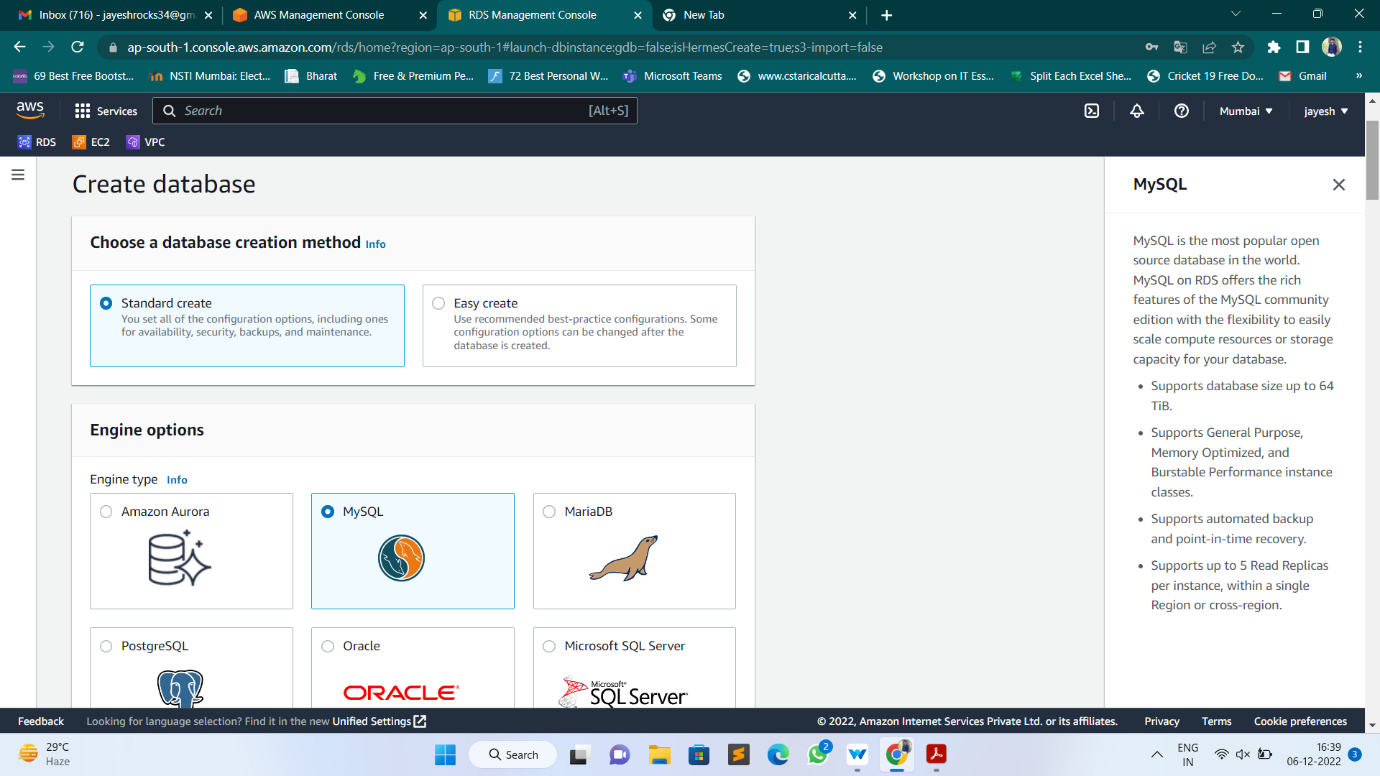
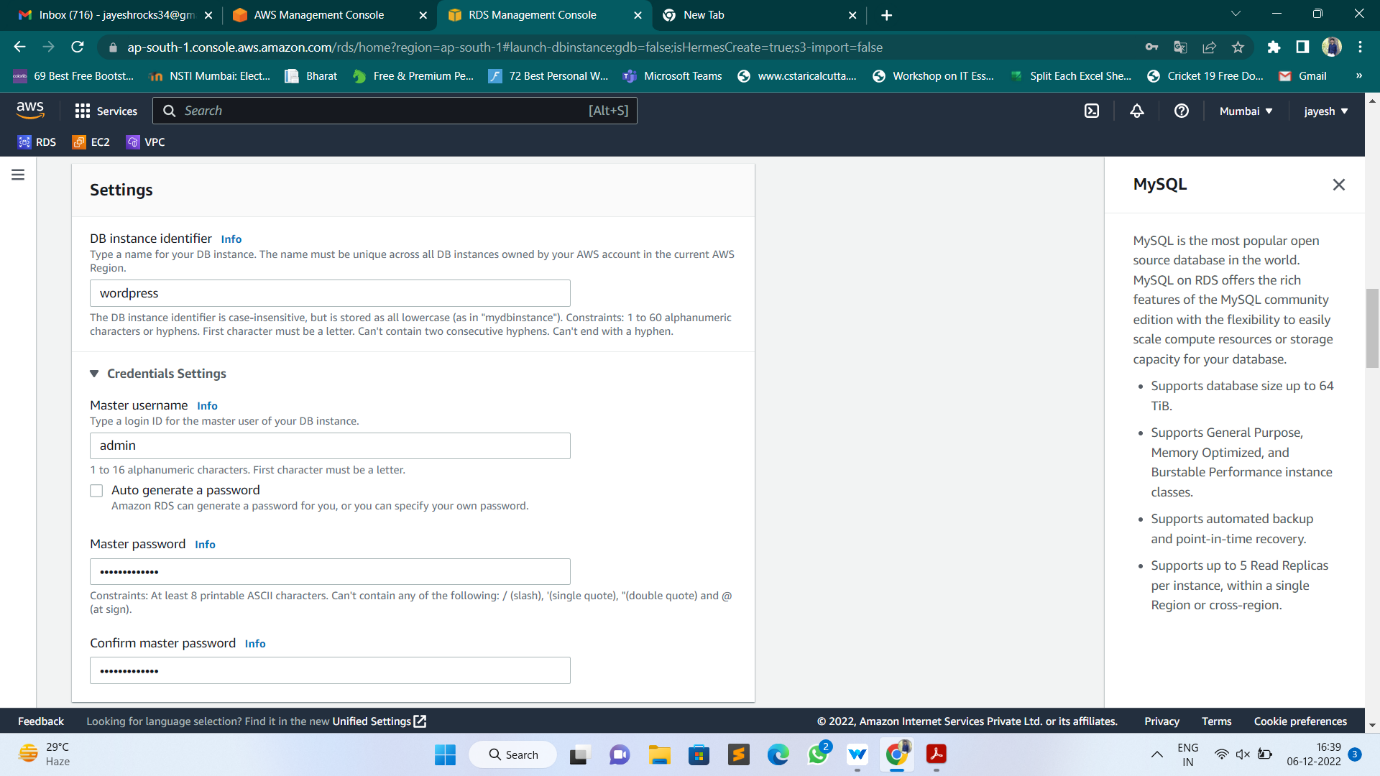
**Deploy WordPress with Amazon RDS**

**Lab – 1: Creating a MySQL Database with RDS.**

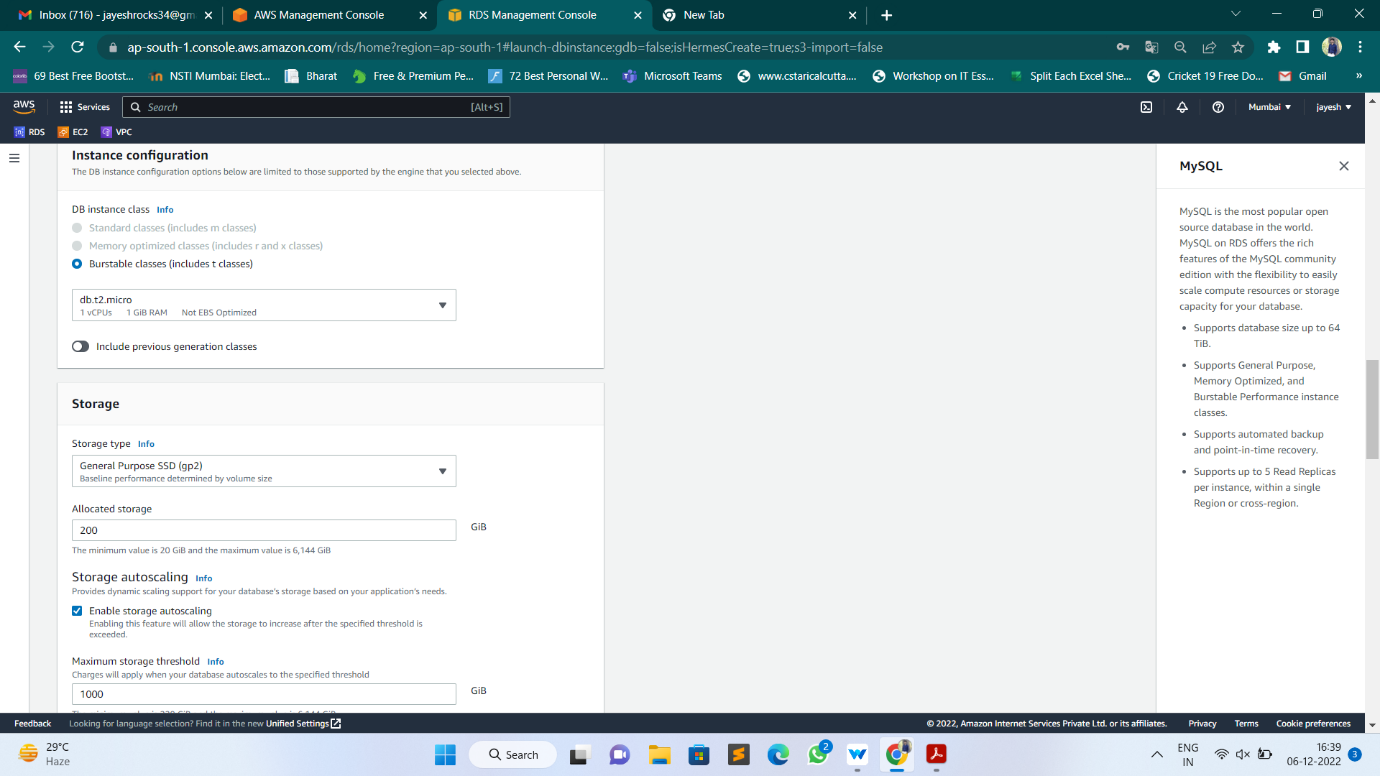
* To begin, go to Amazon RDS in the AWS console. Click the orange Create database button to get started.
* The first step is to choose the database engine you want to use.
* Amazon RDS supports six different engines, from popular open-source options like MySQL and PostgreSQL, to commercial options like Oracle and Microsoft SQL Server, to a cloud-native option called Amazon Aurora that was custom-built to take advantage of the cloud.
* WordPress uses MySQL, so select that engine now.



* Next, you will specify the authentication settings for your MySQL deployment. This includes the database name and the master username and password
* In the Settings section, enter WordPress as your DB instance identifier.
* Then specify the master username and password for your database.
* Choose a strong, secure password to help protect your database.
* Store the username and password for safekeeping as you will need it in a later module.



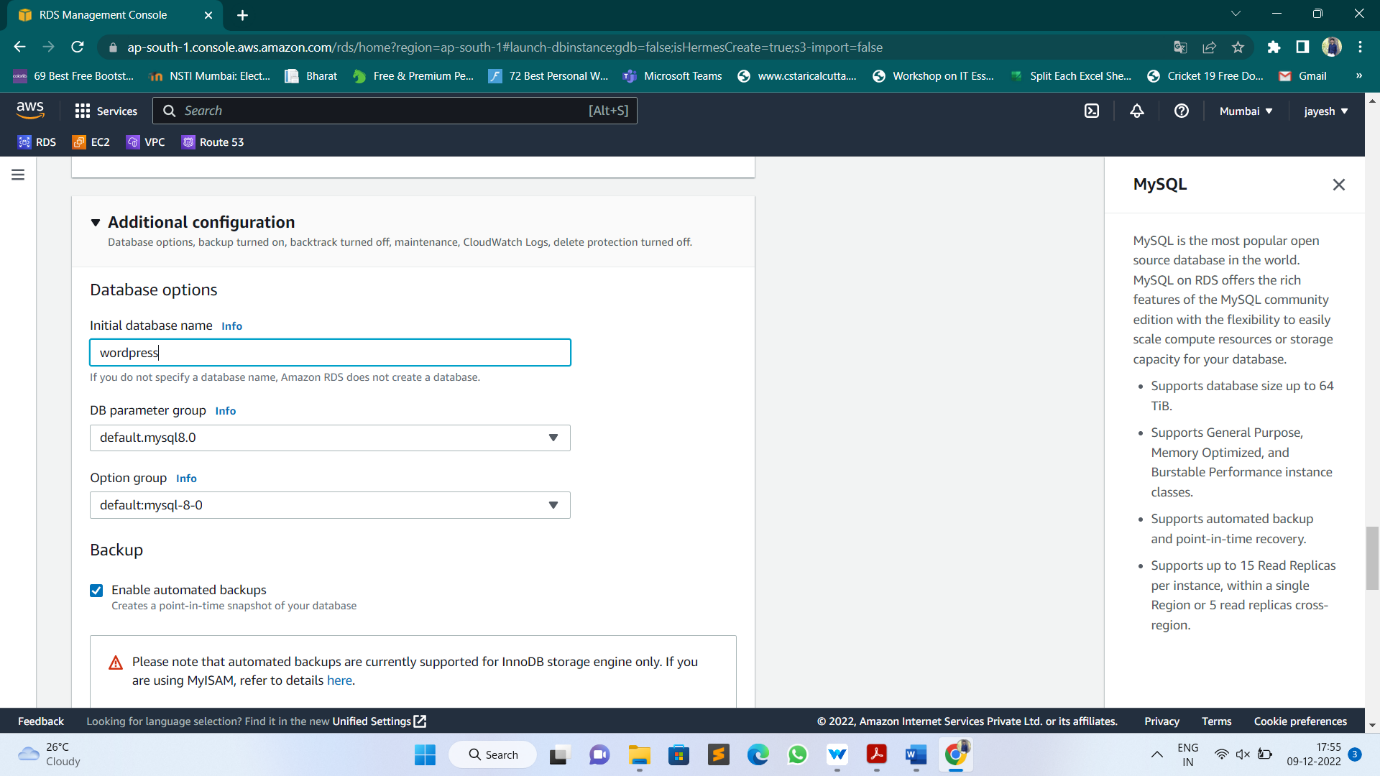
* After setting your username and password, you can select key details about your MySQL deployment. This includes the instance class and storage details.
* The default settings will work for this lab. We will use a small instance class that is suitable for testing or small-scale applications, and it fits within the AWS Free Tier.
* If you don’t want to use the AWS Free Tier, you could set a larger instance class or alter the storage configuration options.



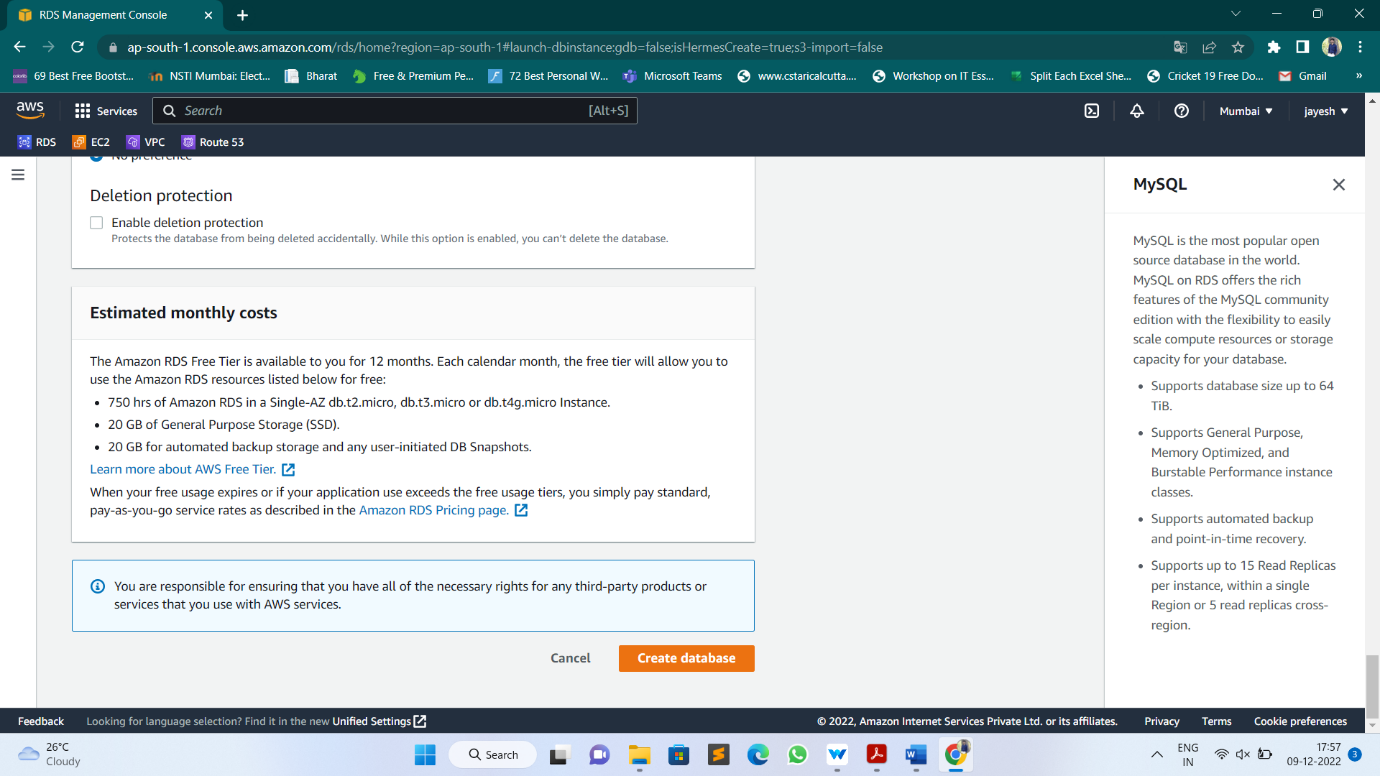
* Next, you can configure connectivity and network configuration. Amazon RDS instances must be created in an Amazon VPC, which is a logically-separate network where your provisioned resources will live.
* VPCs are an advanced topic outside the scope of this lab. Fortunately, AWS has created a default VPC in each region in your account.
* The default VPC is already selected for you, and you can launch your RDS instance in this VPC.
* Finally, RDS provides a number of additional configuration options to customize your deployment. You need to make one change in this area. Click on the Additional configuration line to expand the options.



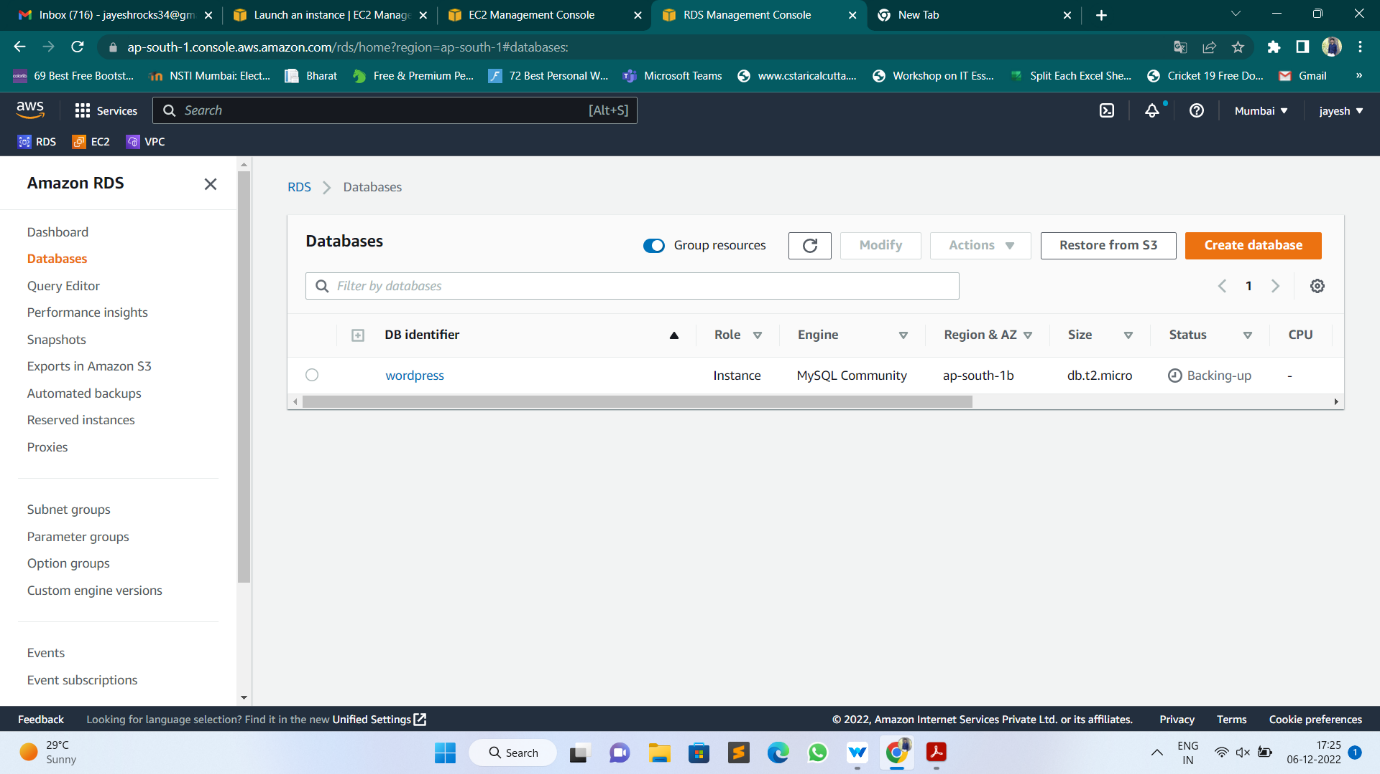
* Set the initial database name to WordPress. This will ensure RDS creates the database in your MySQL instance upon initialization.
* We will use this database name when connecting to your database.



* At the bottom of the creation wizard, AWS will show you estimated monthly costs for your RDS database. If you are still eligible for the Amazon RDS Free Tier, you will see a note that the database will be free to you for up to 12 months.
* Click the orange Create database button to create your database.

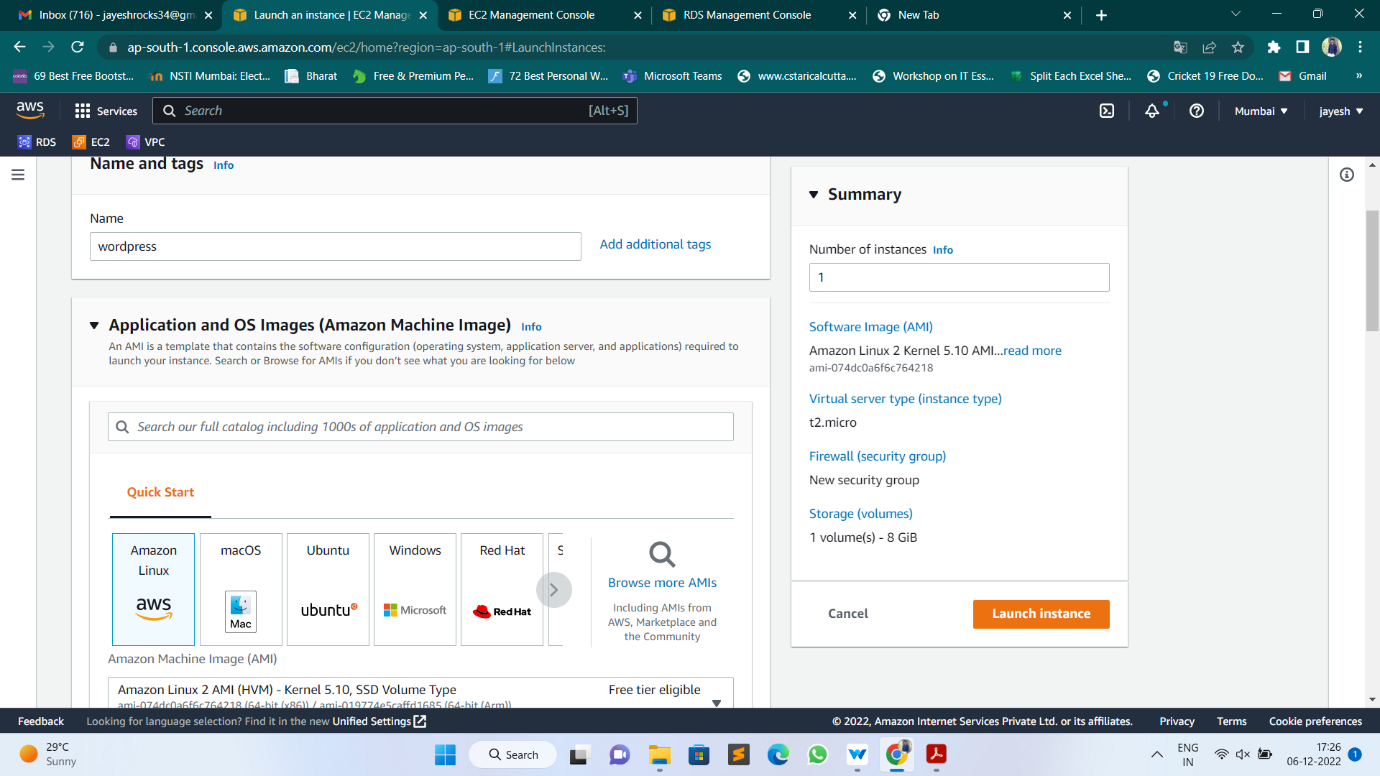


* You should see a success notice indicating that your database is being created.



* In this module, you created a fully-managed MySQL database using Amazon RDS.
* In the next module, we will create an Amazon EC2 instance for running your WordPress site.

**Lab – 2: Creating an EC2 Instance to run WordPress site.**



**Lab – 3: Configuring your RDS database.**

* At this point, you have created an RDS database and an EC2 instance. In this module, we will configure the RDS database to allow access to specific entities.
* Configure the RDS database to allow access to specific entities

**Database Security Methods**

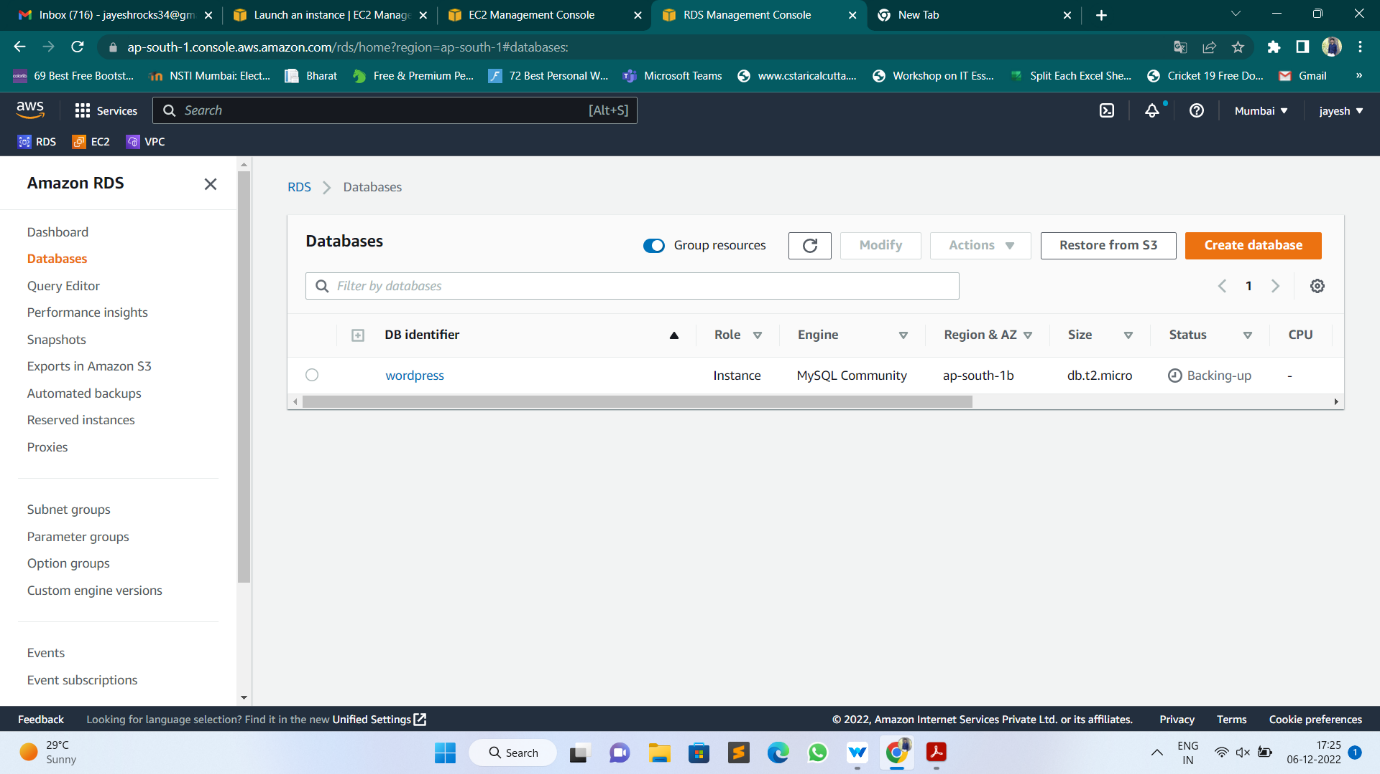
* It is critical to secure your database from unauthorized access, and there are a number of strategies you can use to add security to your database. You will learn two of them in this module. They are:

1. Network security: Limiting access to your database instance by rejecting traffic that’s not from authorized IP addresses
2. Password authentication and authorization: Limiting access to your database by requiring a username and password to access

* You will configure each of these in the steps below.

**Implementation:**

* Allow your EC2 Instance to access your RDS Database
* First, you will modify your RDS database to allow network access from your EC2 instance.
* In the previous module, you created security group rules to allow SSH and HTTP traffic to your WordPress EC2 instance. The same principle applies here.
* This time, you want to allow certain traffic from your EC2 instance into your RDS database.
* To configure this, go to the RDS databases in the AWS console. Click on the MySQL database you created in an earlier module in this lab.



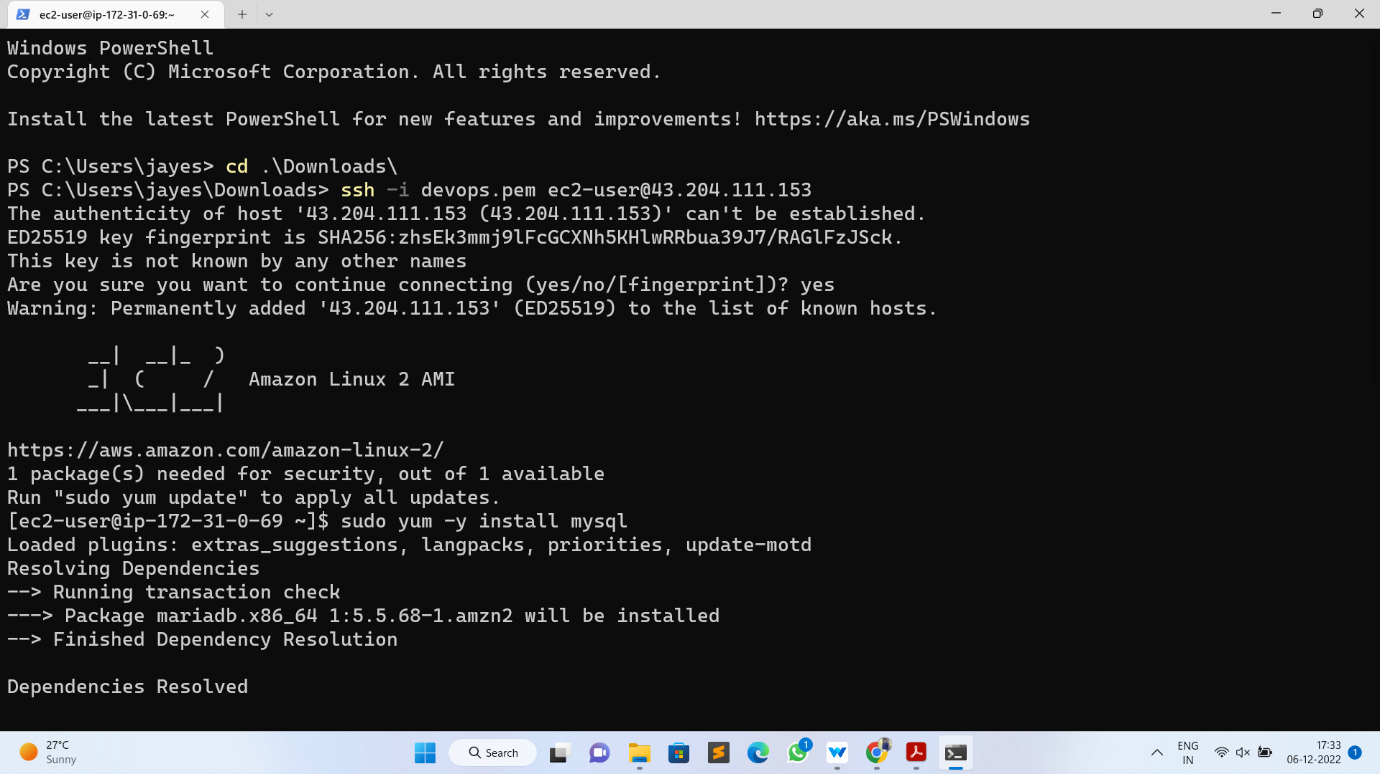
* Scroll to the Connectivity & security tab in the display, and click on the security group listed in VPC security groups.
* The console will take you to the security group configured for your database.
* Click the Inbound tab, then click the Edit button to change the rules for your security group.
* The default security group has a rule that allows all inbound traffic from other instances in the default security group.
* However, since your WordPress EC2 instance is not in that security group, it will not have access to the RDS database.
* Change the Type property to MYSQL/Aurora, which will update the Protocol and Port Range to the proper values.
* Then, remove the current security group value configured for the rule, and type “WordPress” instead. The console will show the available security groups that are configured.
* Click on the “WordPress” security group that you used for your EC2 instance.
* After you click, it will fill in the security group ID. This rule will allow MySQL access to any EC2 instance with that security group configured.
* When you’re finished, hit the blue Save button to save your changes.

**SSH into your EC2 Instance**

* Now that your EC2 instance has access to your RDS database, you will SSH into your EC instance and run some configuration commands.
* Go to the EC2 instances page in the AWS console. You should see the EC2 instance you created for the WordPress installation.
* Click on it, and you will see a public IP address labelled IPv4 Public IP in the instance description.
* Previously, you downloaded the .pem file for the key pair of your instance. Locate that file now. It will likely be in a Downloads folder on your desktop.

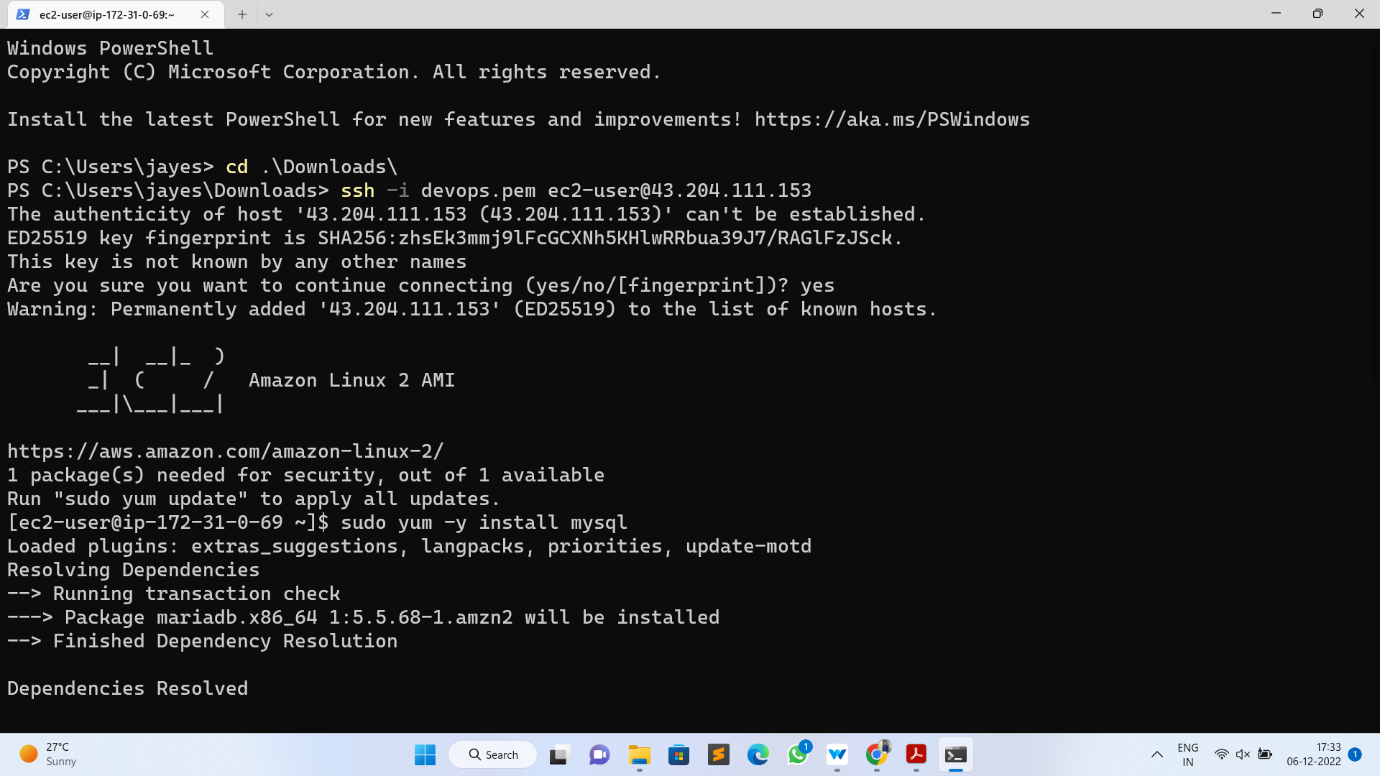
**For Mac or Linux users:**

* Open a terminal window. If you are on a Mac, you can use the default Terminal program that is installed, or you can use your own terminal.
* In your terminal, run the following commands to SSH into your instance. Replace the “<path/to/pem/file>” with the path to your file, e.g. “~/Downloads/wordpress.pem”, and the “<publicIpAddress>” with the public IP address for your EC2 instance.
* chmod 600 <path/to/pem/file>
* ssh -i <path/to/pem/file> ec2-user@<publicIpAddress>
* You should see the following in your terminal to indicate that you connected successfully:

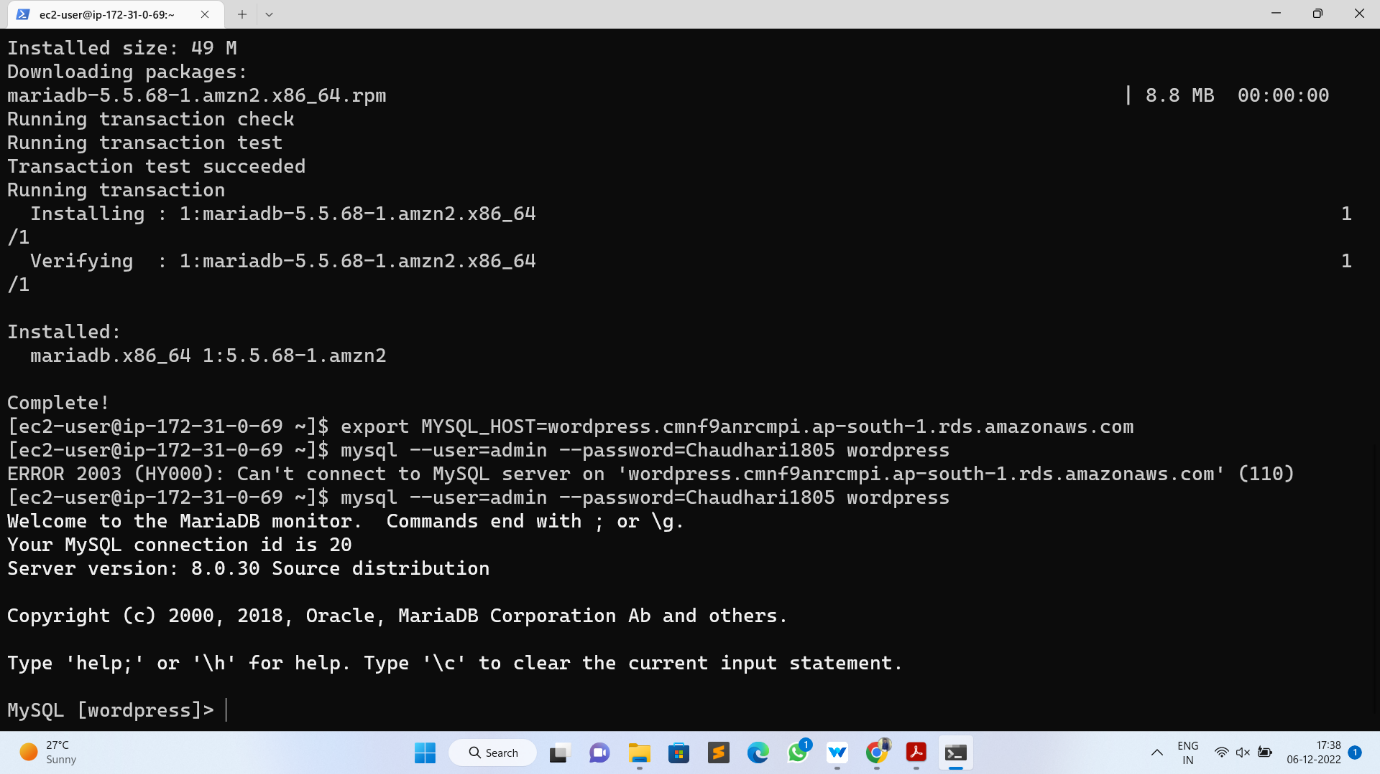


**Create a Database User**

* You should have an active SSH session to your EC2 instance in the terminal. Now, you will connect to your MySQL database.
* First, run the following command in your terminal to install a MySQL client to interact with the database.
* sudo yum install -y mysql



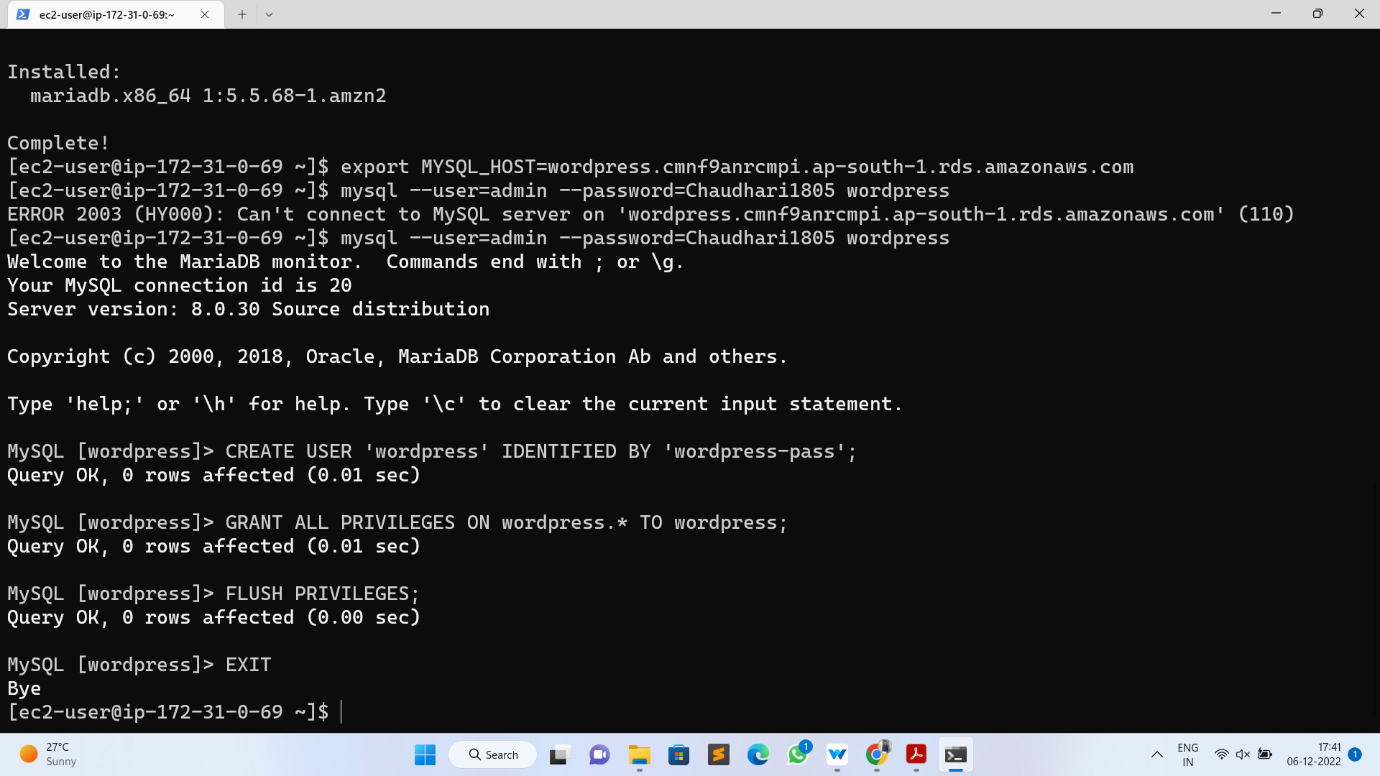
* Next, find the hostname for your RDS database in the AWS console. In the details of your RDS database, the hostname will be shown as the Endpoint in the Connectivity & security section.
* In your terminal, enter the following command to set an environment variable for your MySQL host. Be sure to replace “<your-endpoint>” with the hostname of your RDS instance.
* export MYSQL\_HOST=<your – rds – endpoint>
* Next, run the following command in your terminal to connect to your MySQL database. Replace “<user>” and “<password>” with the master username and password you configured when creating your RDS database.
* mysql --user=<user> --password=<password> wordpress
* If you connected successfully, your terminal should indicate connection to the MySQL database as shown in the following image.



* Finally, create a database user for your WordPress application and give it permission to access the “WordPress” database.

Run the following commands in your terminal:

* CREATE USER 'wordpress' IDENTIFIED BY 'wordpress-pass';
* GRANT ALL PRIVILEGES ON wordpress.\* TO wordpress;
* FLUSH PRIVILEGES;
* EXIT



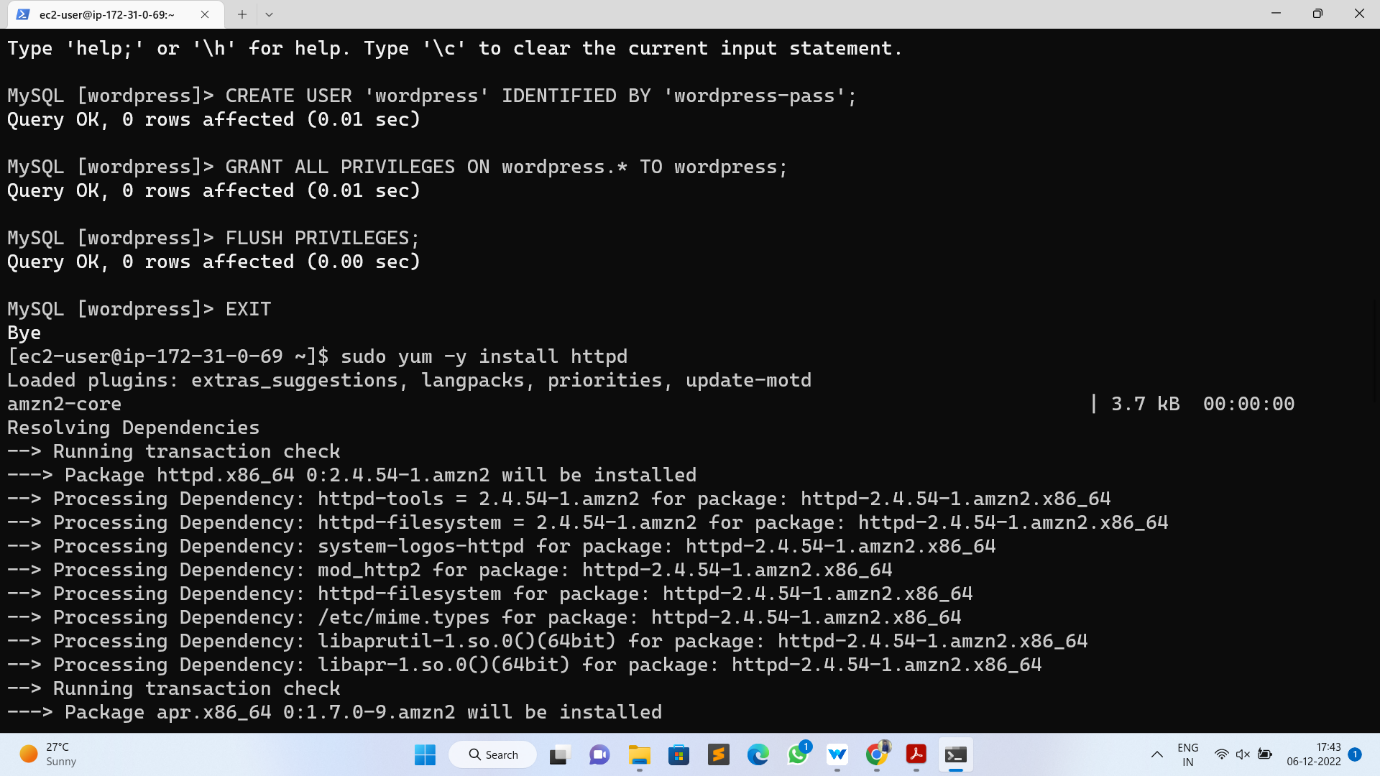
* You should use a better password than “WordPress-pass” to secure your database.
* Write down both the username and password that you configure, as it will be needed in the next module when setting up your WordPress installation.
* In this module, you learned how to configure network and password security for your RDS database. Your EC2 instance now has network access to your RDS database. Further, you created a database user to be used by your WordPress application.
* In the next module, you will configure your EC2 instance to run the WordPress application.

**Lab – 4: Configuring WordPress on EC2.**

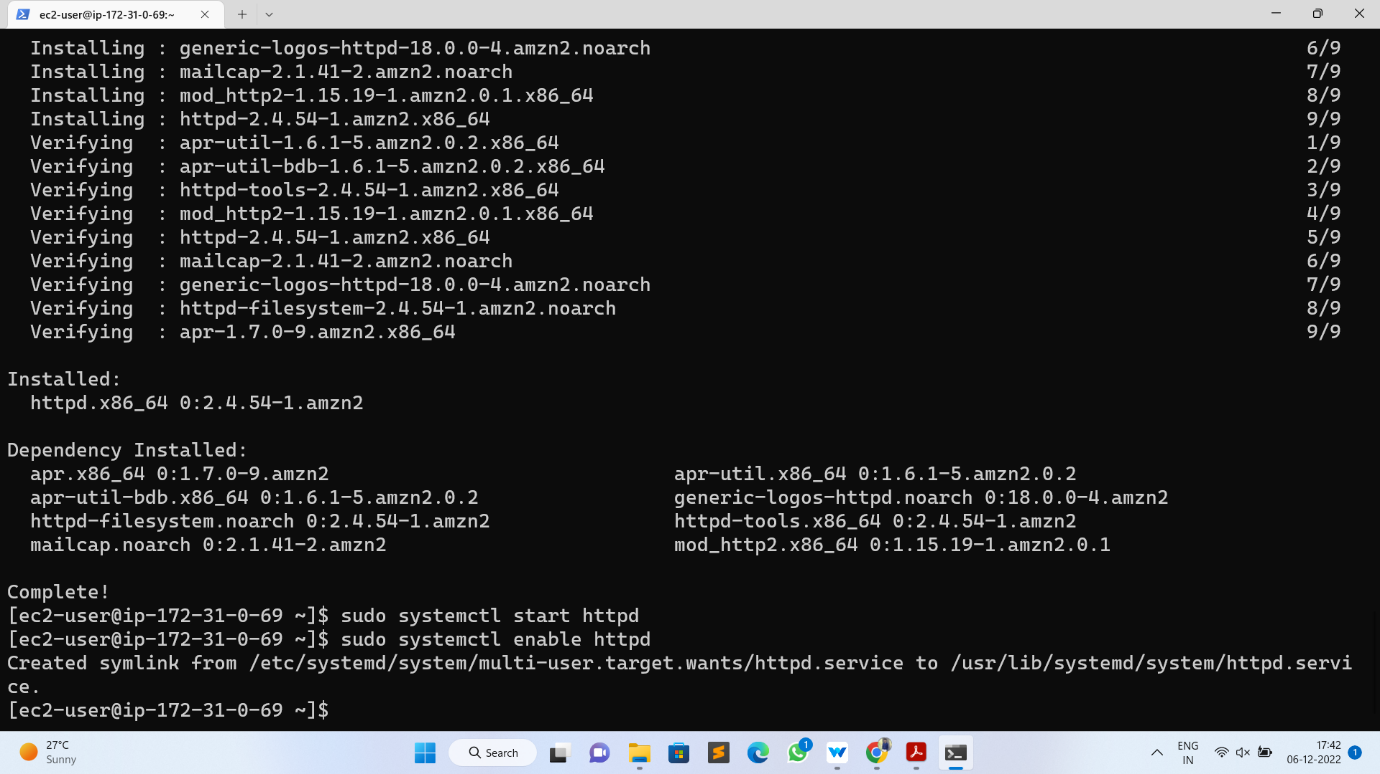
* To this point, you have done a lot of configuration setup. You created an RDS instance and an EC2 instance.
* You allowed network access from your EC2 instance to your RDS instance. You learned how to SSH into your EC2 instance and configured a database user to be used by WordPress.
* In this module, you will finish up the work to make your WordPress site live. You will install the WordPress application and dependencies on the EC2 instance.
* At the end of this module, you will have a WordPress installation that is accessible in the browser from anywhere in the world.
* To complete the steps in this module, you will need to SSH into your EC2 instance. Please review the steps in the previous module if you need to reconnect to your EC2 instance via SSH.
* Install the WordPress application and dependencies on the EC2 instance.

**Implementation**:

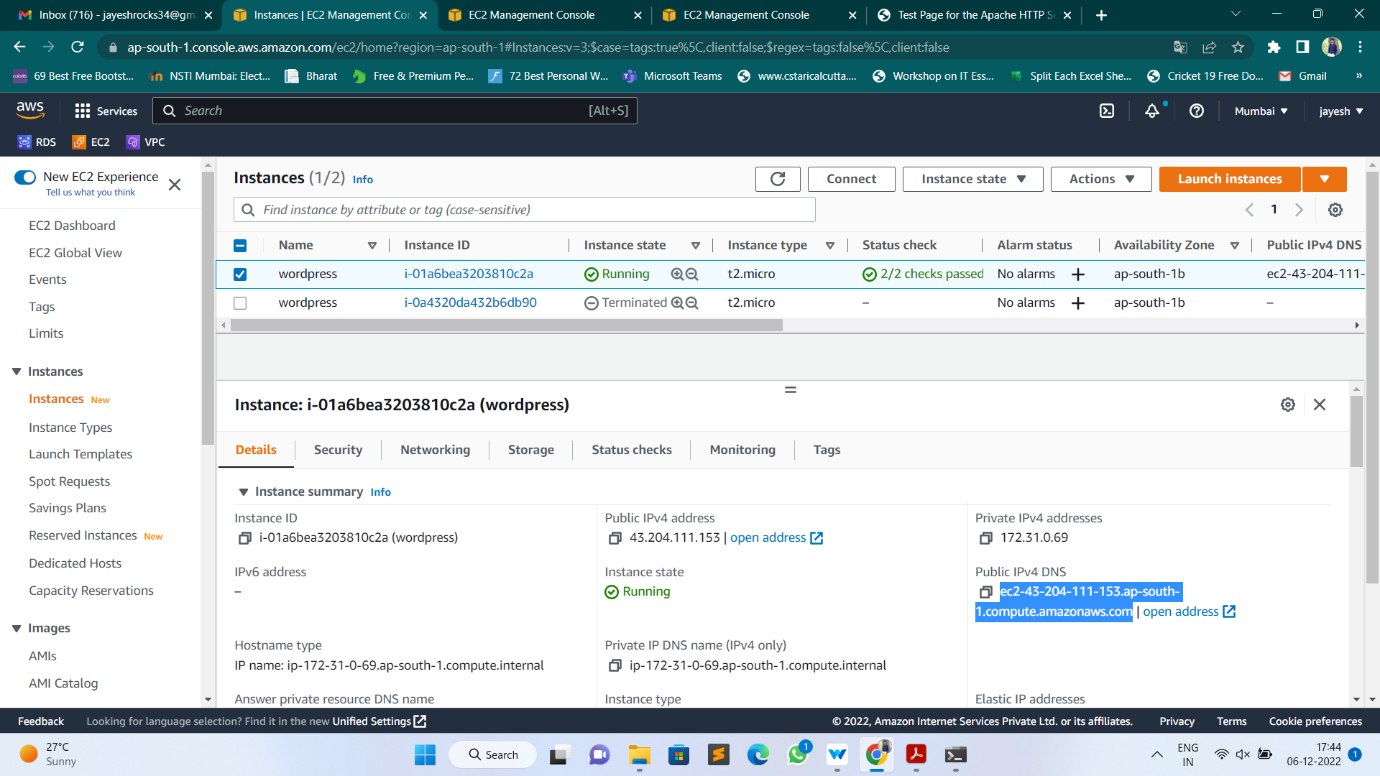
* Installing the Apache Web Server
* To run WordPress, you need to run a web server on your EC2 instance. The open source Apache web server is the most popular web server used with WordPress.
* To install Apache on your EC2 instance, run the following command in your terminal:
* sudo yum install -y httpd



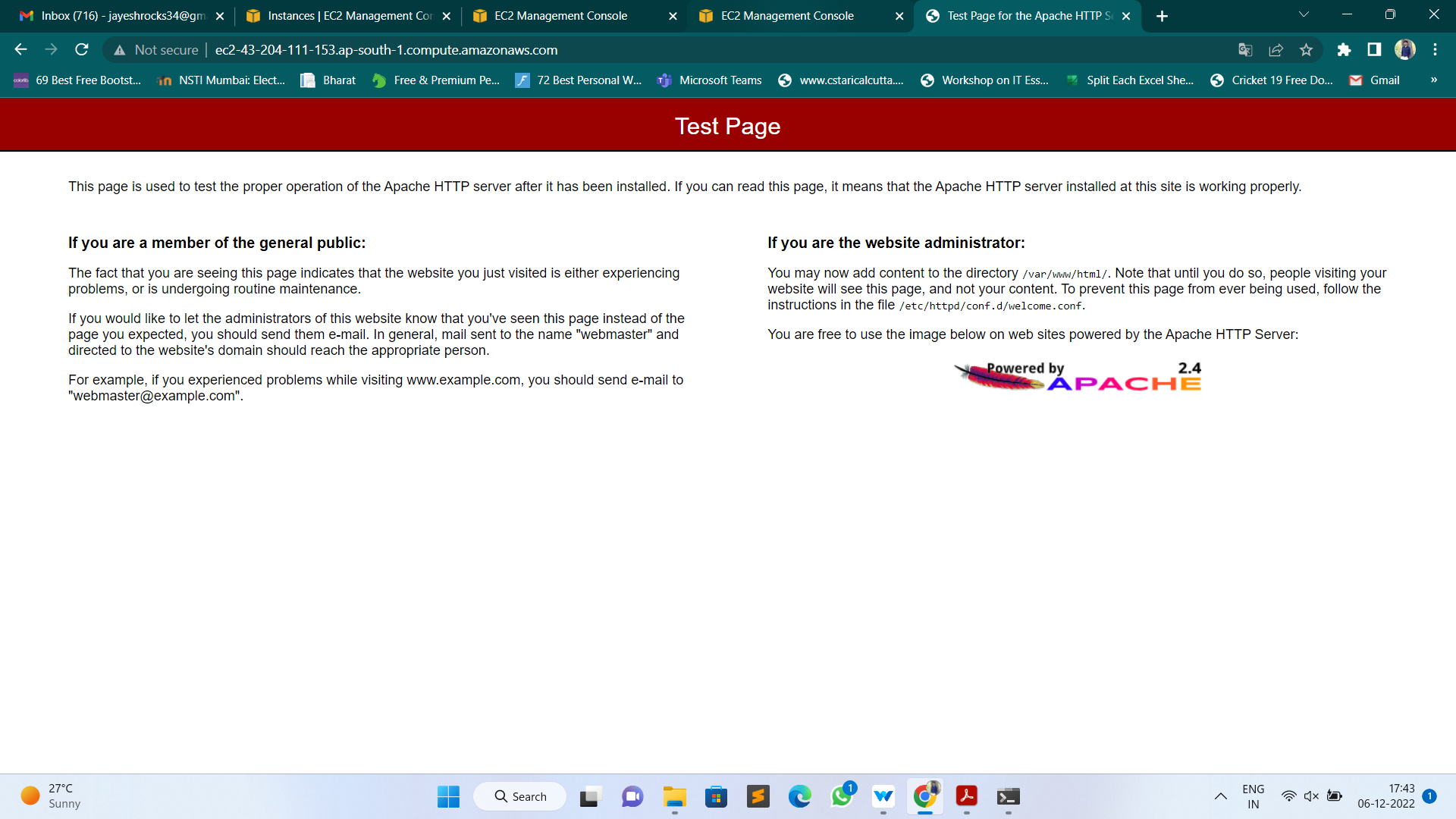
* You should see some terminal output of the necessary packages being installed.
* To start the Apache web server, run the following command in your terminal:
* sudo service httpd start



* You can see that your Apache web server is working and that your security groups are configured correctly by visiting the public DNS of your EC2 instance in your browser.
* Go to the EC2 Instances page and find your instance. In the Description below, find the Public DNS (IPv4) of your instance.



* Enter this value into your web browser, and you should see an Apache test page.



* Now that your Apache web server is working, it’s time to download and configure WordPress.

**Download and Configure WordPress**

* In this step, you will download the WordPress software and set up the configuration.
* First, download and uncompressing the software by running the following commands in your terminal:
* wget <https://wordpress.org/latest.tar.gz>
* tar -xzf latest.tar.gz
* If you run “ls” to view the contents of your directory, you will see a tar file and a directory called wordpress with the uncompressed contents.
* ls
* latest.tar.gz wordpress
* Change into the wordpress directory and create a copy of the default config file using the following commands:
* cd wordpress
* cp wp-config-sample.php wp-config.php
* Then, open the wp-config.php file using the vi editor by running the following command.
* vi wp-config.php
* You need to edit two areas of configuration.
* First, edit the database configuration by changing the following lines:

// \*\* MySQL settings - You can get this info from your web host \*\* //

/\*\* The name of the database for WordPress \*/

define( 'DB\_NAME', 'database\_name\_here' );

/\*\* MySQL database username \*/

define( 'DB\_USER', 'username\_here' );

/\*\* MySQL database password \*/

define( 'DB\_PASSWORD', 'password\_here' );

/\*\* MySQL hostname \*/

define(‘DB\_HOST', 'localhost' );

* The values should be:

DB\_NAME: “WordPress”

DB\_USER: The name of the user you created in the database in the previous module

DB\_PASSWORD: The password for the user you created in the previous module.

DB\_HOST: The hostname of the database that you found in the previous module

* The second configuration section you need to configure is the Authentication Unique Keys and Salts. It looks as follows in the configuration file:

/\*\*#@+

\* Authentication Unique Keys and Salts.

\*

\* Change these to different unique phrases!

\* You can generate these using the {@link https://api.wordpress.org/secret-key/1.1/salt/

WordPress.org secret-key service}

\* You can change these at any point in time to invalidate all existing cookies. This will force all

users to have to log in again.

\*

\* @since 2.6.0

\*/

define( 'AUTH\_KEY', 'put your unique phrase here' );

define( 'SECURE\_AUTH\_KEY', 'put your unique phrase here' );

define( 'LOGGED\_IN\_KEY', 'put your unique phrase here' );

define( 'NONCE\_KEY', 'put your unique phrase here' );

define( 'AUTH\_SALT', 'put your unique phrase here' );

define( 'SECURE\_AUTH\_SALT', 'put your unique phrase here' );

define( 'LOGGED\_IN\_SALT', 'put your unique phrase here' );

define( 'NONCE\_SALT', 'put your unique phrase here' );

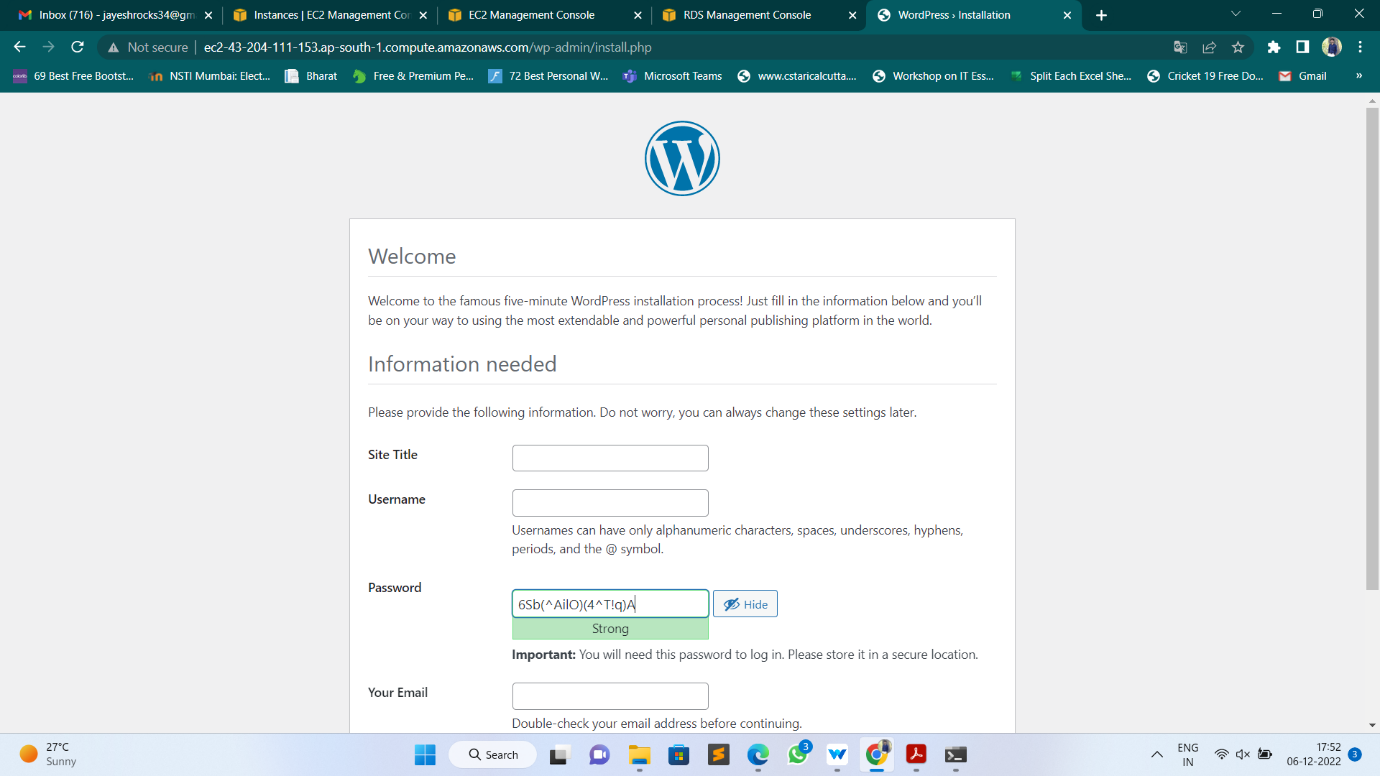
* Go to this link to generate values for this configuration section. You can replace the entire content in that section with the content from the link.

<https://api.wordpress.org/secret-key/1.1/salt/>

* You can save and exit from vi editor by entering Esc then type wq and hit Enter.
* With the configuration updated, you are almost ready to deploy your WordPress site. In the next step, you will make your WordPress site live.

**Deploying WordPress Site**

* In this step, you will make your Apache web server handle requests for WordPress.
* First, install the application dependencies you need for WordPress. In your terminal, run the following command.
* sudo amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
* Second, change to the proper directory by running the following command:
* cd /home/ec2-user
* Then, copy your WordPress application files into the /var/www/html directory used by Apache.
* sudo cp -r wordpress/\* /var/www/html/
* Finally, restart the Apache web server to pick up the changes.
* sudo service httpd restart
* You should see the WordPress welcome page and the five-minute installation process.



* That’s it. You have a live, publicly-accessible WordPress installation using a fully-managed MySQL database on Amazon RDS.
* In the next module, you will clean up your resources and see some next steps for your WordPress installation.