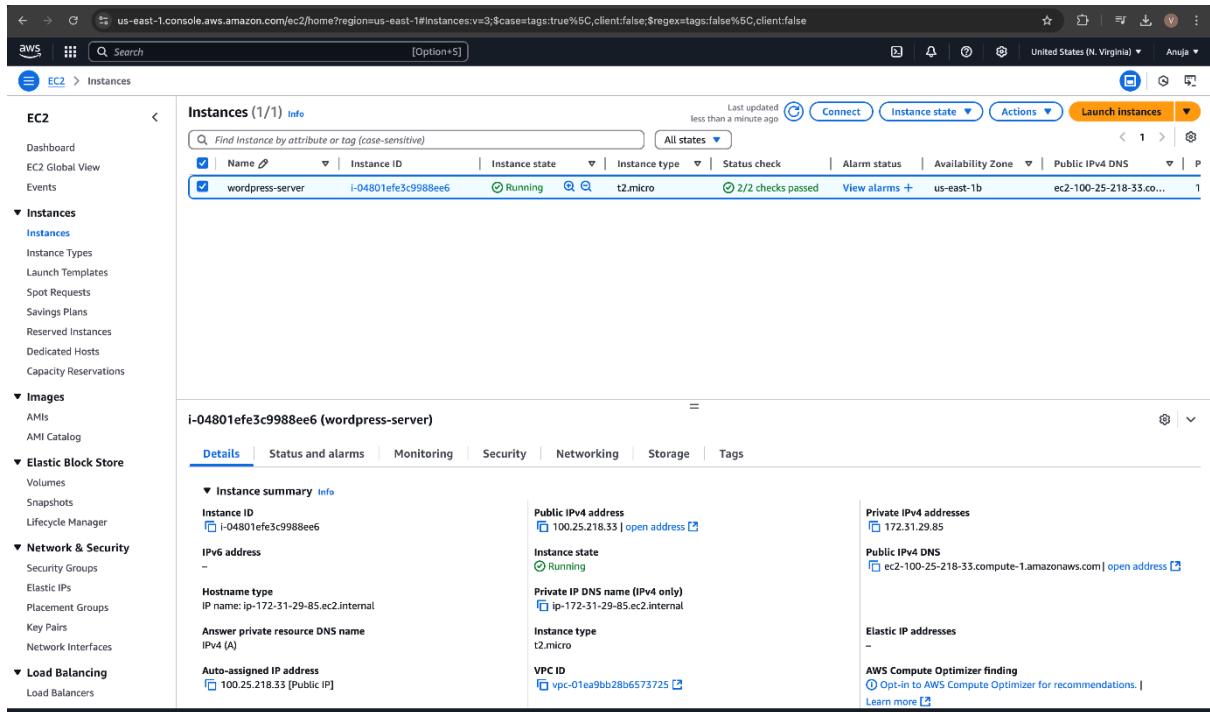
Dependency Installed:
apr.x86_64 0:1.7.2-1.amzn2.0.1  apr-util.x86_64 0:1.6.3-1.amzn2.0.1  apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1 generic-logos-httdp.noarch 0:18.0.0-4.amzn2  httpd-filesystem.noarch 0:2.4.62-1.amzn2.0.2  httpd-tools.x86_64 0:2.4.62-1.amzn2.0.2  mailcap.noarch 0:2.1.41-2.amzn2  mod_http2.x86_64 0:1.15.19-1.amzn2.0.2

Complete!
[ec2-user@ip-172-31-29-85 ~]$
```

After that start the service using this command **sudo service httpd start**

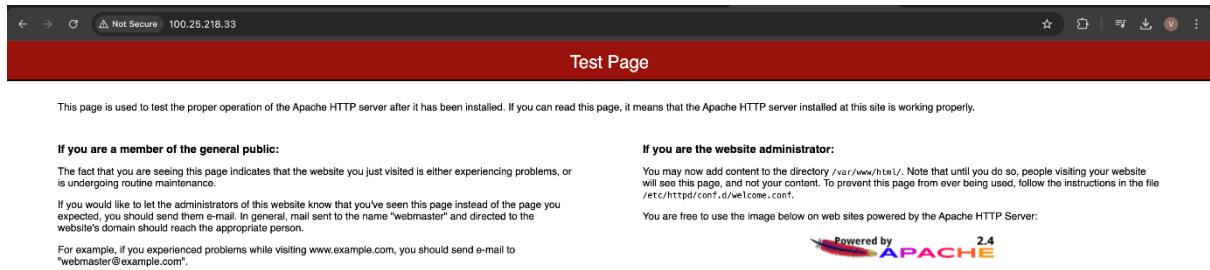
```
[ec2-user@ip-172-31-29-85 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-172-31-29-85 ~]$
```

Then open your EC2 at down you will find public IP copy that Ip.



The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various EC2-related options like Dashboard, Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area shows a table of instances. One instance is selected, named 'wordpress-server' with the ID 'i-04801efe3c998ee6'. The instance is 'Running' on a 't2.micro' type in the 'us-east-1b' availability zone. It has a Public IPv4 DNS of 'ec2-100-25-218-33.compute-1.amazonaws.com'. The 'Details' tab is selected, showing more instance details such as Public IPv4 address (100.25.218.33), Instance state (Running), and VPC ID (vpc-01ea9bb28b6573725).

Open that public IP in a browser. If you get Apache test page then your Apache server is ready.



The screenshot shows a browser window with the URL '100.25.218.33'. The page title is 'Test Page'. The content of the page is the Apache Test Page, which includes a message about the proper operation of the Apache HTTP server, instructions for website administrators, and a 'Powered by APACHE 2.4' logo.

And next step is to install php in the ec2 server. Using this command.

```
sudo amazon-linux-extras install php8.2 -y
```

```
[ec2-user@ip-172-31-29-85 ~]$ sudo amazon-linux-extras install php8.2 -y
```

After installation it will show as installation complete.

```
Installed:
  php-cli.x86_64 0:8.2.27-1.amzn2.0.5      php-common.x86_64 0:8.2.27-1.amzn2.0.5  php-fpm.x86_64 0:8.2.27-1.amzn2.0.5
  php-mysqlnd.x86_64 0:8.2.27-1.amzn2.0.5  php-pdo.x86_64 0:8.2.27-1.amzn2.0.5

Dependency Installed:
  libzip.x86_64 0:1.3.2-1.amzn2.0.1

Complete!
  2  httpd_modules           available  [ =1.0  =stable ]
  3  memcached1.5           available  \
    [ =1.5.1  =1.5.16  =1.5.17 ]
  9  R3.4                   available  [ =3.4.3  =stable ]
 10  rust1                  available  \
    [ =1.22.1  =1.26.0  =1.26.1  =1.27.2  =1.31.0  =1.38.0
    =stable ]
 18  libreoffice            available  \
    [ =5.0.6.2_15  =5.3.6.1  =stable ]
 19  gimp                   available  [ =2.8.22 ]
 20  tdocker=latest         enabled    \
```

You can see the installation status using the `php -v` command.

```
[ec2-user@ip-172-31-29-85 ~]$ php -v
PHP 8.2.27 (cli) (built: Feb  6 2025 17:35:56) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.27, Copyright (c) Zend Technologies
[ec2-user@ip-172-31-29-85 ~]$
```

And next we need to install MariaDB SQL server using the following command.

```
sudo amazon-linux-extras install mariadb10.5 -y
```

```
[ec2-user@ip-172-31-29-85 ~]$ sudo amazon-linux-extras install mariadb10.5 -y
Topic mariadb10.5 has end-of-support date of 2025-06-24
Installing mariadb
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-kernel-5.10 amzn2extra-mariadb10.5 amzn2extra-php8.2
22 metadata files removed
8 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
amzn2extra-docker
amzn2extra-kernel-5.10
amzn2extra-mariadb10.5
amzn2extra-php8.2
(1/11): amzn2-core/2/x86_64/group_gz | 3.6 kB 00:00:00
(2/11): amzn2-core/2/x86_64/updateinfo | 2.9 kB 00:00:00
(3/11): amzn2extra-docker/2/x86_64/updateinfo | 3.0 kB 00:00:00
(4/11): amzn2extra-mariadb10.5/2/x86_64/updateinfo | 3.0 kB 00:00:00
(5/11): amzn2extra-kernel-5.10/2/x86_64/updateinfo | 2.9 kB 00:00:00
(6/11): amzn2extra-docker/2/x86_64/primary_db | 2.7 kB 00:00:00
(7/11): amzn2extra-php8.2/2/x86_64/updateinfo | 1.0 MB 00:00:00
(8/11): amzn2extra-php8.2/2/x86_64/primary_db | 22 kB 00:00:00
(9/11): amzn2extra-mariadb10.5/2/x86_64/primary_db | 7.8 kB 00:00:00
(10/11): amzn2extra-kernel-5.10/2/x86_64/primary_db | 104 kB 00:00:00
(11/11): amzn2-core/2/x86_64/primary_db | 120 kB 00:00:00
Resolving Dependencies
--> Running transaction check
| 6.6 kB 00:00:00
| 111 kB 00:00:00
| 129 kB 00:00:00
| 34 MB 00:00:01
| 74 MB 00:00:01
```

After installation we can see the below installation completed status.

```
Installed:
  mariadb.x86_64 3:10.5.25-1.amzn2

Dependency Installed:
  Judy.x86_64 0:1.0.5-8.amzn2.0.1
  krb5-devel.x86_64 0:1.15.1-55.amzn2.2.8
  libkadm5.x86_64 0:1.15.1-55.amzn2.2.8
  libsepol-devel.x86_64 0:2.5-10.amzn2.0.1
  libverto-devel.x86_64 0:0.2.5-4.amzn2.0.2
  mariadb-common.x86_64 3:10.5.25-1.amzn2
  mariadb-connect-engine.x86_64 3:10.5.25-1.amzn2
  mariadb-devel.x86_64 3:10.5.25-1.amzn2
  mariadb-gssapi-server.x86_64 3:10.5.25-1.amzn2
  mariadb-pam.x86_64 3:10.5.25-1.amzn2
  mariadb-server.x86_64 3:10.5.25-1.amzn2
  mariadb-sphinx-engine.x86_64 3:10.5.25-1.amzn2
  openssl-devel.x86_64 1:1.0.2k-24.amzn2.0.14
  perl-Compress-Raw-Bzip2.x86_64 0:2.061-3.amzn2.0.2
  perl-DBD-MySQL.x86_64 0:4.023-6.amzn2
  perl-Data-Dumper.x86_64 0:2.145-3.amzn2.0.2
  perl-Net-Daemon.noarch 0:0.48-5.amzn2
  perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2
  sphinx.x86_64 0:2.2.11-5.amzn2.0.1

  keyutils-libs-devel.x86_64 0:1.5.8-3.amzn2.0.2
  libcom_err-devel.x86_64 0:1.42.9-19.amzn2.0.1
  libselinux-devel.x86_64 0:2.5-12.amzn2.0.2
  libspnixclient.x86_64 0:2.2.11-5.amzn2.0.1
  mariadb-backup.x86_64 3:10.5.25-1.amzn2
  mariadb-config.x86_64 3:10.5.25-1.amzn2
  mariadb-cracklib-password-check.x86_64 3:10.5.25-1.amzn2
  mariadb-errmsg.x86_64 3:10.5.25-1.amzn2
  mariadb-oqgraph-engine.x86_64 3:10.5.25-1.amzn2
  mariadb-rocksdb-engine.x86_64 3:10.5.25-1.amzn2
  mariadb-server-utils.x86_64 3:10.5.25-1.amzn2
  mytop.noarch 0:1.7-20.b737f60.amzn2
  pcre-devel.x86_64 0:8.32-17.amzn2.0.3
  perl-Compress-Raw-Zlib.x86_64 1:2.061-4.amzn2.0.2
  perl-DBI.x86_64 0:1.627-4.amzn2.0.2
  perl-IO-Compress.noarch 0:2.061-2.amzn2
  perl-PlRPC.noarch 0:0.2020-14.amzn2
  postgresql-libs.x86_64 0:9.2.24-8.amzn2.0.5
  zlib-devel.x86_64 0:1.2.7-19.amzn2.0.3

Dependency Updated:
  mariadb-libs.x86_64 3:10.5.25-1.amzn2

Complete!
  2 httpd_modules           available  [ =1.0  =stable ]
  3 memcached1.5            available  \
    [ =1.5.1  =1.5.16  =1.5.17 ]
  9 R3.4                    available  [ =3.4.3  =stable ]
 10 rust1                  available  \
    [ =1.22.1  =1.26.0  =1.26.1  =1.27.2  =1.31.0  =1.38.0
```

start MariaDB using the following command.

sudo systemctl start mariadb

```
[ec2-user@ip-172-31-29-85 ~]$ sudo systemctl start mariadb
[ec2-user@ip-172-31-29-85 ~]$
```

you can see the completion status as below.

```
[ec2-user@ip-172-31-29-85 ~]$ sudo service mariadb status
Redirecting to /bin/systemctl status mariadb.service
● mariadb.service - MariaDB 10.5 database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled; vendor preset: disabled)
  Active: active (running) since Wed 2025-02-26 06:59:16 UTC; 53s ago
    Docs: man:mariadb(8)
          https://mariadb.com/kb/en/library/systemd/
  Process: 5027 ExecStartPost=/usr/libexec/mariadb-check-upgrade (code=exited, status=0/SUCCESS)
  Process: 4863 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exited, status=0/SUCCESS)
  Process: 4839 ExecStartPre=/usr/libexec/mariadb-check-socket (code=exited, status=0/SUCCESS)
Main PID: 4973 (mariadb)
  Status: "Taking your SQL requests now..."
  CGroup: /system.slice/mariadb.service
          └─4973 /usr/libexec/mariadb --basedir=/usr

Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: The second is mysql@localhost, it has no ...ut
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: you need to be the system 'mysql' user to...t.
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: After connecting you can set the password...be
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: able to connect as any of these users wit...do
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: See the MariaDB Knowledgebase at https://...ra
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: Please report any problems at https://mar...ra
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: The latest information about MariaDB is a.../.
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: Consider joining MariaDB's strong and vib...y:
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal mariadb-prepare-db-dir[4863]: https://mariadb.org/get-involved/
Feb 26 06:59:16 ip-172-31-29-85.ec2.internal systemd[1]: Started MariaDB 10.5 database server.
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-29-85 ~]$
```

Now we are all set to install our wordpress server in our ec2 server. First, we need to change our path and make sure we are installing all in correct place.

```
[ec2-user@ip-172-31-29-85 ~]$ cd ~
[ec2-user@ip-172-31-29-85 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-29-85 ~]$
```

Download all wordpress files using the following command.

[wget https://wordpress.org/latest.tar.gz](https://wordpress.org/latest.tar.gz)

```
[ec2-user@ip-172-31-29-85 ~]$ wget https://wordpress.org/latest.tar.gz
--2025-02-26 07:02:31-- https://wordpress.org/latest.tar.gz
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 26780969 (26M) [application/octet-stream]
Saving to: 'latest.tar.gz'

  0%          --.-K/s          5%  1,510,991   7.20MB/s      55%  14,831,183  35.2MB/s
  0% 26,780,969  43.6MB/s  in 0.6s

2025-02-26 07:02:31 (43.6 MB/s) - 'latest.tar.gz' saved [26780969/26780969]

[ec2-user@ip-172-31-29-85 ~]$ ls
latest.tar.gz
[ec2-user@ip-172-31-29-85 ~]$
```

unzip the tar file using the following command.

[tar -xvf latest.tar.gz](#) after installation you can find the wordpress file.

```
[ec2-user@ip-172-31-29-85 ~]$ ls
latest.tar.gz
[ec2-user@ip-172-31-29-85 ~]$ tar -xzf latest.tar.gz
[ec2-user@ip-172-31-29-85 ~]$ ls
latest.tar.gz  wordpress
[ec2-user@ip-172-31-29-85 ~]$
```

Till now we installed all servers like php, mariadb and wordpress for our website now we will create a database to connect our server.

Go to aws console and search for rds.

click on create database.

Amazon RDS

Dashboard

Databases

Query Editor

Performance Insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL Integrations [New](#)

Events

Event subscriptions

Recommendations [0](#)

Certificate update

Resources

You are using the following Amazon RDS resources in the US East (N. Virginia) region (used/quota)

DB Instances (0/40)

- Allocated storage (0 TB/100 TB)
- Instances and storage include Neptune and DocumentDB.
- [Increase DB instances limit](#)

DB Clusters (0/40)

- Reserved instances (0/40)
- Snapshots (2)
 - Manual
 - DB Cluster (0/100)
 - DB Instance (1/100)
- Automated
- DB Cluster (0)
 - DB Instance (1)
- Recent events (8)
- Event subscriptions (0/20)

Parameter groups (1)

- Default (1)
- Custom (0/100)

Option groups (1)

- Default (1)
- Custom (0/20)

Subnet groups (1/50)

- Supported platforms [VPC](#)
- Default network vpc-01ea9b28b6573725

Recommended services

Customers like you also use these services.

No recommendations yet

Recommended services will display based on your AWS console usage.

Recommended for you

Amazon RDS Backup and Restore using AWS Backup

Learn how to backup and restore Amazon RDS databases using AWS Backup in just 10 minutes. [Learn more](#)

Time-Series Tables in PostgreSQL

Step-by-step guide to design high-performance time series data tables on Amazon RDS for PostgreSQL. [Learn more](#)

Migrate SSRS to RDS for SQL Server

Learn how you can migrate existing SSRS content to an Amazon RDS for SQL Server instance using a PowerShell module. [Learn more](#)

Implementing Cross-Region DR

Learn how to set up Cross-Region disaster recovery (DR) for Aurora PostgreSQL using an Aurora global database spanning multiple Regions. [Learn more](#)

Additional information

[Getting started with RDS](#) [View service health dashboard](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

choose database creation method as **standard create** and engine option as **MySQL**.

Choose a database creation method

Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

MySQL **Aurora (MySQL Compatible)** **Aurora (PostgreSQL Compatible)** **PostgreSQL** **Oracle** **IBM Db2** **MariaDB** **Microsoft SQL Server**

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Select engine version as **MySQL 8.0.40** and template as **free tier**.

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

RDS > Create database

MySQL

MySQL Community

Engine version [Info](#)
View the engine versions that support the following database features.

Hide filters

Show only versions that support the Multi-AZ DB cluster [Info](#)
Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Show only versions that support the Amazon RDS Optimized Writes [Info](#)
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Engine version
MySQL 8.0.40

Enable RDS Extended Support [Info](#)
Amazon RDS Extended Support is a [paid offering](#). By selecting this option, you consent to being charged for this offering if you are running your database major version past the RDS end of standard support date for that version. Check the end of standard support date for your major version in the [RDS for MySQL documentation](#).

Templates
Choose a sample template to meet your use case.

Production
Use defaults for high availability and fast, consistent performance.

Dev/Test
This instance is intended for development use outside of a production environment.

Free tier
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

Availability and durability as **single AZ-db instance** give your database a name as **wordpressdb**.

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

RDS > Create database

MySQL

Availability and durability

Deployment options [Info](#)
Choose the deployment option that provides the availability and durability needed for your use case. AWS is committed to a certain level of uptime depending on the deployment option you choose. Learn more in the [Amazon RDS service level agreement \(SLA\)](#).

Multi-AZ DB cluster deployment (3 instances)
Creates a primary DB instance with two readable standbys in separate Availability Zones. This setup provides:

- 99.5% uptime
- Redundancy across Availability Zones
- Increased read capacity
- Reduced write latency

Multi-AZ instance deployment (2 instances)
Creates a primary DB instance with a non-readable standby instance in a separate Availability Zone. This setup provides:

- 99.5% uptime
- Redundancy across Availability Zones

Single-AZ DB instance deployment (1 instance)
Creates a single DB instance without standby instances. This setup provides:

- 99.5% uptime
- No data redundancy

Settings

DB instance identifier [Info](#)
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

set credentials and give a password for your database make sure to remember your credentials.

Credentials Settings

Master username: **admin** (Info)

Master password: ********* (Info)

Confirm master password: *********

Instance configuration

DB instance class: **db.t4g.micro** (Info)

Hide filters

- Show instance classes that support Amazon RDS Optimized Writes (Info)
- Include previous generation classes

Storage type: **General Purpose SSD (gp2)**

Allocated storage: **20** GiB

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

select instance configuration as **burstable classes t4g.micro** and storage as **general purpose SSD (gp2)**

Instance configuration

DB instance class: **db.t4g.micro** (Info)

Hide filters

- Show instance classes that support Amazon RDS Optimized Writes (Info)
- Include previous generation classes

Storage

Storage type: **General Purpose SSD (gp2)**

Allocated storage: **20** GiB

MySQL

MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.

- Supports database size up to 64 TiB.
- Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
- Supports automated backup and point-in-time recovery.
- Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

select connectivity as don't connect to ec2 instance. And select VPC and give a subnet group. You need to give **public access as YES** to accessible over internet.

The screenshot shows the 'Create database' wizard on the AWS RDS console. The 'Connectivity' section is expanded, showing options for 'Compute resource' (radio buttons for 'Don't connect to an EC2 compute resource' and 'Connect to an EC2 compute resource'), 'Virtual private cloud (VPC)' (radio button for 'Default VPC'), and 'DB subnet group' (radio button for 'Yes'). The 'DB subnet group' section shows a dropdown menu with 'default-vpc-01e9b28b6573725' selected. The 'Public access' section shows 'Yes' selected. The right sidebar displays information about MySQL, including its popularity and various features.

create a security group name as **rds-securitygroup**.

The screenshot shows the 'Create database' wizard on the AWS RDS console. The 'VPC security group (firewall)' section is expanded, showing 'Create new' selected and a button to 'Create new VPC security group'. The 'New VPC security group name' field contains 'rds-securitygroup'. The 'Availability Zone' section shows 'No preference'. The 'RDS Proxy' section shows 'Create an RDS Proxy' unchecked. The 'Certificate authority - optional' section shows a dropdown with 'rds-ca-rsa2048-g1 (default)' selected. The right sidebar displays information about MySQL.

You can leave remaining setting as it is and click on create database.

Log exports
Select the log types to publish to Amazon CloudWatch Logs

- Audit log
- Error log
- General log
- iam-db-auth-error log
- Slow query log

IAM role
The following service-linked role is used for publishing logs to CloudWatch Logs.

RDS service-linked role

Additional configuration
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Estimated monthly costs
The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:

- 750 hrs of Amazon RDS in a Single-AZ db.t2.micro, db.t3.micro or db.t4g.micro Instance.
- 20 GB of General Purpose Storage (SSD).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

Learn more about AWS Free Tier. [?](#)
When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the [Amazon RDS Pricing page](#).

ⓘ You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

Cancel Create database

You can see the `wordpressdb` is in creating status it will take 5 to 10 mins to get ready our db instance.

Creating database `wordpressdb`
Your database might take a few minutes to launch. You can use settings from `wordpressdb` to simplify configuration of suggested database add-ons while we finish creating your DB for you.

View credential details X

Databases (1)

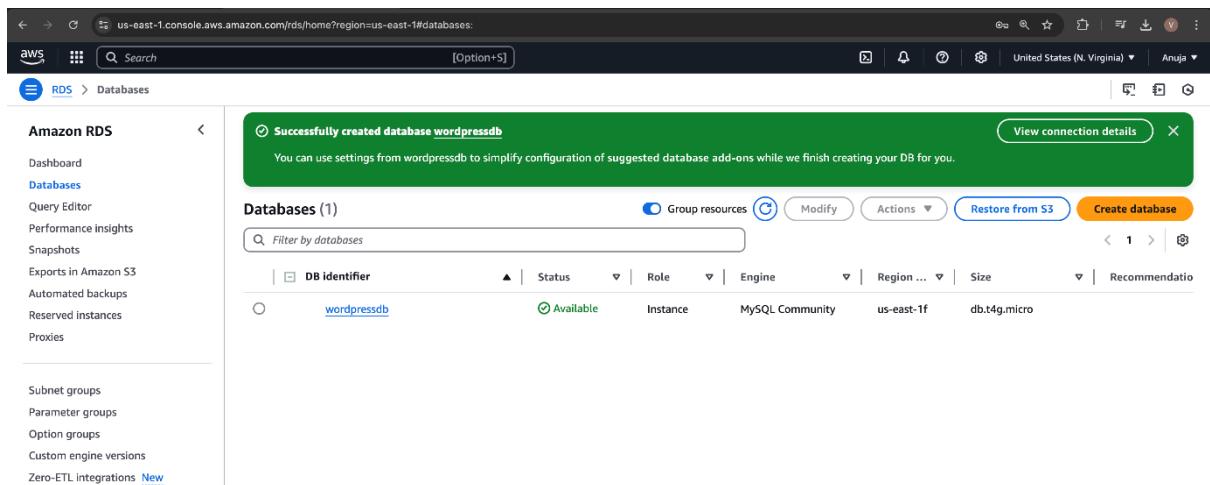
Group resources C Modify Actions Restore from S3 Create database

Filter by databases

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations
<code>wordpressdb</code>	Creating	Instance	MySQL Community	-	db.t4g.micro	

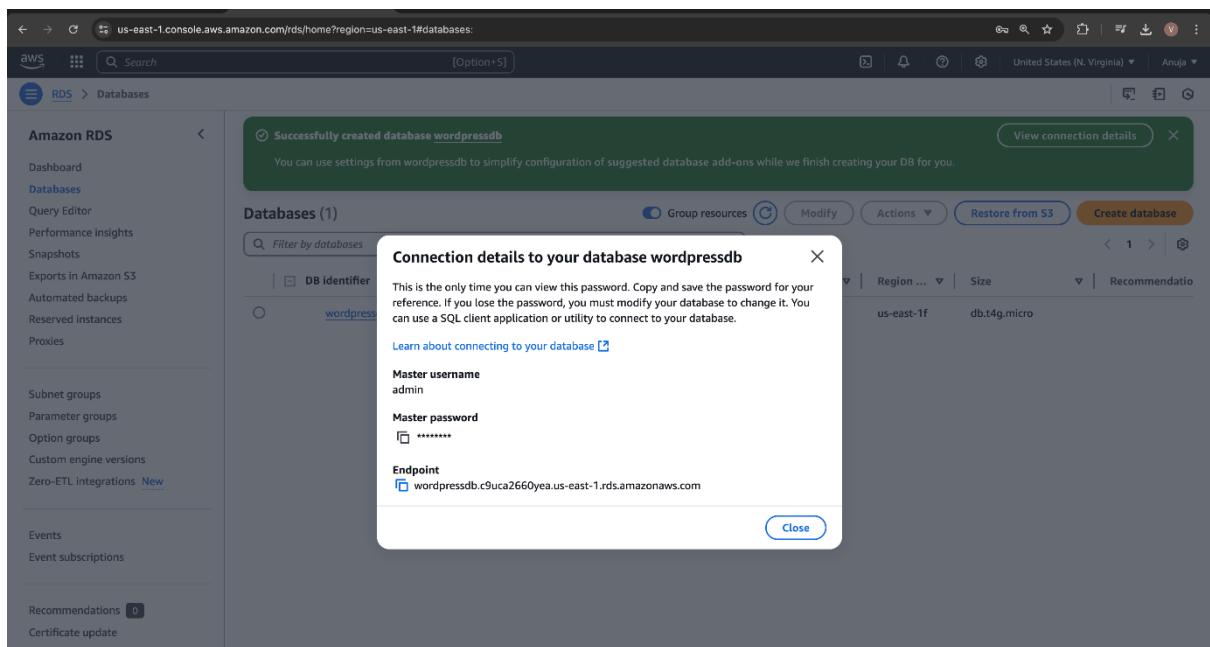
Subnet groups
Parameter groups
Option groups
Custom engine versions
Zero-ETL integrations New

once your database is created you can see the below status. It will show database status as available.



The screenshot shows the AWS RDS console with the URL us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databases. The left sidebar is titled 'Amazon RDS' and includes links for Dashboard, Databases (which is selected), Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, and Zero-ETL integrations. The main content area shows a green success message: 'Successfully created database wordpressdb'. Below it, a table lists the database 'wordpressdb' with details: Status (Available), Role (Instance), Engine (MySQL Community), Region (us-east-1), and Size (db.t4g.micro). There are buttons for 'Group resources', 'Modify', 'Actions', 'Restore from S3', and 'Create database'. A 'View connection details' button is also present.

On top of this page, you can see in green colour view connection details. If you press that one you will find your database credentials make sure to note them.



The screenshot shows the same AWS RDS console as above, but with a modal window open titled 'Connection details to your database wordpressdb'. The modal contains the following information: 'Master username' (admin), 'Master password' (redacted), and 'Endpoint' (wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com). A 'Learn about connecting to your database' link is also present. The 'Close' button is at the bottom right of the modal.

After that go to databases then you press on our database then you will find the end point to our database this will be most important for our website.

Amazon RDS

Dashboard

Databases

Query Editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations [New](#)

Events

Event subscriptions

Recommendations [0](#)

Certificate update

wordpressdb

Summary

DB identifier	wordpressdb	Status	Available	Role	Instance	Engine	MySQL Community	Recommendations
CPU	3.68%	Class	db.t4g.micro	Current activity	0 Connections	Region & AZ	us-east-1f	

Connectivity & security

Endpoint & port

Endpoint	wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com
Port	3306

Networking

Availability Zone	us-east-1f
VPC	vpc-01ea9bb28b6573725
Subnet group	default-vpc-01ea9bb28b6573725
Subnets	subnet-03af60b058d09d9d4 subnet-08f524f51ca3bb40d subnet-0080e93ce86fc270 subnet-00c63de386d4938c5 subnet-09c3306a74c21e14 subnet-0b4fecfaf4d0d7fe0

Security

VPC security groups	rds-securitygroup (sg-0a7476b4e4b21bf7a) Active
Publicly accessible	Yes
Certificate authority	Info rds-ca-rsa2048-g1
Certificate authority date	May 26, 2061, 05:04 (UTC+05:30)
DB instance certificate expiration date	February 26, 2026, 12:43 (UTC+05:30)

Go to security group of your db instance and add the inbound rules.
Copy your ec2 security group.

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

Security Groups (3) [Info](#)

Create security group

Name	Security group ID	Security group name	VPC ID	Description
-	sg-0547c5dc688a8eb0f	default	vpc-01ea9bb28b6573725	default VPC security
-	sg-0a7476b4e4b21bf7a	rds-securitygroup	vpc-01ea9bb28b6573725	Created by RDS mana
-	sg-03d2101970d2d74a6	launch-wizard-2	vpc-01ea9bb28b6573725	launch-wizard-2 creat

EC2 > Security Groups > sg-0a7476b4e4b21bf7a - rds-securitygroup

sg-0a7476b4e4b21bf7a - rds-securitygroup

Details

Security group name rds-securitygroup	Security group ID sg-0a7476b4e4b21bf7a	Description Created by RDS management console	VPC ID vpc-01ea9bb28b6573725
Owner 061039761121	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

Inbound rules | Outbound rules | Sharing - new | VPC associations - new | Tags

Inbound rules (1)

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-030510988db4c1141	IPv4	MySQL/Aurora	TCP	3306

Then in inbound rule add your rule MySql/Aurora and in source add your security group. And click on save rules.

EC2 > Security Groups > sg-0a7476b4e4b21bf7a - rds-securitygroup > Edit inbound rules

Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-030510988db4c1141	MySQL/Aurora	TCP	3306	Custom	106.221.177.21/32
-	MySQL/Aurora	TCP	3306	Custom	sg-03d2101970d2d74a6

Add rule | Cancel | Preview changes | Save rules

And you can see your Ec2 security group is added in the database security group.

The screenshot shows the AWS EC2 Security Groups page. A green success message at the top right states: "Inbound security group rules successfully modified on security group (sg-0a7476b4e4b21bf7a | rds-securitygroup) Details". The main section displays the security group "sg-0a7476b4e4b21bf7a - rds-securitygroup". The "Details" table includes fields: Security group name (rds-securitygroup), Security group ID (sg-0a7476b4e4b21bf7a), Description (Created by RDS management console), VPC ID (vpc-01ea9bb28b6573725), Owner (061039761121), Inbound rules count (2 Permission entries), and Outbound rules count (1 Permission entry). Below the table are tabs for Inbound rules, Outbound rules, Sharing - new, VPC associations - new, and Tags. The Inbound rules table shows two entries:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-030510988db4c1141	IPv4	MySQL/Aurora	TCP	3306
-	sgr-03f9ce0f61230adf3	-	MySQL/Aurora	TCP	3306

To check your database you can connect to ec2 instance connect and type this command.

Mysql -h Your database end point -u admin -p and enter your database password.

The screenshot shows an EC2 instance terminal. The session starts with a welcome message for Amazon Linux 2, followed by a note about the end of life for AL2. It then prompts the user to enter a password for the MySQL command. The command entered is "mysql -h wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com -u admin -p".

```

Last login: Wed Feb 26 06:50:01 2025 from 106.221.177.21
# Amazon Linux 2
-- AL2 End of Life is 2026-06-30.
-- A newer version of Amazon Linux is available!
-- Amazon Linux 2023, GA and supported until 2028-03-15.
-- https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-172-31-29-85 ~]$ mysql -h wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com -u admin -p
Enter password: 

```

Finally you can see database is added.

```

Last login: Wed Feb 26 06:50:01 2025 from 106.221.177.21
[ec2-user@ip-172-31-29-85 ~]$ mysql -h wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 27
Server version: 8.0.40 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> 

```

And then press **show databases;** command you can see your word press db instance.

```

MySQL [(none)]> CREATE DATABASE wordpressdb;
Query OK, 1 row affected (0.011 sec)

MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| wordpressdb |
+-----+
5 rows in set (0.002 sec)

```

Now copy your wordpress folder to **var/www/html/** using the following command.

```
sudo cp -r wordpress/* /var/www/html/
```

```

[ec2-user@ip-172-31-29-85 ~]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-172-31-29-85 ~]$ 

```

Now Adding your current user to Apache group.

```

[ec2-user@ip-172-31-29-85 ~]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-172-31-29-85 ~]$ sudo usermod -a -G apache ec2-user
[ec2-user@ip-172-31-29-85 ~]$ 

```

Then you need to exit from the current Ec2 instance and login back again.

```
[ec2-user@ip-172-31-29-85 ~]$ sudo cp -r wordpress/* /var/www/html/
[ec2-user@ip-172-31-29-85 ~]$ sudo usermod -a -G apache ec2-user
[ec2-user@ip-172-31-29-85 ~]$ exit
logout
Connection to ec2-100-25-218-33.compute-1.amazonaws.com closed.
raj@Rajs-Laptop desktop %
```

Now we need to change permission on var/www folder using these 3 commands.

```
sudo chown -R ec2-user:apache /var/www
```

```
sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod
2775 {} \;
```

```
find /var/www -type f -exec sudo chmod 0664 {} \;
```

and go to that `cd /var/www/html` and do `ls` we will find all the required files.

```
[ec2-user@ip-172-31-29-85 ~]$ sudo chown -R ec2-user:apache /var/www
[ec2-user@ip-172-31-29-85 ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod 2775 {} \;
[ec2-user@ip-172-31-29-85 ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
[ec2-user@ip-172-31-29-85 ~]$ cd /var/www/html
[ec2-user@ip-172-31-29-85 html]$ ls
index.php      wp-activate.php      wp-comments-post.php  wp-cron.php      wp-load.php      wp-settings.php  xmlrpc.php
license.txt    wp-admin            wp-config-sample.php wp-includes      wp-login.php    wp-signup.php
readme.html    wp-blog-header.php  wp-content          wp-links-opml.php wp-mail.php    wp-trackback.php
[ec2-user@ip-172-31-29-85 html]$
```

Now we need to copy the config file to another file. Using this command.

```
cp wp-config-sample.php wp-config.php
```

```
[ec2-user@ip-172-31-29-85 ~]$ sudo chown -R ec2-user:apache /var/www
[ec2-user@ip-172-31-29-85 ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod 2775 {} \;
[ec2-user@ip-172-31-29-85 ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
[ec2-user@ip-172-31-29-85 ~]$ cd /var/www/html
[ec2-user@ip-172-31-29-85 html]$ ls
index.php  wp-activate.php  wp-comments-post.php  wp-cron.php  wp-load.php  wp-settings.php  xmlrpc.php
license.txt  wp-admin  wp-config-sample.php  wp-includes  wp-login.php  wp-signup.php
readme.html  wp-blog-header.php  wp-content  wp-links-opml.php  wp-mail.php  wp-trackback.php
[ec2-user@ip-172-31-29-85 html]$ cp wp-config-sample.php wp-config.php
[ec2-user@ip-172-31-29-85 html]$ ls
index.php  wp-activate.php  wp-comments-post.php  wp-content  wp-links-opml.php  wp-mail.php  wp-trackback.php
license.txt  wp-admin  wp-config.php  wp-cron.php  wp-load.php  wp-settings.php  xmlrpc.php
readme.html  wp-blog-header.php  wp-config-sample.php  wp-includes  wp-login.php  wp-signup.php
[ec2-user@ip-172-31-29-85 html]$
```

And now we need to open config file using Nano editor.

nano wp-config.php

```
[ec2-user@ip-172-31-29-85 html]$ nano wp-config.php
```

Here we need to add all our database details like user name, password, our db end point.

```
GNU nano 2.9.8                               wp-config.php

<?php
/** 
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the website, you can copy this file to "wp-config.php"
 * and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * Database settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://developer.wordpress.org/advanced-administration/wordpress/wp-config/
 *
 * @package WordPress
 */

// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'database_name_here' );

/** Database username */
define( 'DB_USER', 'username_here' );

/** Database password */
define( 'DB_PASSWORD', 'password_here' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );

[ Read 102 lines (Converted from DOS format) ]
^G Get Help      ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos    M-U Undo    M-A Mark Text
^X Exit        ^R Read File    ^\ Replace    ^U Uncut Text  ^T To Spell   ^_ Go To Line  M-E Redo    M-6 Copy Text
```

After editing we can see the file like this press **ctrl+s** to save and **ctrl+x** to exit.

```
// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpressdb' );

/** Database username */
define( 'DB_USER', 'admin' );

/** Database password */
define( 'DB_PASSWORD', 'admin123' );

/** Database hostname */
define( 'DB_HOST', 'wordpressdb.c9uca2660yea.us-east-1.rds.amazonaws.com' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

/**#@+
 * Authentication unique keys and salts.
 */
```

After that open config file using vim editor.

```
sudo vim /etc/httpd/conf/httpd.conf
```

```
[ec2-user@ip-172-31-29-85 html]$ sudo vim /etc/httpd/conf/httpd.conf
```

after that file looks like this.

```

#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/html"

#
# Relax access to content within /var/www.
#
<Directory "/var/www">
    AllowOverride None
    # Allow open access:
    Require all granted
</Directory>

# Further relax access to the default document root:
<Directory "/var/www/html">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important. Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    # for more information.
    #

```

Now we need to change **AllowOverride none** to **All**

```

#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/html"

#
# Relax access to content within /var/www.
#
<Directory "/var/www">
    AllowOverride All
    # Allow open access:
    Require all granted
</Directory>

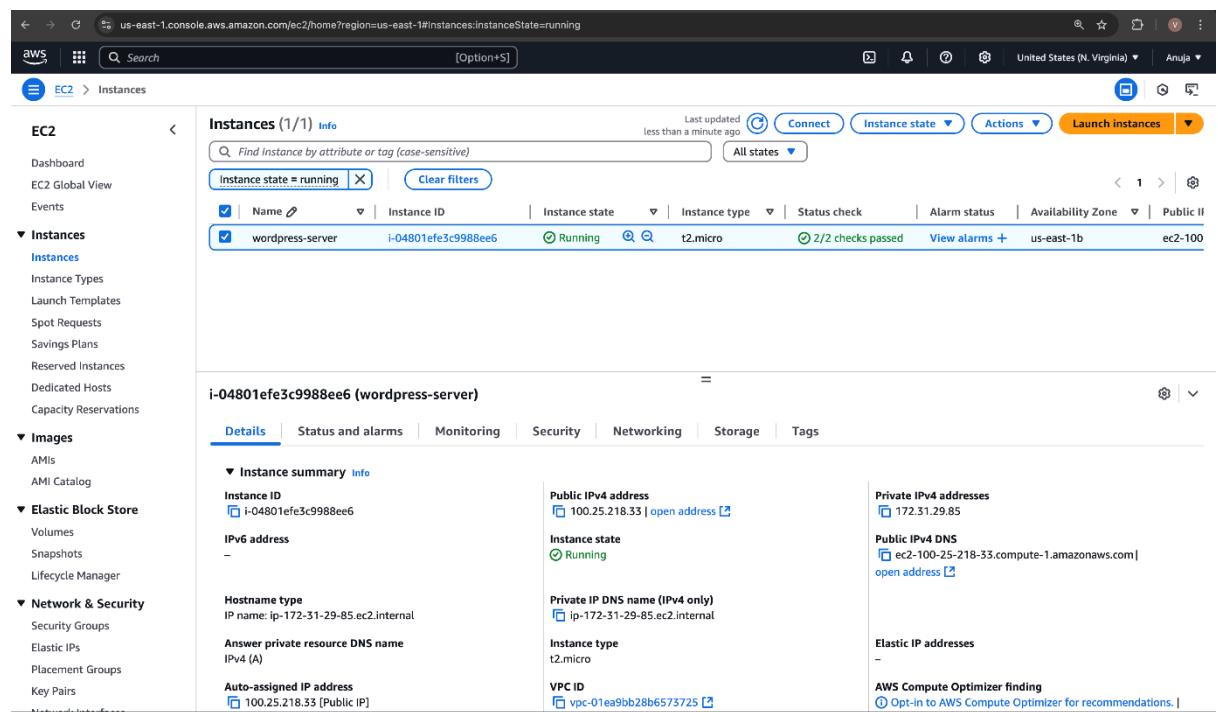
# Further relax access to the default document root:
<Directory "/var/www/html">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important. Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    #
    Options Indexes FollowSymLinks

```

Now we need to restart the httpd server.

```
[ec2-user@ip-172-31-29-85 html]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-29-85 html]$
```

Finally you are ready to open your instance. In a browser open your public ip. Then you will find the wordpress website opened.

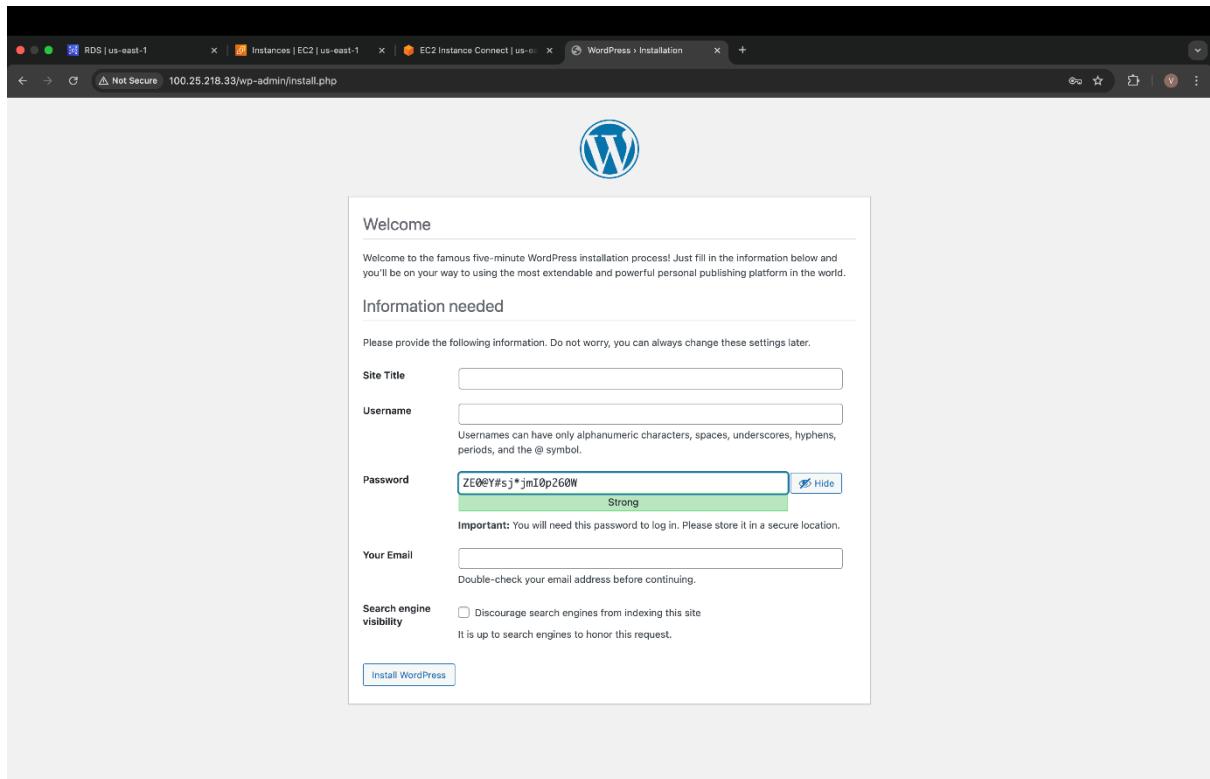


The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main content area displays a table of instances. One instance is listed:

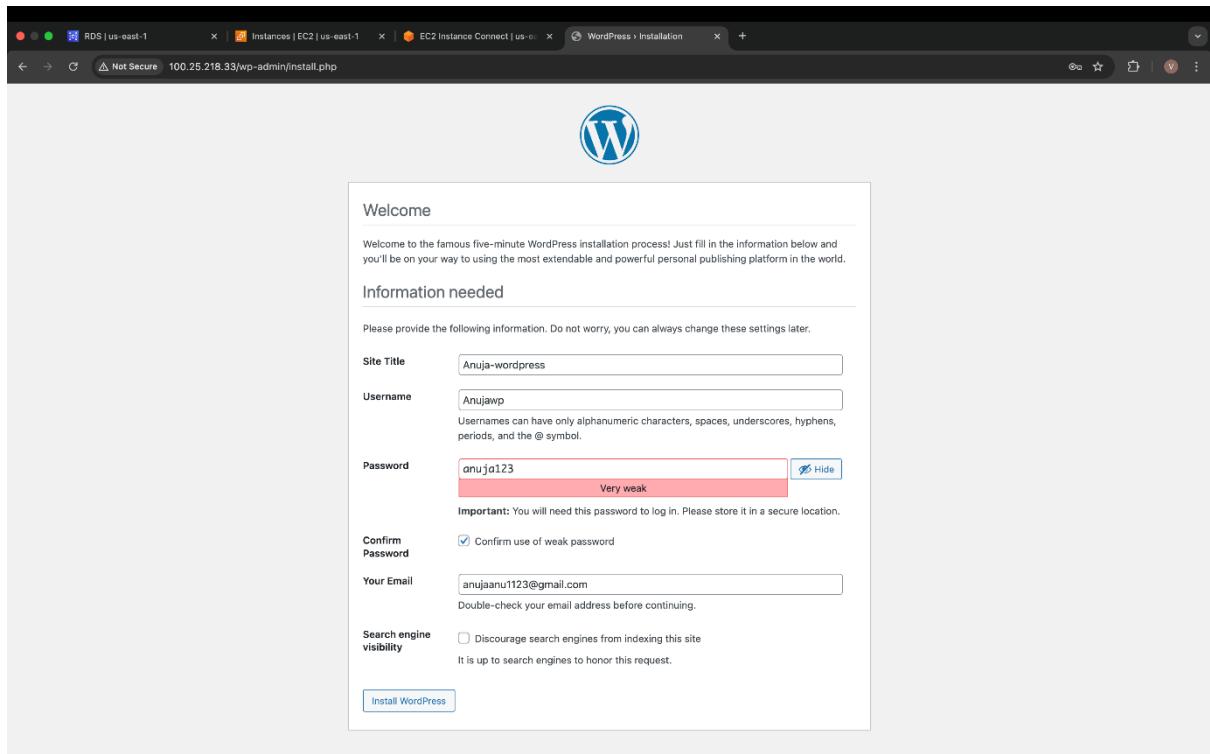
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
wordpress-server	i-04801efe3c9988ee6	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-100-25-218-33.compute-1.amazonaws.com

Below the table, the details for the instance i-04801efe3c9988ee6 are shown. The 'Details' tab is selected. The instance summary section includes:

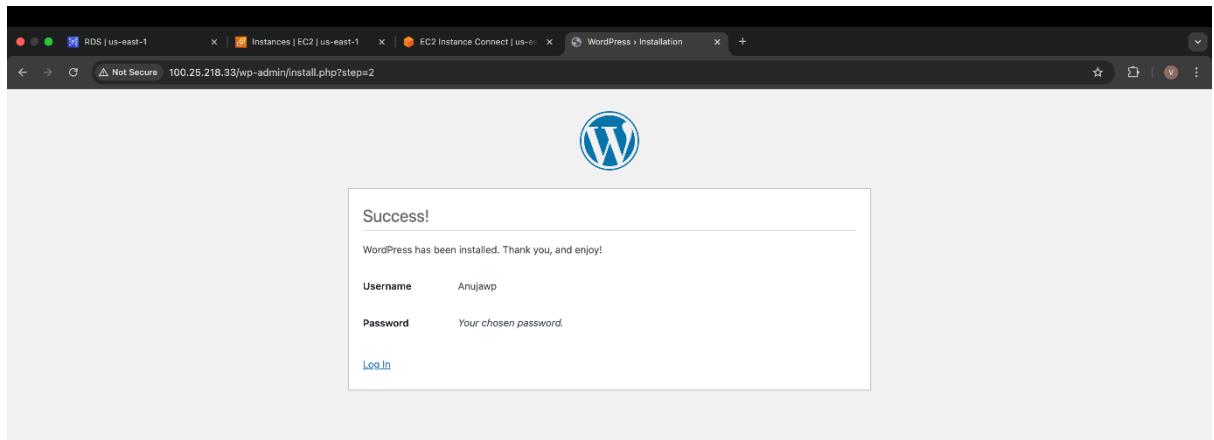
- Instance ID: i-04801efe3c9988ee6
- IPv6 address: -
- Hostname type: IP name: ip-172-31-29-85.ec2.internal
- Answer private resource DNS name: IPv4 (A)
- Auto-assigned IP address: 100.25.218.33 [Public IP]
- Public IP4 address: 100.25.218.33 [open address]
- Instance state: Running
- Private IP4 DNS name (IPv4 only): ip-172-31-29-85.ec2.internal
- Instance type: t2.micro
- VPC ID: vpc-01ea9bb28b6573725 [2]
- Private IP4 addresses: 172.31.29.85
- Public IP4 DNS: ec2-100-25-218-33.compute-1.amazonaws.com [open address]
- Elastic IP addresses: -
- AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations.



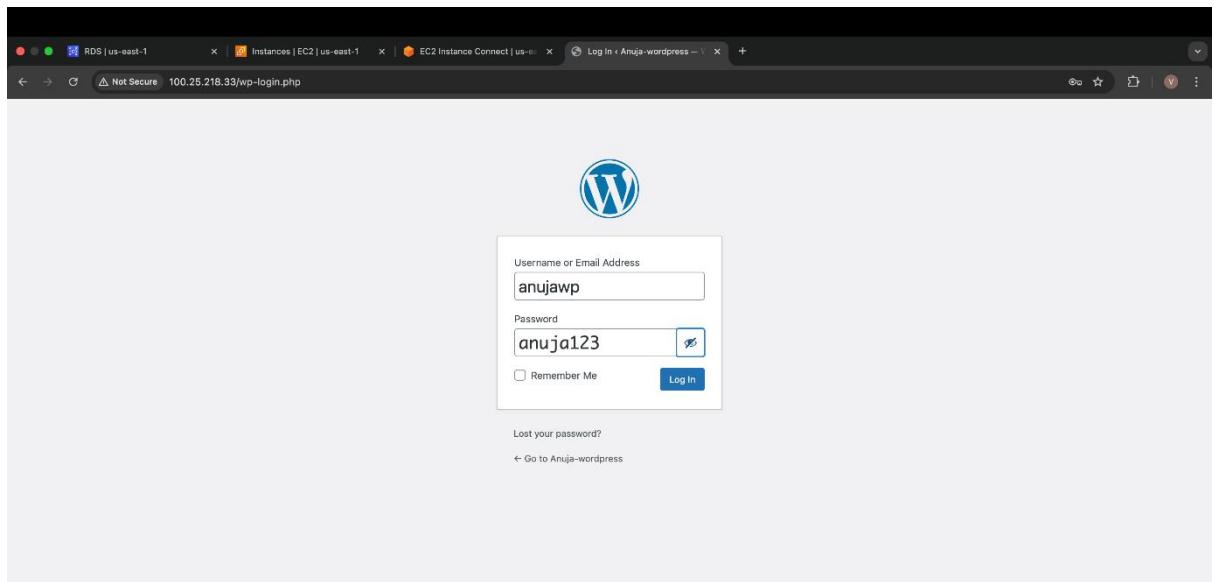
Fill this form and then press install wordpress.



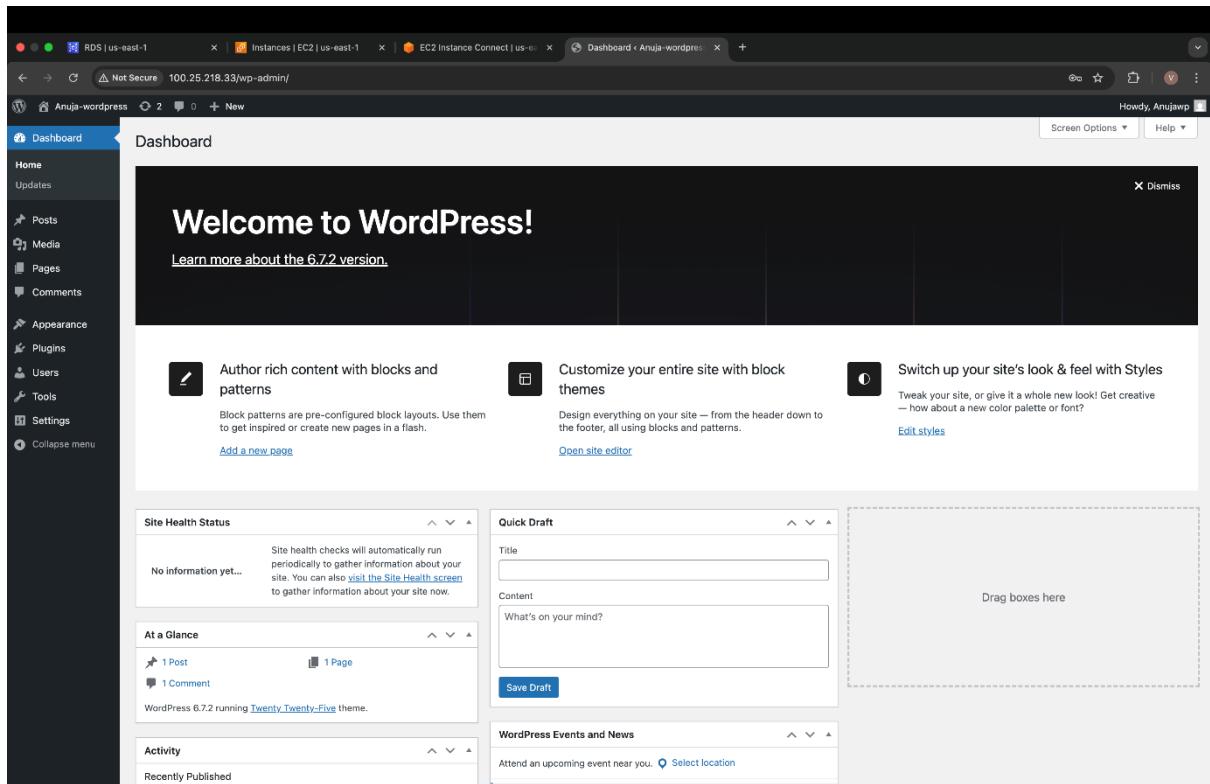
After that you will find success page then login to word press instance.



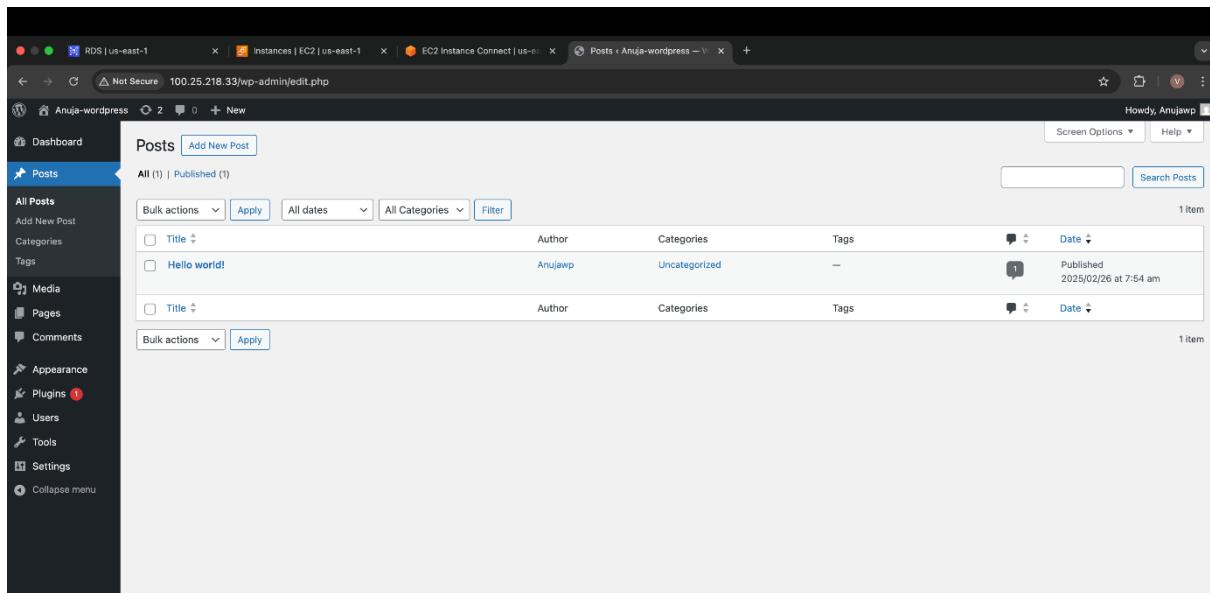
Use your username and password to login to wordpress site.



Finally your word press website is ready.



The screenshot shows the WordPress dashboard with a dark theme. The top navigation bar includes tabs for 'RDS | us-east-1', 'Instances | EC2 | us-east-1', 'EC2 Instance Connect | us-east-1', and 'Dashboard < Anuja-wordpress'. The dashboard features a prominent 'Welcome to WordPress!' banner with a 'Dismiss' button. The left sidebar contains links for 'Home', 'Updates', 'Posts', 'Media', 'Pages', 'Comments', 'Appearance', 'Plugins', 'Users', 'Tools', 'Settings', and a 'Collapse menu' button. The main content area includes sections for 'Author rich content with blocks and patterns', 'Customize your entire site with block themes', and 'Switch up your site's look & feel with Styles'. It also features a 'Site Health Status' box, a 'Quick Draft' editor, and a 'Drag boxes here' area. The bottom left sidebar shows 'At a Glance' statistics: 1 Post, 1 Page, and 1 Comment. The bottom right sidebar shows 'WordPress Events and News'.



The screenshot shows the 'Posts' screen in the WordPress admin. The top navigation bar includes tabs for 'RDS | us-east-1', 'Instances | EC2 | us-east-1', 'EC2 Instance Connect | us-east-1', and 'Posts < Anuja-wordpress'. The sidebar on the left includes links for 'All Posts', 'Add New Post', 'Categories', 'Tags', 'Media', 'Pages', 'Comments', 'Appearance', 'Plugins', 'Users', 'Tools', 'Settings', and a 'Collapse menu' button. The main content area displays a table of published posts. The table columns are 'Title', 'Author', 'Categories', 'Tags', and 'Date'. The first post listed is 'Hello world!' by 'Anujawp' in the 'Uncategorized' category, published on '2025/02/26 at 7:54 am'. The table includes 'Bulk actions' and 'Apply' buttons at the top, and a 'Search Posts' bar at the bottom. A total of '1 item' is shown.

