

# Deploying WordPress website on AWS using elastic beanstalk

## What is Elastic beanstalk

It is simply called serverless computing or you just need to provide a code of your application to aws and aws itself manage everything in their end such as managing load on server, storage, type of instance etc...

## Prerequisites

- An AWS account.
- An SSH key pair for EC2 instances.

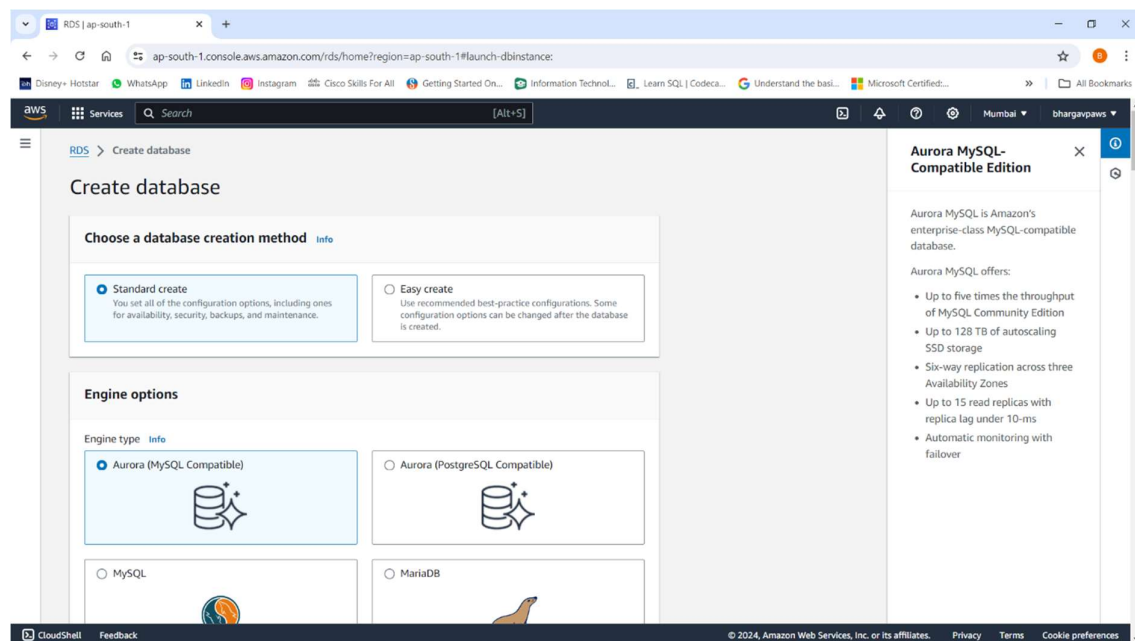
## Launch RDS Database

### 1. Search RDS on the AWS Console:

- Open the AWS Management Console.
- In the search bar, type "RDS" and select it.

### 2. Create Database:

- Click on "Create database."
- Choose "Standard create."
- Select "MySQL" as the engine.



### 3. Configure Database:

- Set a master username and password (or auto-generate the password).
- Under "Additional configuration," give your database a name.
- Keep other settings as default.
- Click on "Create database."
- It takes up to 3-4 minutes to configure.

The screenshot shows the 'Settings' page for a new Aurora MySQL database instance in the AWS console. The 'DB cluster identifier' is set to 'database-1'. Under 'Credentials Settings', the 'Master username' is 'admin'. The 'Credentials management' section shows 'Self managed' as the selected option. The 'Master password' field is empty, with a note that Amazon RDS can generate a password or the user can specify their own. A sidebar on the right provides information about Aurora MySQL, including its throughput and storage capabilities.

**Settings**

**DB cluster identifier** Info  
Enter a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.  
  
The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

**▼ Credentials Settings**

**Master username** Info  
Type a login ID for the master user of your DB instance.  
  
1 to 32 alphanumeric characters. The first character must be a letter.

**Credentials management**  
You can use AWS Secrets Manager or manage your master user credentials.  
☐ Managed in AWS Secrets Manager - most secure  
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.  
☒ Self managed  
Create your own password or have RDS create a password that you manage.  
☐ Auto generate password  
Amazon RDS can generate a password for you, or you can specify your own password.

**Master password** Info  
  
Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / ' \* @

**Aurora MySQL- Compatible Edition**

Aurora MySQL is Amazon's enterprise-class MySQL-compatible database.

Aurora MySQL offers:

- Up to five times the throughput of MySQL Community Edition
- Up to 128 TB of autoscaling SSD storage
- Six-way replication across three Availability Zones
- Up to 15 read replicas with replica lag under 10-ms
- Automatic monitoring with failover

The screenshot shows the 'database-1' instance details page in the AWS console. A green banner at the top indicates 'Successfully created database database-1'. Below this, a table lists related databases. The table has columns for DB identifier, Status, Role, Engine, Region & AZ, Size, Recommendations, CPU, and Current activity. The 'database-1' instance is listed as 'Available' and is a 'Regional cluster' using 'Aurora MySQL' in the 'ap-south-1' region with '2 instances'.

**Successfully created database database-1**  
You can use settings from database-1 to simplify configuration of suggested database add-ons while we finish creating your DB for you.

**database-1**

**Related**

DB identifier	Status	Role	Engine	Region & AZ	Size	Recommendations	CPU	Current activity
database-1	Available	Regional cluster	Aurora MySQL	ap-south-1	2 instances			

**Endpoints (2)**

Endpoint name	Status	Type	Port
---------------	--------	------	------

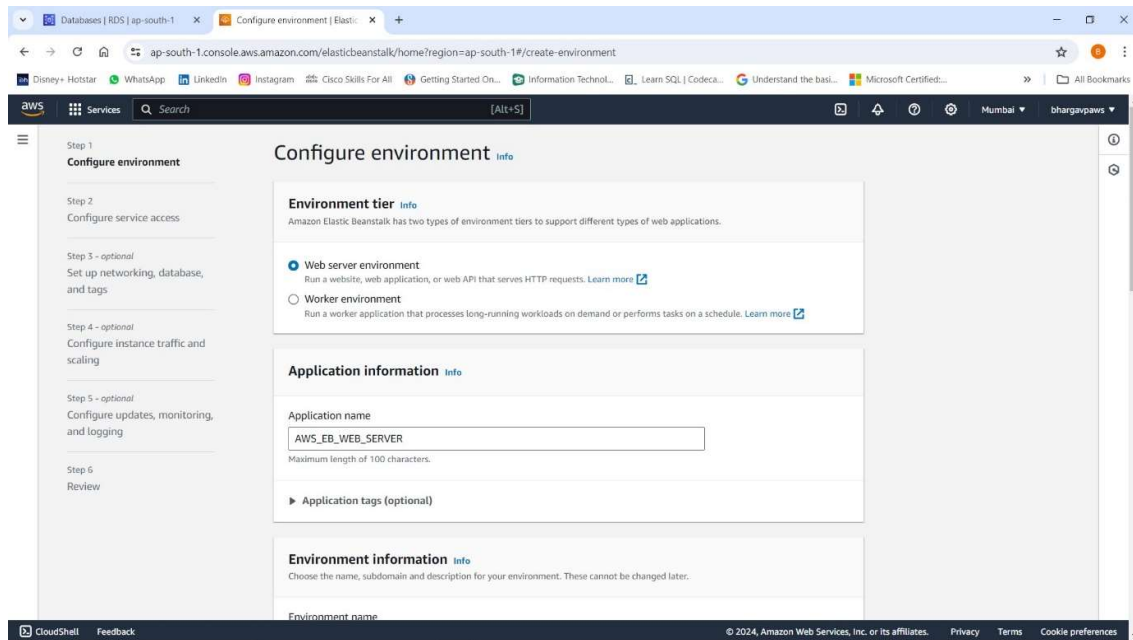
# Launch Elastic Beanstalk Environment

## 1. Search Elastic Beanstalk on the AWS Console:

- In the search bar, type "Elastic Beanstalk" and select it.
- Click on "Create application."

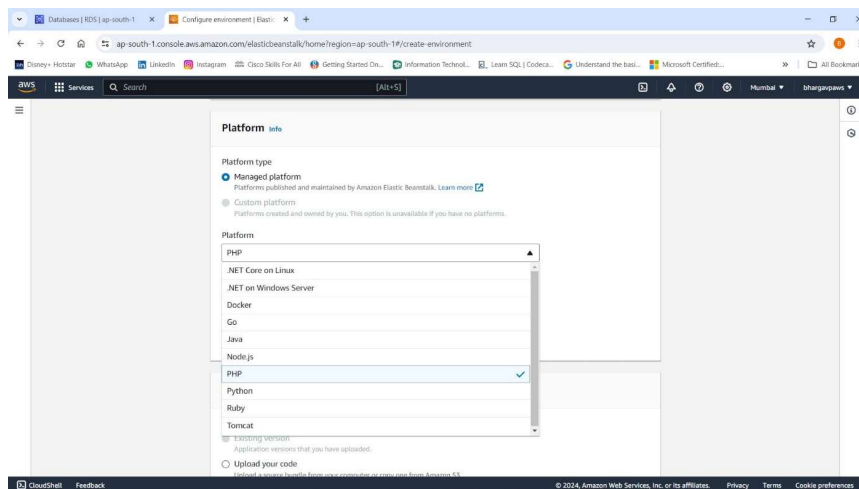
## 2. Configure Application:

- Choose "Web server environment."
- Provide a name for your application.



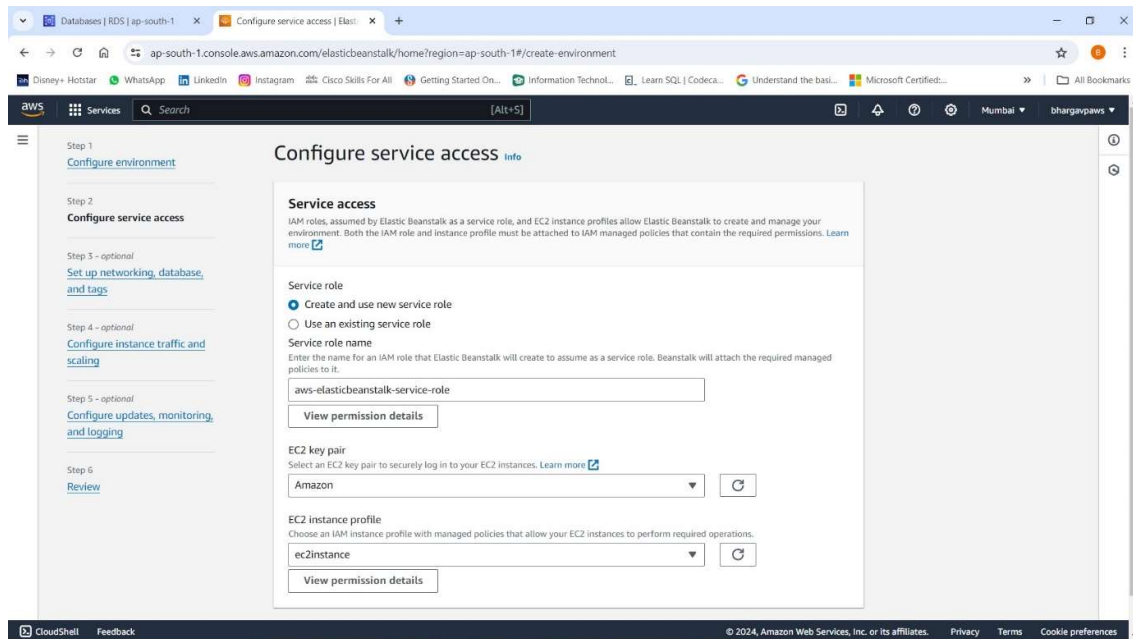
## 3. Choose Platform:

- Select "PHP" from the managed platform options.
- Click "Next."



#### 4. Service Role:

- Click "Create a new service role."
- Select aws-elasticbeanstalk-service-role.

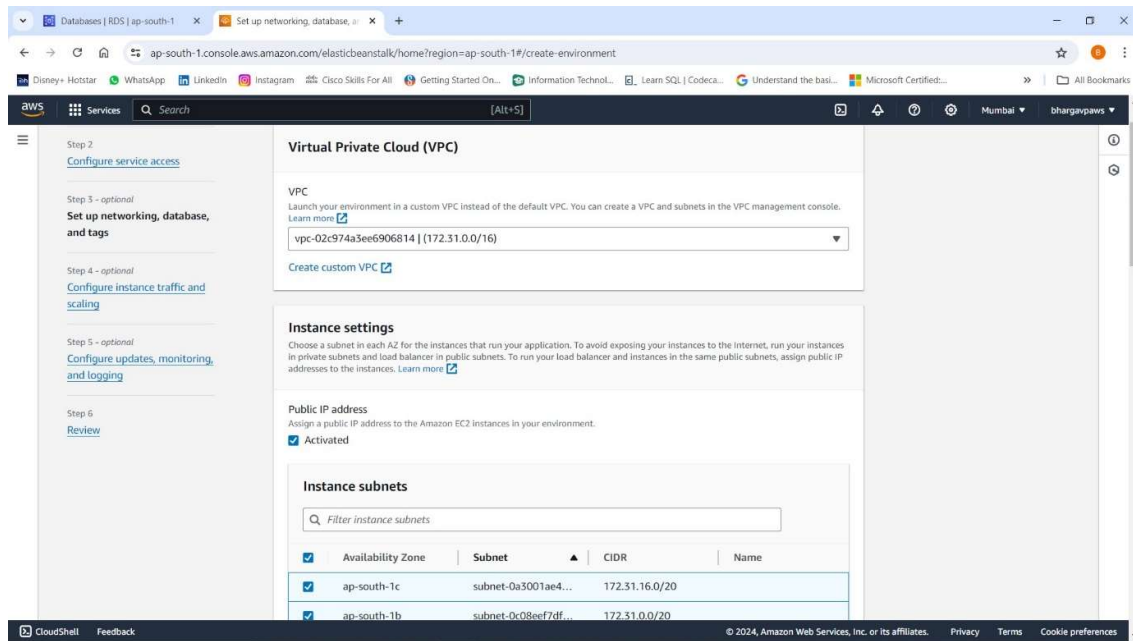


#### 5. EC2 Key Pair and Instance Profile:

- Choose your EC2 key pair.
- Set the EC2 instance profile to ec2instance.
- Note: Ensure the key pair is available on your local machine.

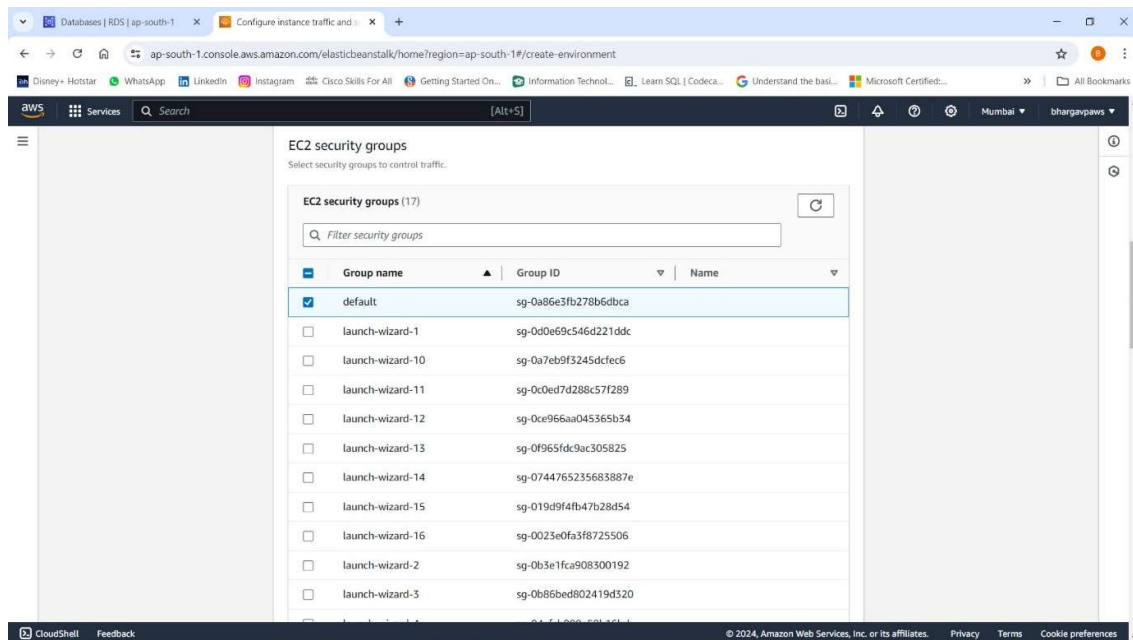
#### 6. VPC and Subnets:

- Select the default VPC.
- Choose the appropriate instance and database subnets.
- Click "Next."



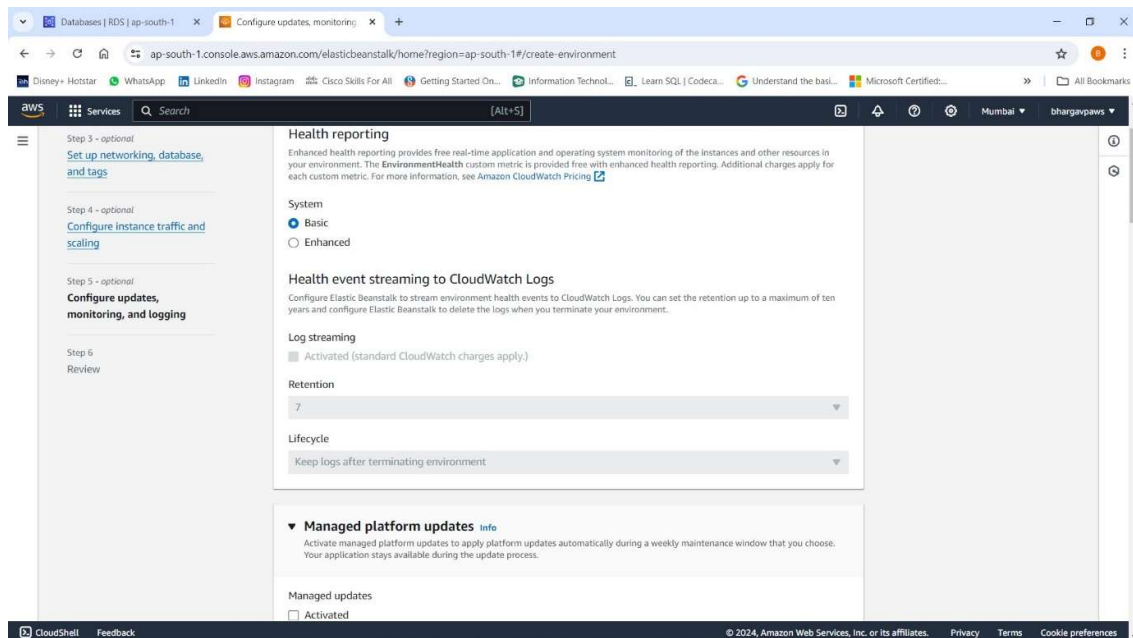
## 7. Security Groups:

- Select the default security group.
- Click "Next."



## 8. Health Reporting:

- Choose "Basic" health reporting.
- Untick the "Managed updates" option.
- Click "Next."



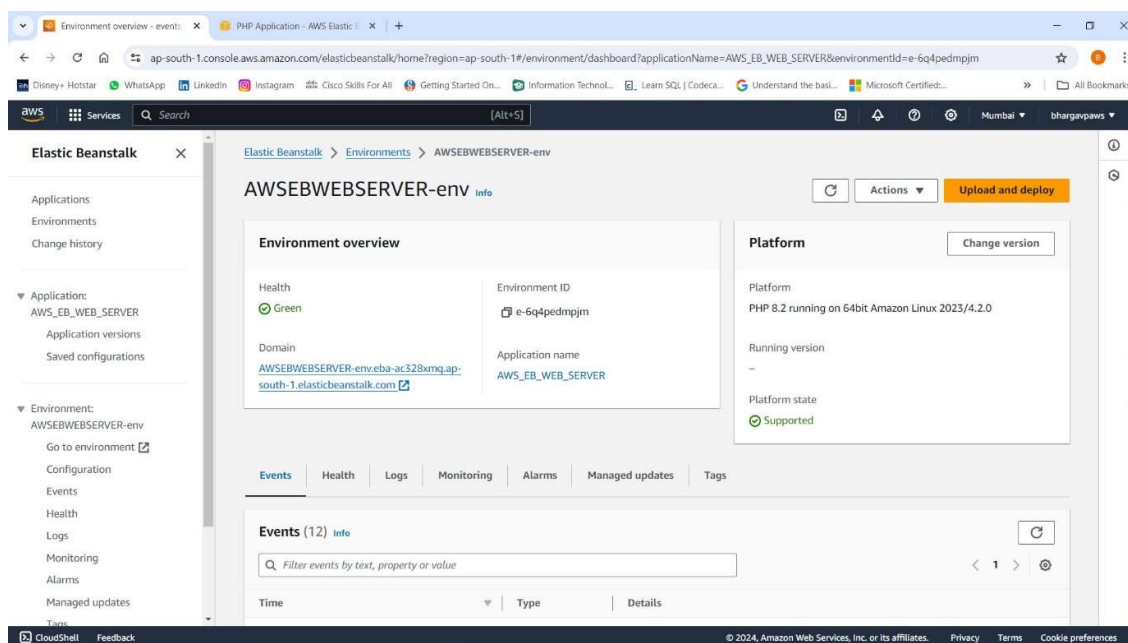
## 9. Connect to Database:

- Add the following environment properties:
  - RDS\_HOSTNAME: The hostname of the DB instance (found under the Connectivity & security tab in the RDS console).
  - RDS\_PORT: The port where the DB instance accepts connections (found under the Connectivity & security tab in the RDS console).
  - RDS\_DB\_NAME: The database name, typically ebdb (found under the Configuration tab in the RDS console).
  - RDS\_USERNAME: The master username you configured for your database (found under the Configuration tab in the RDS console).
  - RDS\_PASSWORD: The master password you configured (not available in the RDS console, use what you set initially).

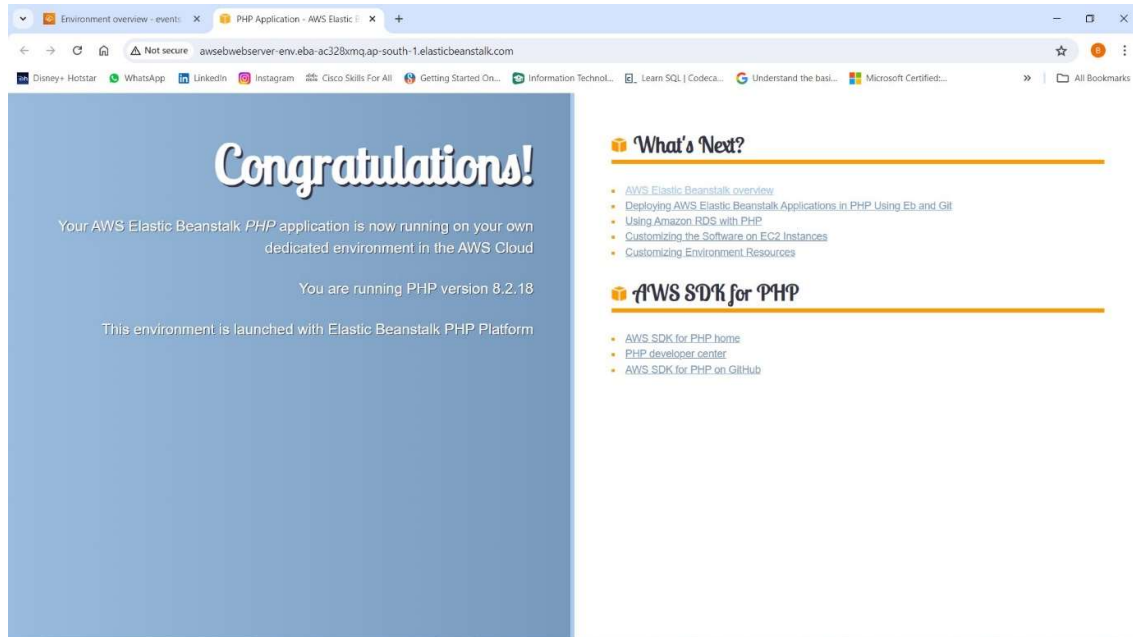
Property	Value
RDS_HOSTNAME	your-db-instance-endpoint
RDS_PORT	3306
RDS_DB_NAME	ebddb
RDS_USERNAME	your-db-username
RDS_PASSWORD	your-db-password

## 10. Review and Create:

- Review your configuration.
- Click on "Submit."
- It takes around 5 minutes to create your application.



- Once created, you will receive a URL to access your Elastic Beanstalk environment.



## Download WordPress

### 1. Prepare Your Environment:

- Open your terminal and create a directory:

```
mkdir git  
cd ~/git
```

### 2. Install Git:

- If not already installed:

```
sudo apt-get update      # If the instance created is ubuntu machine.  
sudo apt-get install git  If it is a centos or Linux use according commands.
```

### 3. Clone WordPress Repository:

- Clone the repository:

```
git clone https://github.com/user-name/deploy-wordpress-
```

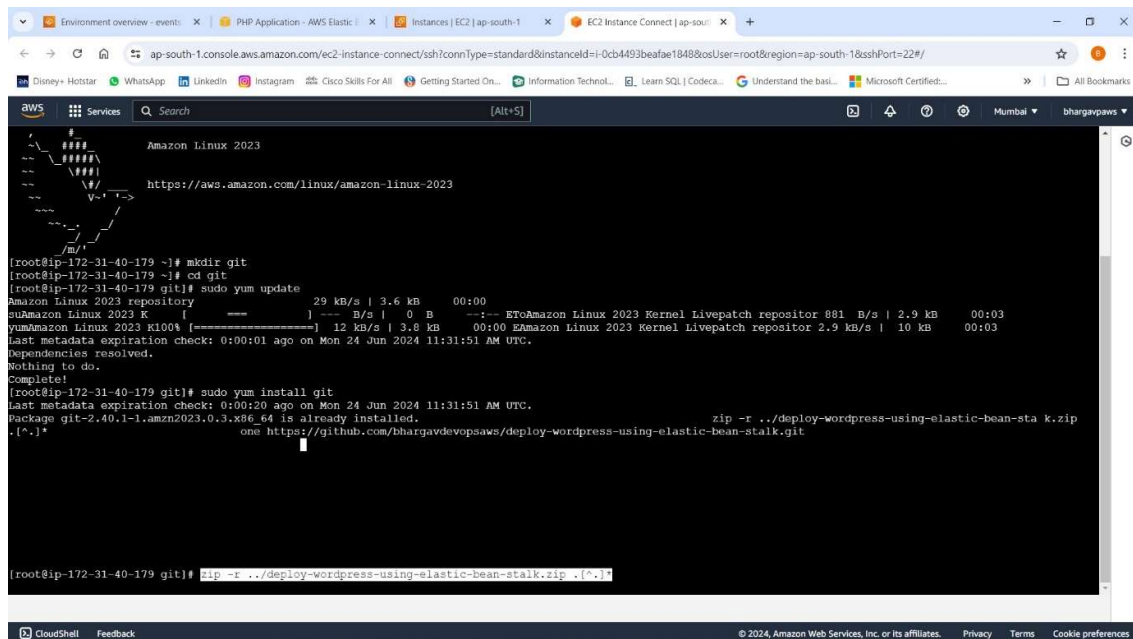
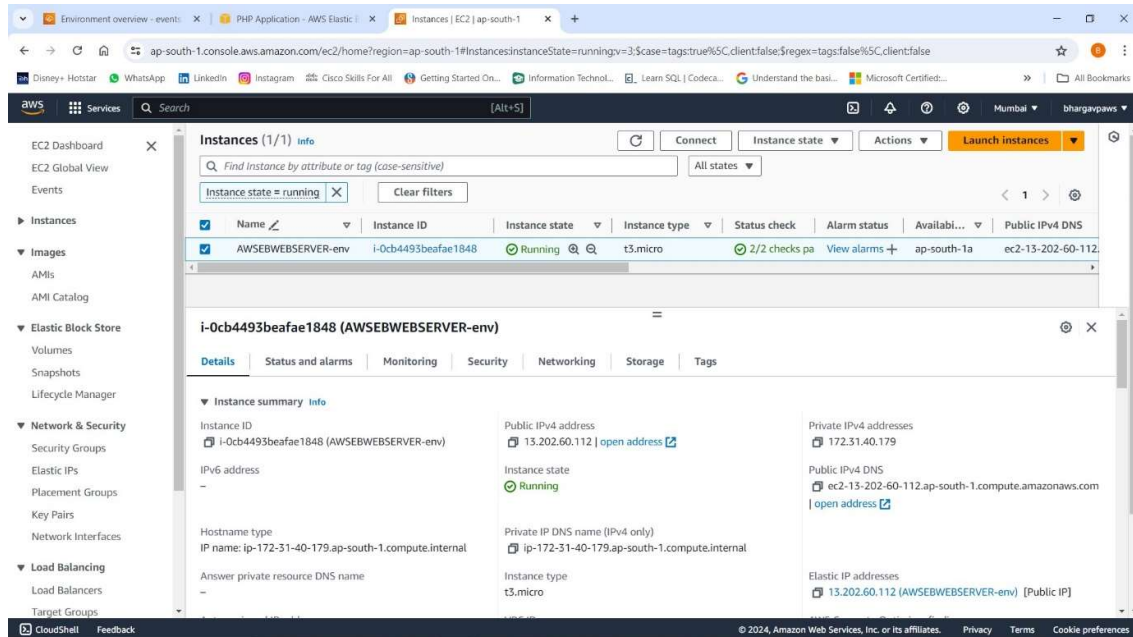


#### 4. Zip the WordPress Files:

- Create a zip file of the WordPress files:

`zip -r ../your_folder_name.zip .[^.]*`

eg: `zip -r ../deploy-elasticbeanstalk-using-wordpress-.zip .[^.]*`

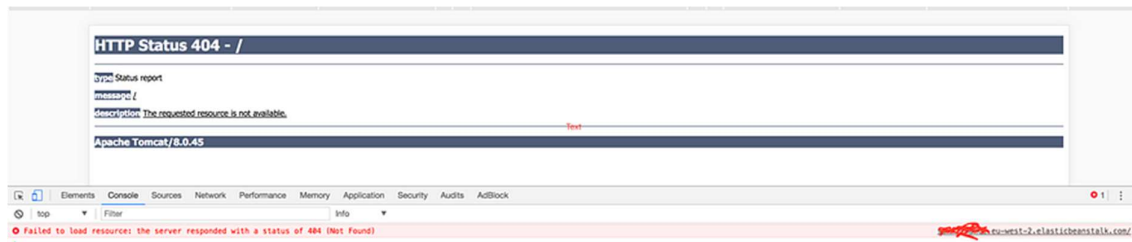


## Upload and Deploy WordPress

### 1. Upload to Elastic Beanstalk:

- Go to your Elastic Beanstalk console.
- Select the environment you created.
- Choose the "Upload and deploy" option.
- Select the WordPress zip file you created and click on "Deploy."
- Deployment takes around 5 minutes.

If 404 error occurs follow these steps



### 2. Fix 404 Nginx Not Found Error:

- Go to the EC2 dashboard.
- Find the instance created by Elastic Beanstalk.
- Copy the public IP address of the instance.

### 3. SSH into the EC2 Instance:

- Open your terminal and SSH into the instance:

```
ssh -i <your-key-pair-name> ec2-user@<public-ip>
```

### 4. Move WordPress Files:

- Switch to the superuser:

```
sudo su
```

- Navigate to the HTML directory:

```
cd /var/www/html/
```

- Move the WordPress files:

```
mv wordpress/* /var/www/html/
```

- Remove the empty WordPress directory:

```
rm -rf wordpress
```

## 5. Access Your WordPress Site:

- Return to the Elastic Beanstalk console.
- Click on the provided URL.
- You should see the WordPress setup page.

