Module Description

Module 1

- In this module we are creating a linux machine on EC2 for that the followingsteps are to be followed:
  - We create a AWS account on https://aws.amazon.com or login if the account is already created.
  - Access the console of the AWS account as in figure 4.2 and find EC2 in the services tab and click on it.

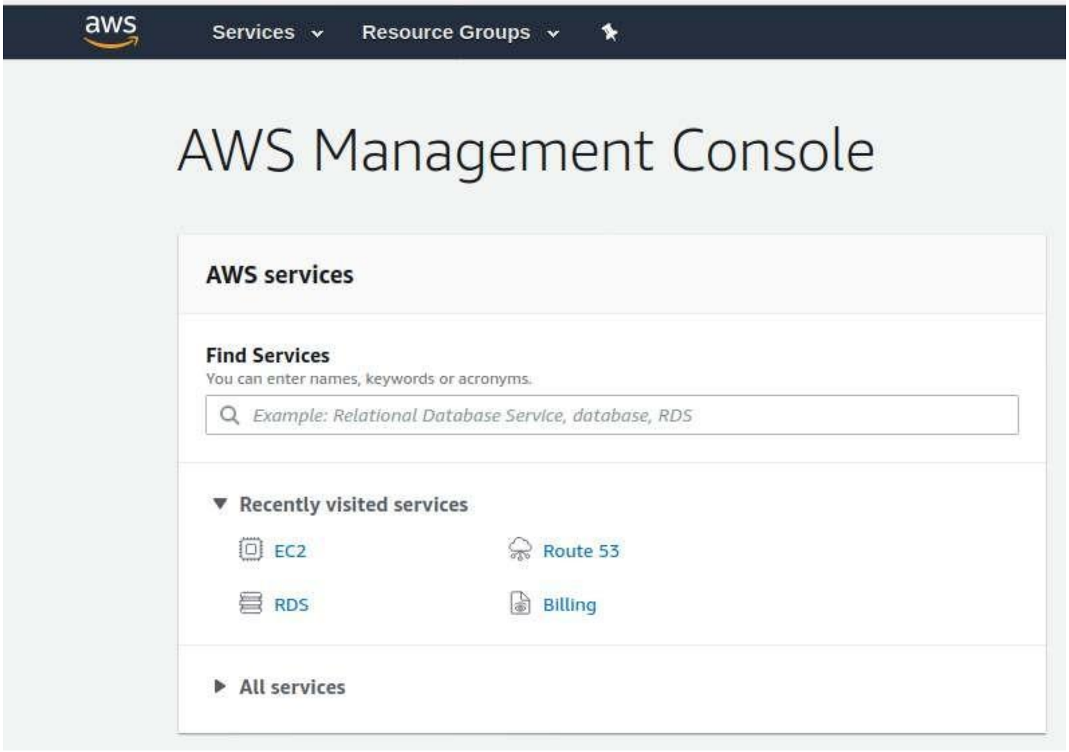
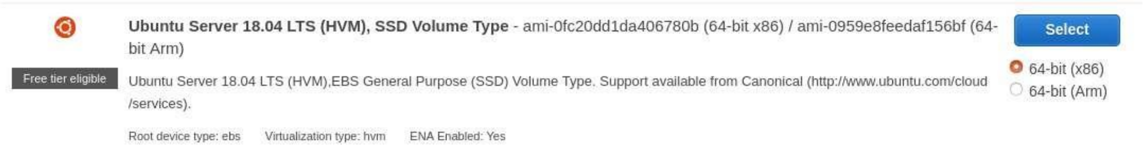

aws
Services
Resource Groups
AWS Management Console
AWS services
Find Services
You can enter names, keywords or acronyms.
Example: Relational Database Service, database, RDS
Recently visited services
EC2
Route 53
RDS
Billing
All services

Figure 4.2: AWS Console

  - Select the ubuntu ×64 bit machine AMI as in figure 4.3 and follow the instructions to finally create a EC2 instance and see weather its is live or not as in figure 4.4.

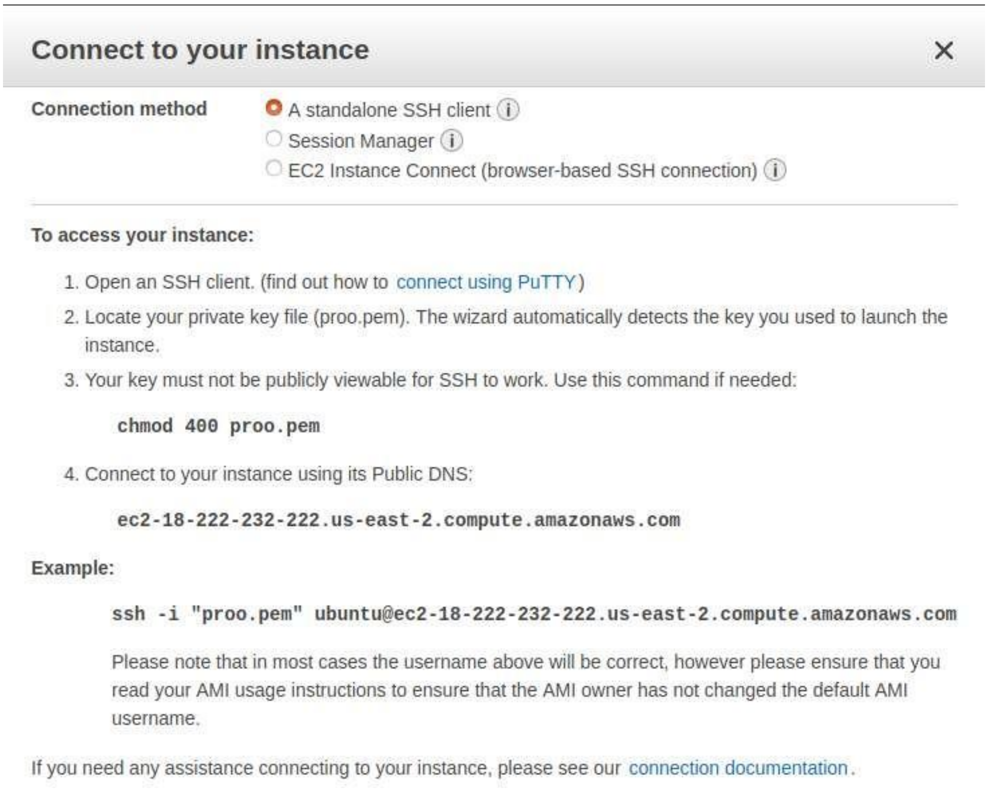


Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0fc20dd1da406780b (64-bit x86) / ami-0959e8feedaf156bf (64-bit Arm)

Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Figure 4.3: Ubuntu AMI



Connection method
● A standalone SSH client ⓘ
○ Session Manager ⓘ
○ EC2 Instance Connect (browser-based SSH connection) ⓘ

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (proo.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 proo.pem
```

4. Connect to your instance using its Public DNS:

```
ec2-18-222-232-222.us-east-2.compute.amazonaws.com
```

Example:

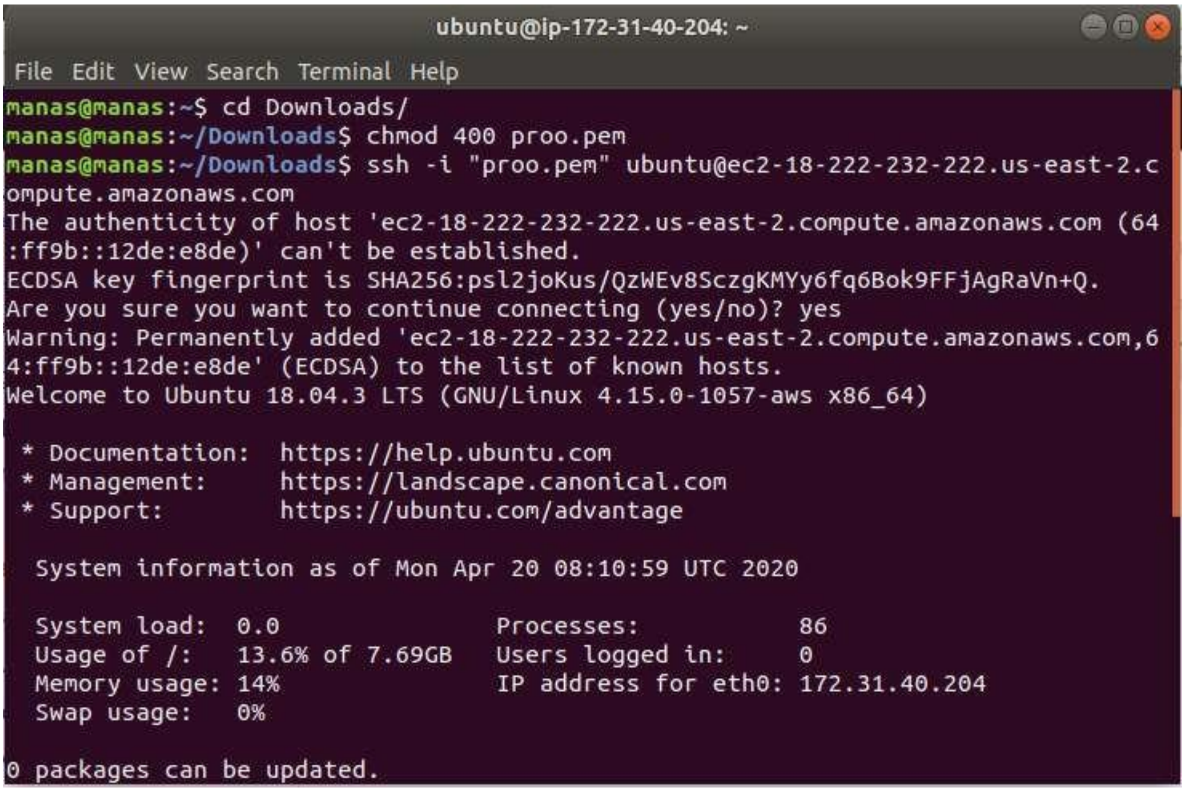
```
ssh -i "proo.pem" ubuntu@ec2-18-222-232-222.us-east-2.compute.amazonaws.com
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

Figure 4.4: EC2 Instance Is Live

– Launch the EC2 instance with the help of Secure Shell (SSH) as shown in figure 4.5



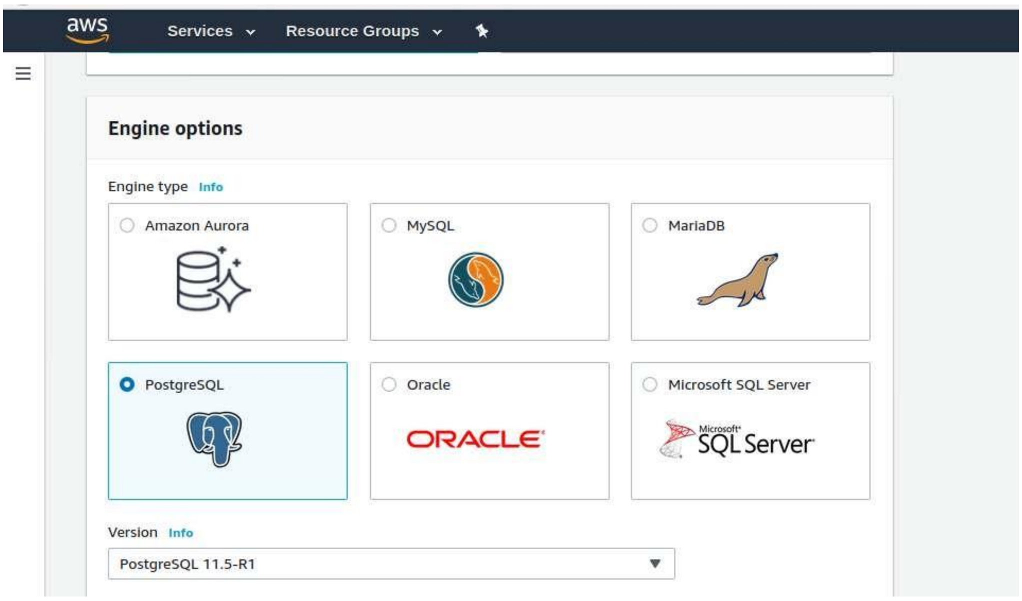
ubuntu@ip-172-31-40-204: ~
File Edit View Search Terminal Help

```
manas@manas:~$ cd Downloads/
manas@manas:~/Downloads$ chmod 400 proo.pem
manas@manas:~/Downloads$ ssh -i "proo.pem" ubuntu@ec2-18-222-232-222.us-east-2.c
ompute.amazonaws.com
The authenticity of host 'ec2-18-222-232-222.us-east-2.compute.amazonaws.com (64
:ff9b::12de:e8de)' can't be established.
ECDSA key fingerprint is SHA256:psl2joKus/QzWEv8SczgKMYy6fq6Bok9FFjAgRaVn+Q.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-18-222-232-222.us-east-2.compute.amazonaws.com,6
4:ff9b::12de:e8de' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1057-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Apr 20 08:10:59 UTC 2020

  System load:  0.0               Processes:           86
  Usage of /:   13.6% of 7.69GB   Users logged in:     0
  Memory usage: 14%               IP address for eth0: 172.31.40.204
  Swap usage:   0%

0 packages can be updated.
```

Figure 4.5: Connecting Through SSH

Module 2

- In Module 2 we are using the RDS service of the AWS for creating a postgresql database. The steps which are required to make a postgres database instance for storing the project login details and launching it on the AWS are as follows:
  - In the AWS console look for the RDS service of the AWS and click on it.
  - After selecting the RDS select the postgre sql option on the screen as shown in fig:4.6 and then the additional settings related to postgres will appear on the screen.



Engine options

Engine type Info

○ Amazon Aurora ○ MySQL ○ MariaDB

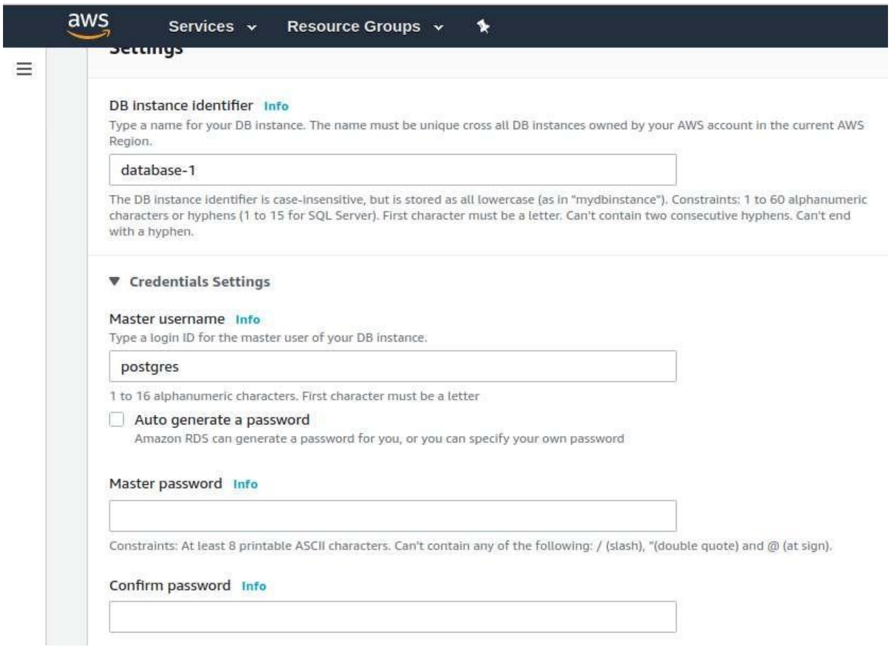
● PostgreSQL ○ Oracle ○ Microsoft SQL Server

Version Info

PostgreSQL 11.5-R1

Figure 4.6: Postgres And Other DB Options In AWS

  - Fill the essential details like master username and password for the postgres as in fig:4.7 and also select the same security group as of the EC2 instance.



DB instance identifier Info

Type a name for your DB instance. The name must be unique cross all DB instances owned by your AWS account in the current AWS Region.

database-1

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens (1 to 15 for SQL Server). 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.

postgres

1 to 16 alphanumeric characters. First character must be a letter

□ Auto generate a password

Amazon RDS can generate a password for you, or you can specify your own password

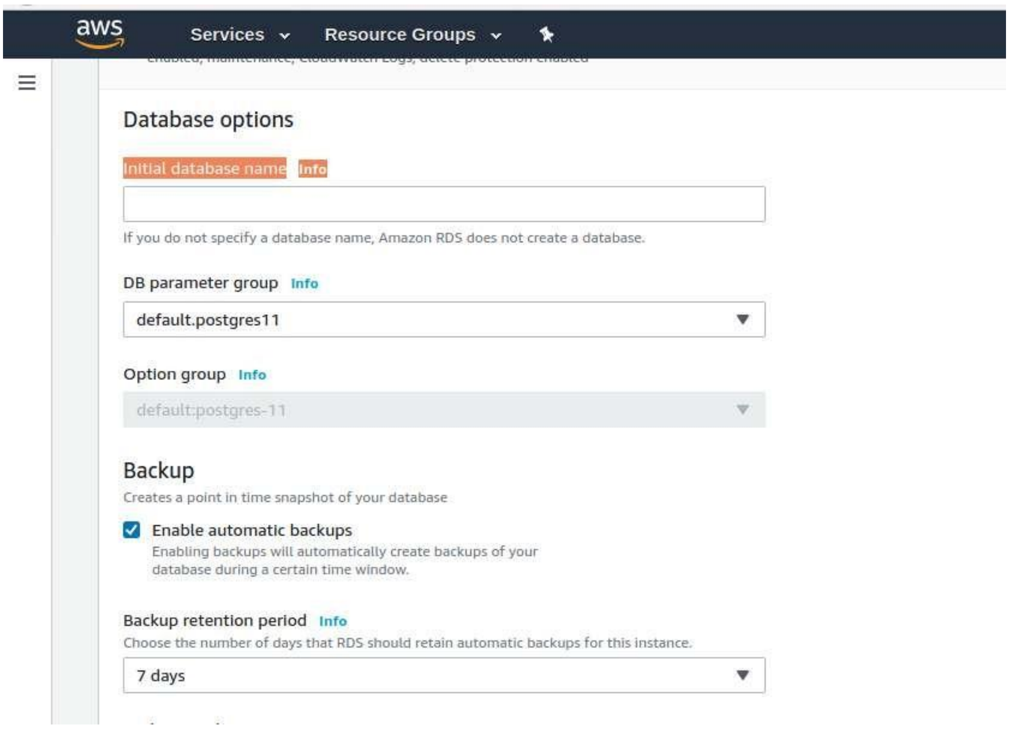
Master password Info

Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), "(double quote) and @ (at sign).

Confirm password Info

Figure 4.7: Postgres Master Username

- Enter the name of the database which you want to create in the postgres as in fig:4.8 and in which the data related to the project will be saved and then click on create instance which will create a postgres instance.



Database options

Initial database name Info

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group Info

default.postgres11

Option group Info

default:postgres-11

Backup

Creates a point in time snapshot of your database

☑ Enable automatic backups

Enabling backups will automatically create backups of your database during a certain time window.

Backup retention period Info

Choose the number of days that RDS should retain automatic backups for this instance.

7 days

Figure 4.8: Database Name And Details

- The postgres instance will provide the Amazon Resource Name (ARN) which is used to access the instance.

Module 3

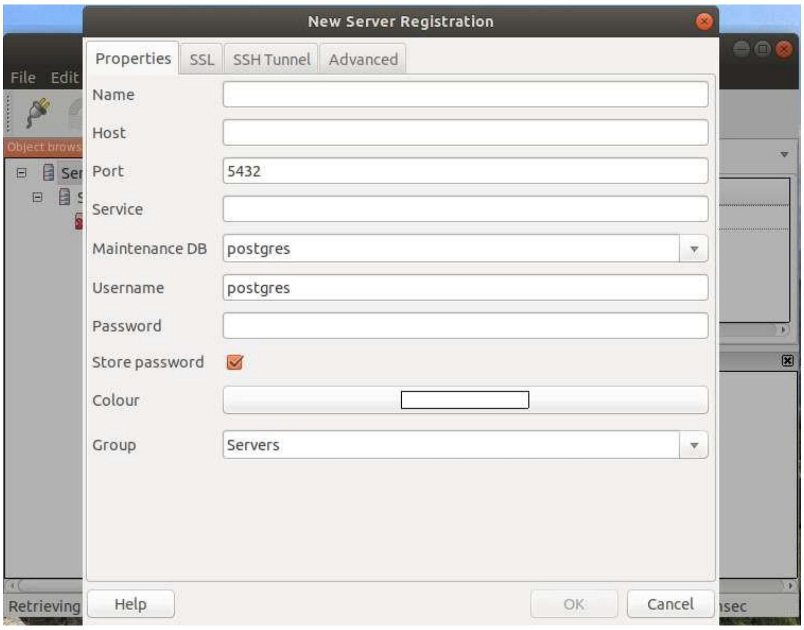
- The project when uploaded by the Secure Copy (SCP) command takes a lot of time so to solve this problem the project is first uploaded to a version control system like GIT or some storage service like AWS S3, dropbox or any other storage service. We have uploaded the project to dropbox and there are basic commands to download the poject from there to the EC2 instance. The very simple command "wget" is used to download the project in the form of a zip file and then the "unzip" command is used to extract the project.

  The project runs in the python environment so we have installed python3, pip and venv which is used to create a separate virtual environment for the project. In that virtual environment all the libraries which are required are installed and django is also installed to that the project can run without any error.

Module 4

- The module 4 is for accessing the database on the local machine by down- loading pgadmin. Pgadmin is used for the management of the postgres database when the postgres instance is formed on the AWS then it can be altered with the help of pgad-min with the help of ARN provides by the running RDS postgres instance. To access the remote database on the local machine the following steps are to be followed:

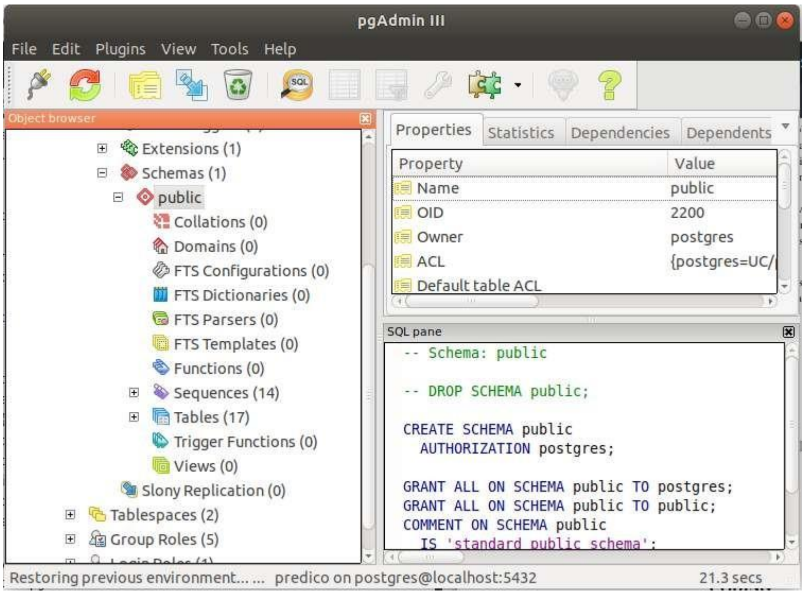
- Download the pgadmin from the official website of the Postgres in our case we are using ubuntu in the local system also so pgadmin 3 is available in the app store and can directly be downloaded.
- Open pgadmin and click on add connection button on the left corner.
- On clicking the add connection a new window will appear as shown in fig:4.9 that will ask for the name in which the master username has to be entered and the host in which the ARN of the instance is to be entered and the password is also entered.



Name	
Host	
Port	5432
Service	
Maintenance DB	postgres
Username	postgres
Password	
Store password	☑
Colour	
Group	Servers

Figure 4.9: Pgadmin Details To Access Instance

- After filling all the details connect to the instance and then the DB instance will be open as shown in fig:4.10 and the user can alter the details as per his or her choice.



Extensions (1)
Schemas (1)
public
Collations (0)
Domains (0)
FTS Configurations (0)
FTS Dictionaries (0)
FTS Parsers (0)
FTS Templates (0)
Functions (0)
Sequences (14)
Tables (17)
Trigger Functions (0)
Views (0)
Slony Replication (0)
Tablespaces (2)
Group Roles (5)

Property	Value
Name	public
OID	2200
Owner	postgres
ACL	{postgres=UC/
Default table ACL	

```
-- Schema: public

-- DROP SCHEMA public;

CREATE SCHEMA public
  AUTHORIZATION postgres;

GRANT ALL ON SCHEMA public TO postgres;
GRANT ALL ON SCHEMA public TO public;
COMMENT ON SCHEMA public
  IS 'standard public schema';
```

Restoring previous environment... ... predico on postgres@localhost:5432 21.3 secs

Figure 4.10: Postgres Instance In Pgadmin

- The postgres instance which is accessed by the pgadmin also has a pre-createddatabase in it for which the name was given while creating the db instance on AWS.

Module 5

- This module is is about creating the migrations. Migrations help in creating the schema of the tables required in the database. Before starting the project migrations are to be made so that the schema of all the tables which are required by the project comes in the database on itself. These migrations are based on the models.py file in the project, in models.py the class of the table is created and the fields are set as an objectin that class all this process is known as the Object Relational Model (ORM). ORM is avery helpful technique in django which helps in very simple and easy creation of tables in the database and for pushing the data into the database we then also create a object which insert the data into the database.

  For running the project manage.py file is used, Django has its inbuilt web server for running the project. As soon as we start the web server the project will be up and runningand the post number set in django web server is 8000.

Module 6

- When the project is running on the web server of the django then it can be accessed globally with the help of the public ip of the EC2 instance, the public ip of theEC2 instance in given in the instance details we just have to open the browser on any mobile phone or PC and just have to type that ip followed by the port number 8000 as the django web server runs on that particular port only.